Title of Thesis

Proposals of Multiplication and Inversion Methods in Extension Field for Scalable Asymmetric–key and Fast Symmetric–key Cryptosystems

March, 2013

Kenta NEKADO

The Graduate School of
Natural Science and Technology
(Doctor's Course)
OKAYAMA UNIVERSITY

Acknowledgment

First, I hope to express my warm and sincere gratitude to Associate Professor Yasuyuki Nogami, who is my supervisor. His warm supports and sincere encouragements made enormous contribution to my research activities. And, for earning a degree of doctor's course and writing this thesis, I received generous supports from Professor Nobuo Funabiki and Associate Professor Toru Nakanishi, who are my cosupervisors. I am deeply grateful to them. I would particularly like to thank to Emeritus Professor Yoshitaka Morikawa, who is my former supervisor, and gives me a lot of constructive comments and much warm encouragement.

I also wish to thank the other teachers, especially Professor Satoshi Denno and Associate Professor Nobumoto Yamane, who belong to my common laboratory which consists of Multimedia Radio System Laboratory, Information Transfer Laboratory and Secure Wireless System Laboratory, Okayama University. Special thanks also to my seniors, juniors, and friends in the common laboratory for creating a great work atmosphere.

I am also grateful to administrative officers Ms. Yumiko Kurooka and Ms. Midori Onishi, who have warm relationships with me.

Finally, I wish to appreciate my family for their understanding, endless patience, and encouragements.

Research Activity

Refereed Papers

1. **K. Nekado**, Y. Nogami, H. Kato, and Y. Morikawa, "Cyclic Vector Multiplication Algorithm and Existence Probability of Gauss Period Normal Basis," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E94–A, No. 1, pp. 172–179, Jan., 2011.
2. Y. Nogami, **K. Nekado**, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed Bases for Efficient Inversion in $\mathbb{F}_{(2^2)^2)^2}$ and Conversion Matrices of SubBytes of AES," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E94–A, No. 6, pp. 1318–1327, June, 2011.

Other Paper (with Review)

1. Y. Nogami, H. Kato, **K. Nekado**, S. Uehara, and Y. Morikawa, "Finding a Basis Conversion Matrix Using a Polynomial Basis Derived by a Small Multiplicative Cyclic Group," IEEE Transactions on Information Theory, Vol. 58, No. 7, pp. 4936–4947, June 2012.

Other Papers (without Review)

1. **K. Nekado**, H. Kato, Y. Nogami, and Y. Morikawa, "Efficient Squaring Algorithm for Xate Pairing with Freeman Curve," Memoirs of the Faculty of Engineering, Okayama university, Vol. 44, pp. 69–72, available at '`http://www.eng.okayama-u.ac.jp/up_load_files/kiyou/44/No9.pdf`', 2010.
2. **K. Nekado**, Y. Takai, Y. Nogami, and Y. Morikawa, "Squaring Algorithm Efficient for Cubic Extension Field Derived with Pseudo Gauss Period Normal Basis," Memoirs of the Faculty of Engineering, Okayama university, Vol. 45, pp. 54–59, available at '`http://www.eng.okayama-u.ac.jp/up_load_files/kiyou/45/No06.pdf`', 2011.
3. **K. Nekado**, Y. Takai, and Y. Nogami, "Lazy Random Walk Efficient for Pollard's Rho Method Attacking on $\mathbb{G}_3$ over Barreto–Naehrig Curve (Corrected)," Memoirs of the Faculty of Engineering, Okayama university, Vol. 47, pp. 26–32, available at '`http://www.eng.okayama-u.ac.jp/up_load_files/kiyou/47/No03.pdf`', 2013.

International Conferences (with Review)

1. Y. Nogami, **K. Nekado**, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed Bases for Efficient Inversion in $\mathbb{F}_{((2^2)^2)^2}$ and Conversion Matrices of SubBytes of AES," Workshop

on Cryptographic Hardware and Embedded Systems 2010 (CHES2010), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 6225, pp. 234–247, Aug. 19, 2010, in California–state, USA. (Acceptance ratio: 30/108 ≈ 27.8%)

2. **K. Nekado**, T. Yuasa, Y. Nogami, and Y. Morikawa, "Efficient Squaring Algorithm in 2–nd Tower Field Available for Various Pairing–based Cryptographies," First International Workshop on Trustworthy Computing (TwC2010), CD–ROM, No. TwC–2–3, Sep. 14, 2010, in Gifu–pref., Japan.

3. **K. Nekado**, Y. Nogami, and Y. Morikawa, "Ultimately Customized Multiplication Algorithm in the Extension Field for Xate and R–ate Pairing with Freeman Curve," The 2010 IEEE Region 10 Conference (TENCON2010), CD–ROM, No. T4–3.3, Nov. 23, 2010, in Fukuoka–pref., Japan. (Acceptance ratio: 586/695 ≈ 84.3%)

4. Y. Nogami and **K. Nekado**, "A Multiplication Algorithm with Square–free Gauss Period Normal Basis," 8th International Conference on Computing Technology and Information Management (ICCM2012), Vol. 1, NCM Track, pp. 136–140, Apr. 25, 2012, in Korea, Seoul.

5. R. Takahashi, Y. Takai, **K. Nekado**, Y. Nogami, H. Kagotani, and T. Narita, "Memory–saving and Efficient Implementation of Cyclic Vector Multiplication Algorithm with Gauss Period Normal Basis," The 27th International Technical Conference on Circuits/Systems, Computers and Communications (ITC–CSCC2012), CD–ROM, No. P–M2–22, July 16, 2012, in Hokkaido–pref., Japan.

6. Y. Mori, T. Sumou, **K. Nekado**, Y. Nogami, and S. Uehara, "Memory Saving Implementation of Pollard's Rho Method," The 27th International Technical Conference on Circuits/Systems, Computers and Communications (ITC–CSCC2012), CD–ROM, No. F–W1–01, July 18, 2012, in Hokkaido–pref., Japan.

7. **K. Nekado**, Y. Mori, T. Sumou, and Y. Nogami, "Representative Decision Efficient for Pollard's Rho Method on $\mathbb{G}_2$ over Barreto–Naehrig Curve," The 27th International Technical Conference on Circuits/Systems, Computers and Communications (ITC–CSCC2012), CD–ROM, No. F–W1–03, July 18, 2012, in Hokkaido–pref., Japan.

8. **K. Nekado**, Y. Nogami, and K. Iokibe, "Very Short Critical Path Implementation of AES with Direct Logic Gates," The 7th International Workshop on Security (IWSEC2012), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 7631, pp. 51–68, Nov. 7, 2012, in Fukuoka–pref., Japan. (Acceptance ratio: 16/53 ≈ 30.2%)

9. Y. Takai, **K. Nekado**, and Y. Nogami, "The Pollard's Rho Method with XTR Group on $\mathbb{G}_3$ over Barreto–Naehrig Curve," 7th International Conference on Computer Sciences and Convergence Information Technology (ICCIT2012), DVD–ROM, pp. 619–622, Dec. 3, 2012, in Seoul, Korea.

Domestic Conferences

1. **根角 健太**, 柳 枝里佳, 吉田 知輝, 那須 弘明, 野上 保之, 森川 良孝, "Freeman 曲線を用いた Xate ペアリングに適した拡大体の構成法," 暗号と情報セキュリティシンポジウム 2009（SCIS2009）, 予稿集 CD-ROM, No. 3C4–1, 2009/1/22, in 大津プリンスホテル（滋賀県）.

2. 加藤 英洋, **根角 健太**, 柳 枝里佳, 野上 保之, 森川 良孝, “ペアリング計算での利用を考慮した拡大体上 2 乗算の改良,” 暗号と情報セキュリティシンポジウム 2009（SCIS2009）, 予稿集 CD–ROM, No. 3C3–5, 2009/1/22 in 大津プリンスホテル（滋賀県）.

3. 柳 枝里佳, **根角 健太**, 野上 保之, 森川 良孝, “ガウス周期正規基底に基づく乗算アルゴリズム CVMA の改良,” 情報セキュリティ研究会 (ISEC), 電子情報通信学会技術研究報告書（IEICE Technical Report）, Vol. 109, No. 42, pp. 55–60, 2009/5/22, in 機械振興会館（東京都）.

4. **根角 健太**, 湯浅 達也, 野上 保之, 森川 良孝, “Freeman 曲線を用いた Xate および R–ate ペアリングのための定義体における乗算アルゴリズム,” コンピュータセキュリティシンポジウム 2009（CSS2009）, 論文集, pp. 39–44, 2009/10/26, in 富山国際会議場（富山県）.

5. 湯浅 達也, 柳 枝里佳, **根角 健太**, 西井 一志, 竹内 翔一, 野上 保之, 森川 良孝, “$\mathbb{F}_{p^4}$ の Type–I ONB を用いた 2 次逐次拡大体 $\mathbb{F}_{(p^2)^2}$ の構成とその効率的な乗算の実装,” 第 32 回情報理論とその応用シンポジウム（SITA2009）, 予稿集 CD–ROM, No. W35–2, 2007/12/2, in ホテルかめ福（山口県）.

6. **根角 健太**, 野上 保之, 森川 良孝, “Pippenger 法と Montgomery トリックを効率よく用いたマルチスカラ倍算アルゴリズム,” 暗号と情報セキュリティシンポジウム 2010（SCIS2010）, 予稿集 CD–ROM, No. 2C1–3, 2010/1/20, in サンポート高松（香川県）.

7. 加藤 英洋, **根角 健太**, 野上 保之, 森川 良孝, “暗号応用を目的とした TypeI 最適正規基底を用いた拡大体上 2 乗算に関する一考察,” 暗号と情報セキュリティシンポジウム 2010（SCIS2010）, 予稿集 CD–ROM, No. 3A2–3, 2010/1/20, in サンポート高松（香川県）.

8. **根角 健太**, 野上 保之, 森川 良孝, “ガウス周期正規基底の存在確率,” 情報理論研究会（IT）, 電子情報通信学会技術研究報告書（IEICE Technical Report）, Vol. 109, No. IT–444, pp. 403–407, 2010/3/5, in 信州大学（長野県）.

9. 高井 悠輔, **根角 健太**, 野上 保之, 森川 良孝, 籠谷 裕人, “MRCP の分類とその素体上の乗算への循環ベクトル乗算アルゴリズムの適用,” 情報理論研究会（IT）, 電子情報通信学会技術研究報告書（IEICE Technical Report）, Vol. 110, No. 137, pp. 19–24, 2010/7/22, in 工学院大学（東京都）.

10. 村上 拓, **根角 健太**, 野上 保之, 森川 良孝, “AES の SubBytes における $\mathbb{F}_{(2^4)^2}$ を用いた効率的な逆元計算,” コンピュータセキュリティシンポジウム 2010（CSS20010）, 論文集, pp. 345–350, 2010/10/20, in 岡山コンベンションセンター（岡山県）.

11. 高井 悠輔, **根角 健太**, 野上 保之, 森川 良孝, “MRCP の分類とこれを法とする素体上乗算へ循環ベクトル乗算アルゴリズムを適用した場合の性能評価,” 平成 22 年度電気情報関連学会中国支部第 61 回連合大会, 予稿集 CD–ROM, pp. 534–535, 2010/10/23, in 岡山県立大学（岡山県）.

12. **根角 健太**, 野上 保之, 森川 良孝, “ガウス周期正規基底の存在確率に関する定理とその証明,” 第 33 回情報理論とその応用シンポジウム（SITA2010）, 予稿集 CD–ROM, No. 25.2, 2010/12/2, in 信州松代ロイヤルホテル（長野県）.

13. **根角 健太**, 加藤 英洋, 野上 保之, 森川 良孝, “循環ベクトル乗算アルゴリズムの理論的な評価とその改良,” 第 33 回情報理論とその応用シンポジウム（SITA2010）, 予稿集 CD–ROM, No. 25.3, 2010/12/2, in 信州松代ロイヤルホテル（長野県）.

14. 湯浅 達也, **根角 健太**, 野上 保之, 森川 良孝, "Type $\langle k, 4\rangle$ GNB を用いた 2 次逐次拡大体 $\mathbb{F}_{(p^2)^2}$ の構成とその効率的な乗算の実装," 第 33 回情報理論とその応用シンポジウム（SITA2010）, 予稿集 CD–ROM, No. 35.3, 2010/12/2, in 信州松代ロイヤルホテル（長野県）.

15. 高井 悠輔, **根角 健太**, 野上 保之, 森川 良孝, "標数次拡大体における効率の良い乗算アルゴリズム," 暗号と情報セキュリティシンポジウム 2011（SCIS2011）, 予稿集 CD–ROM, No. 2C4–3, 2011/1/26, in リーガロイヤルホテル小倉（福岡県）.

16. 高橋 龍介, **根角 健太**, 高井 悠輔, 野上 保之, 籠谷 裕人, 成田 隆, "循環ベクトル乗算アルゴリズムの省メモリ実装," 情報セキュリティ研究会（ISEC）, 電子情報通信学会技術研究報告書（IEICE Technical Report）, Vol. 111, No. 123, pp. 145–150, 2011/7/13, in 静岡大学浜松キャンパス（静岡県）.

17. **根角 健太**, 野上 保之, 森岡 恵理, "冗長表現基底による $\mathbb{F}_{(2^4)^2}$ 上の逆元計算を用いた AES の SubBytes 変換," コンピュータセキュリティシンポジウム 2011（CSS2011）, 予稿 CD–ROM, No. 2C2–4, 2011/10/20, in 朱鷺メッセ（新潟県）.

18. **根角 健太**, 野上 保之, 森岡 恵理, $\mathbb{F}_{(2^4)^2}$ 上の複雑混合基底による基底変換を用いた AES の SubBytes 変換," コンピュータセキュリティシンポジウム 2011（CSS2011）, 予稿 CD-ROM, No. 2C2–5, 2011/10/20, in 朱鷺メッセ（新潟県）.

19. **根角 健太**, 野上 保之, "3 次 All One Polynomial Field 上で効率の良い自乗算アルゴリズム," 平成 23 年度電気情報関連学会第 62 回中国支部連合大会, 予稿集 CD–ROM, pp. 497–498, 2011/10/22, in 広島工業大学（広島県）.

20. 森岡 恵理, **根角 健太**, 野上 保之, "逐次拡大体 $\mathbb{F}_{(2^4)^2}$ を用いた AES の SubBytes における効率の良い演算," 平成 23 年度電気情報関連学会第 62 回中国支部連合大会, 予稿集 CD–ROM, pp. 120–121, 2011/10/22, in 広島工業大学（広島県）.

21. 森 佑樹, **根角 健太**, 野上 保之, "BN 曲線を用いたペアリングの NTL による iPhone 実装," 暗号と情報セキュリティシンポジウム 2012（SCIS2012）, 予稿集 CD–ROM, No. 1B1–2, 2012/1/30, in 金沢エクセルホテル東急（石川県）.

22. **根角 健太**, 森岡 恵理, 野上 保之, "$\mathbb{F}_{(2^4)^2}$ 上の逆元計算を用いた AES 用 SubBytes 変換回路の小型化," 暗号と情報セキュリティシンポジウム 2012（SCIS2012）, 予稿集 CD–ROM, No. 1C2–1, 2012/1/30, in 金沢エクセルホテル東急（石川県）.

23. 河野 祐輝, **根角 健太**, 森 佑樹, 有井 智紀, 野上 保之, "BN 曲線における $\mathbb{G}_2$ 上の ρ 法に関する効率的な代表元決定法," 情報理論研究会（IT）, 電子情報通信学会技術研究報告書（IEICE Technical Report）, No. 9, pp. 1–6, 2012/7/19, in 豊田工業大学（愛知県）.

24. 有井 智紀, **根角 健太**, 野上 保之, "ツイスト曲線上の有理点に対する有理点ノルムの性質と Rho 法への応用," コンピュータセキュリティシンポジウム 2012（CSS2012）, 予稿 CD–ROM, No. 2C3–3, 2012/10/31, in くにびきメッセ（島根県）.

25. 赤木 晶一, 森 佑樹, **根角 健太**, 野上 保之, "OEF を用いた Barreto–Naehrig 曲線上での Xate ペアリング実装," 第 35 回情報理論とその応用シンポジウム（SITA2012）, 予稿集 CD–ROM, No. 3.4.3, pp. 223–228, 2012/12/12, in 別府湾ロイヤルホテル（大分県）.

26. 河野 祐輝, 有井 智紀, **根角 健太**, 野上 保之, "Barreto–Naehrig 曲線適用時の $\mathbb{G}_2$ を攻撃対象とした Pollard の Rho 法に対する効率の良い代表元決定," 第 35 回情報理論とその応用シンポジウム（SITA2012）, 予稿集 CD–ROM, No. 8.4.2, pp. 623–627, 2012/12/14, in 別府湾ロイヤルホテル（大分県）.

27. **根角 健太**, 高井 悠輔, 森 佑樹, 野上 保之, “Barreto–Naehrig 曲線適用時の $\mathbb{G}_3$ を攻撃対象とした Pollard の Rho 法に対して効率の良いものぐさランダムウォーク,” 第 35 回情報理論とその応用シンポジウム（SITA2012）, 予稿集 CD–ROM, No. 8.4.3, pp. 628–633, 2012/12/14, in 別府湾ロイヤルホテル（大分県）.

28. 森 佑樹, 赤木 晶一, **根角 健太**, 野上 保之, “BN 曲線を用いたペアリングの GMP による iPhone 実装,” 暗号と情報セキュリティシンポジウム 2013（SCIS2013）, 予稿集 CD–ROM, No. 2E3–1, 2013/1/23, in ウェスティン都ホテル京都（京都府）.

Abstract

This thesis proposes the *Cyclic Vector Multiplication Algorithm* (CVMA) for Gauss period Normal Basis (GNB). It is an efficient multiplication algorithm in extension field which is flexible for the restriction and scalability of the extension field parameters required by next generation asymmetric–key cryptosystems. Additionally, this thesis also proposes *Redundantly Represented Basis* (RRB) and *More Miscellaneously Mixed Bases* (MMMB) in order to accelerate the computations of several symmetric–key cryptosystems such as Advanced Encryption Standard (AES).

Recently, pairing–based cryptosystems and their applications have attracted much attentions as next generation asymmetric–key cryptosystems. In order to accelerate the computations of these cryptosystems, not only pairing computations but also arithmetic operations, especially multiplications, in the extension field need to be improved. On the other hand, the cryptosystems often restrict the parameters of the extension field $\mathbb{F}_{p^m}$, namely the characteristic p and the extension degree m. Thus, the cryptosystems require an efficient multiplication algorithm which fast performs multiplications in the extension field and is flexible for the above parameters. Several types of CVMAs have been proposed for these demands, and they adopt special classes of GNBs. GNB and its special classes are characterized with a certain positive integer h in addition to p and m. The parameter h needs to satisfy some conditions, and there infinitely exists such h for each pair of p and m; however, such a practical h is limited because the conventional CVMAs become more inefficient as h is larger. In some cases, GNB has the smaller h for p and m than its special classes. Thus, in order to utilize the practical h in more situations, this thesis improves CVMA for GNB. Then, this CVMA acquires the higher flexibility for the parameters of the extension field than the conventional ones. Additionally, in order to demonstrate the flexibility of the improved CVMA, this thesis also proposes an important theorem to derive the existence probability of GNB for any h. According to this theorem, it is theoretically shown that the improved CVMA has the high flexibility.

In the field of symmetric–key cryptosystems, a lot of improvements and optimizations have been reported for the hardware implementation of AES cipher and its similarities. In order to accelerate SubBytes and InvSubBytes of AES which are the most complex procedures, many of these implementations often utilize inversions in the isomorphic towering field (composite field) $\mathbb{F}_{((2^2)^2)^2}$ or $\mathbb{F}_{(2^4)^2}$, instead of those in the AES original $\mathbb{F}_{2^8}$. This thesis focuses on $\mathbb{F}_{(2^4)^2}$ which provides higher–speed inversions than $\mathbb{F}_{((2^2)^2)^2}$, and proposes RRB technique which accelerates the inversions. Within the author's knowledge, the best conventional implementations perform an inversion in $\mathbb{F}_{(2^4)^2}$ at $4T_{\mathrm{AND}} + 10T_{\mathrm{XOR}}$. On the other hand, the implementation with RRB technique achieves to perform an inversion in $\mathbb{F}_{(2^4)^2}$ at $4T_{\mathrm{AND}} + 7T_{\mathrm{XOR}}$. The adoption of $\mathbb{F}_{(2^4)^2}$ also requires the acceleration of multiplications between the constant (8×8)–bit matrix and an 8–bit vector (an element in $\mathbb{F}_{(2^4)^2}$). Because this matrix is derived from a basis conversion matrix between the $\mathbb{F}_{2^8}$ and $\mathbb{F}_{2^2}$, in order to perform the above multiplication faster, an efficient basis conversion matrix must be prepared. Thus, this thesis also proposes MMMB technique which facilitates to select an efficient basis conversion matrix by a computation trick of multiplications

in $\mathbb{F}_{(2^4)^2}$ inside MixColumns and InvMixColumns of AES. Within the author's knowledge, the best conventional implementations perform an automorphism at $3T_{\mathrm{XOR}}$. On the other hand, the implementation with MMB technique achieves to perform an automorphism at $2T_{\mathrm{XOR}}$. By adopting RRB and MMMB, both of the encryption and decryption procedures of AES can be performed at $4T_{\mathrm{AND}} + 13T_{\mathrm{XOR}}$.

概要

本論文では，次世代の非対称鍵暗号方式が課す拡大体のパラメータ制約と，同方式が求める拡大体のパラメータに対する拡張性に，より柔軟に対応できる拡大体での乗算法として，ガウス周期正規基底（Gauss period Normal Basis: GNB）を適用した循環ベクトル乗算アルゴリズム（Cyclic Vector Multiplication Algorithm: CVMA）を提案し，その柔軟性を理論的に検証する．また，Advanced Encryption Standard (AES) をはじめとする対称鍵暗号方式の処理を従来実装よりも高速化するために，冗長表現基底（Redundantly Represented Basis: RRB）および複雑混合基底（More Miscellaneously Mixed Bases: MMMB）を提案する．

近年，次世代非対称鍵暗号方式として，ペアリングに基づく暗号方式が注目を集めている．この暗号方式の処理を高速化する手段として，拡大体での算術計算，とくに乗算の高速化は非常に有効である．一方で，この暗号方式では，拡大体 $\mathbb{F}_{p^m}$ のパラメータである標数 p および拡大次数 m に対して大きな制約を課す場合がある．ゆえに，ある程度の高速処理を可能とし，かつ p および m に対して柔軟に対応できる乗算法が必要とされる．この要求を満たす拡大体での乗算法として，CVMA が提案されている． この CVMA を採用するためには，拡大体を適当な正規基底で構成する必要があるが，その正規基底は GNB の一部である．これらの正規基底は，標数 p と拡大次数 m 以外に，p と m から条件付けされる正整数 h を必要とするが，例外を除けば，この h はそれぞれの p と m の組に対して無限に存在する．しかし，CVMA では h が増大すると致命的な速度低下を招くため，実質的に h の大きさには上限が存在する．ゆえに，p と m によって取りうる最小の h によっては，速度面において CVMA が適用外になる場面も生じる．そこで本稿では，従来の CVMA から，GNB を適用した CVMA へ改良を行う．この拡張によって，より小さな h を利用できるようになる．さらに，CVMA の柔軟性を論証するために，h ごとの GNB の存在確率を導出するための定理を提案する．この定理から，速度面を考慮した際に，p と m に対して改良した CVMA が柔軟性の高いものであることを理論的に示す.

一方で，対称鍵暗号の分野では，AES およびそれに類似する暗号方式のハードウェア実装報告が盛んに行われている．これらの実装の多くは，AES 内の暗号化・復号処理で最も低速な処理である SubBytes および InvSubBytes を高速化するために，本来の AES で採用されているような拡大体 $\mathbb{F}_{2^8}$ での逆元計算に替わり，その同型な逐次拡大体（合成体）である $\mathbb{F}_{((2^2)^2)^2}$ や $\mathbb{F}_{(2^4)^2}$ での逆元計算を採用している．本論文では，$\mathbb{F}_{((2^2)^2)^2}$ よりも逆元計算が高速な $\mathbb{F}_{(2^4)^2}$ に着目し，$\mathbb{F}_{(2^4)^2}$ での逆元計算を高速化できる RRB を提案する．その結果，$\mathbb{F}_{(2^4)^2}$ での逆元計算を，著者が知る限りの既存研究では $4T_{\mathrm{AND}} + 10T_{\mathrm{XOR}}$ で提供されるところを，$4T_{\mathrm{AND}} + 7T_{\mathrm{XOR}}$ で実現する．ただし，T_{AND} および T_{XOR} は AND および XOR ゲートの遅延時間を意味している．また，上記のように $\mathbb{F}_{(2^4)^2}$ での逆元計算を採用する場合，逆元計算前後で必要とされる (8×8)–bit の定行列と 8–bit ベクトルとの乗算を高速化することも重要である． この定行列は $\mathbb{F}_{2^8}$ から $\mathbb{F}_{(2^4)^2}$ への基底変換行列から導出されるため，より高速に上記の乗算を行うためには，効率の良い基底変換行列を準備する必要がある． そこで本論文では，AES 内の処理である MixColumns および InvMixColumns で実行される拡大体での乗算を工夫することよって，効率の良い $\mathbb{F}_{2^8}$ から $\mathbb{F}_{(2^4)^2}$ への基底変換行列とその逆変換行列の組を選択可能にする MMMB を提案する．その結果，行列とベクトル間の乗算を，著者が知る限りの既存研究では $3T_{\mathrm{XOR}}$ で提供されるところを，$2T_{\mathrm{XOR}}$ で実現する． この RRB と MMMB を適用することによって，AES の暗号化・復号処理をともに $4T_{\mathrm{AND}} + 13T_{\mathrm{XOR}}$ で実行できるようになる．

Notations

p	a characteristic of prime field (namely, a prime number)
$\mathbb{F}_p$	a prime field with a characteristic p
m	an extension degree (generally, a positive number larger than 1)
$\mathbb{F}_{p^m}$	an m–th extension field over $\mathbb{F}_p$
$\mathbb{F}_{p^m}^*$	the multiplicative group in $\mathbb{F}_{p^m}$
$E(\mathbb{F}_{p^m})$	an elliptic curve additive group over $\mathbb{F}_{p^m}$
M_m, S_m, A_m, D_m	the calculation costs of a multiplication, a squaring, an addition (or a subtraction), and a doubling in $\mathbb{F}_{p^m}$, respectively
$m \mid n$ and $m \nmid n$ (m, n: positive integers)	They mean that m does and does not divide n.
$\gcd(m, n)$ (m, n: positive integers)	the great common divisor for positive integers m and n
$\mathrm{Hw}(t)$	the Hamming weight of a positive integer t

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the modern society which utilizes Information and Communication Technology (ICT), more information security incidents have been reported as computer and network systems have become more complex. Above all, there is no end to the numbers of identity thefts and data falsifications although this thesis avoids to refer to the concrete incidents. The author concludes that one of the causes is the incompatibility between the CIA triad (Confidentiality, Integrity and Availability [1]) and the usability for their systems. In order to overcome the incompatibility, recently, as next generation asymmetric–key cryptosystems and their applications, *ID–based encryption* [2, 3] and functional encryption [4] have been contrived to provide the confidentiality, and *group signature* [5] has been proposed to provide the integrity.

On the other hand, in recent years, the damages caused by the above incidents become much larger since the amount of information becomes huge due to the increasing network traffic and external storage capacity. Thus, the high–performance cryptosytems are imperative to continuously achieve the practical uses of them. Especially, it is very important to accelerate symmetric–key cryptosytems such as *Triple Data Encryption Standard* (TDES) [6] and *Advanced Encryption Standard* (AES) [7] because the contributions of them are much important for the confidentiality.

Severals of both asymmetric–key cryptosystems and symmetric–key cryptosystems are based on arithmetic operations in *finite field* (*Galois field*) such as *prime field* and *extension field* as illustrated in **Fig**. 1.1. The purpose of this thesis is to propose high–performance arithmetic operations in extension field. Note that the author considers that the properties required for the arithmetic operations differ between the above introduced asymmetric–key and symmetric–key cryptosystems. This thesis contributes for each cryptosystems. In order to make a certain next generation asymmetric–key cryptosystems more scalable, this thesis proposes the *Cyclic Vector Multiplication Algorithm* (CVMA) for Gauss period Normal Basis (GNB). It is an efficient multiplication algorithm in extension field which is flexible for the restriction and scalability of the extension field parameters required by next generation asymmetric–key cryptosystems. On the other hand, in order to accelerate the computations of several symmetric–key cryptosystems such as Advanced Encryption Standard (AES), this thesis also proposes *Redundantly Represented Basis* (RRB) and *More Miscellaneously Mixed Bases* (MMMB).

1.1 Contribution of Asymmetric–key Cryptosystems

Recently, as next generation asymmetric–key cryptosystems, several pairing–based cryptosystems have been proposed such as ID–based encryption [2, 3], group signature [5] and functional

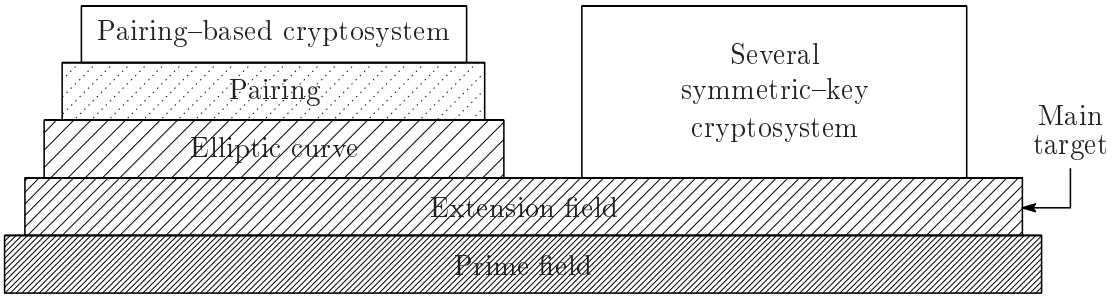


Figure 1.1: The research layers of cryptographies

encryption [4]. In order to accelerate these cryptosystems, not only pairing algorithms but also arithmetic operations, especially multiplications and Frobenius mapping in extension field, need to be improved [8].

As a widely–used extension field which has efficient arithmetic operations, Bailey et al. have proposed Optimal Extension Field (OEF) [9]. It is constructed by the polynomial basis whose modular polynomial is an irreducible binomial. This polynomial basis is sometimes called Optimal Polynomial Basis (OPB). In the case of OEF, some efficient multiplication algorithms can be applied such as schoolbook multiplication, Karatsuba multiplication [10, 11], and Toom–Cook multiplication [12, 13, 14]. As described at the beginning, OEF is widely–adopted because it can be appropriately chosen among several multiplication algorithms according to the situations. However, OEF $\mathbb{F}_{p^m}$ is available only when the following conditions are satisfied.

1) Every prime factor of m divides $p-1$.

2) $4 \mid p-1$ when $4 \mid m$.

As reported by Kato et al. [15], this restriction often causes a critical *mismatch* for pairs of characteristic p and extension degree m. Thus, it can be hardly said that OEF is highly–flexible.

On the other hand, as the other efficient extension field, Nogami et al. have introduced type–I All One Polynomial field (AOPF) [16]. It is constructed by a certain normal basis, namely type–I Optimal Normal Basis (ONB) [17], which is the set of zeros of an *irreducible* All One Polynomial (AOP). Thus, different from OEF constructed by polynomial basis, AOPF does not need any arithmetic operations for Frobenius mapping. As a multiplication algorithm applicable for AOPF, Nogami et al. proposed *Cyclic Vector Multiplication Algorithm* (CVMA) [16], which efficiently performs a multiplication because it is similar to Karatsuba multiplication. Compared to Karatsuba multiplication technique, CVMA is more algorithmically–systematic. Recently, Granger et al. [18] and later Baldwin et al. [19] have reported that CVMA technique is also available for an integer multiplication with *multi–precision* followed by a reduction modulo a special class of prime number, namely *Minimal Redundancy Cyclotomic Prime* (MRCP). A few years after the publication of the original CVMA technique, for type–II ONB [17], Nogami et al. also expanded it without the performance degradation [20]. Since then, in order to avoid name collisions, the extension fields constructed by type–I and type–II ONBs have been respectively called type–I and type–II AOPFs, and the corresponding CVMAs have been respectively prefixed with "type–I" and "type–II".

In the cases of utilizing type–I and type–II ONBs, certain restrictions are imposed such that $m+1$ and $2m+1$ respectively need to be prime numbers, for example. In order to overcome this inconvenience and keep the performance of CVMA technique, Kato et al. have introduced

type–I eXtend normal basis (type I–X NB) [21] and type–II eXtended NB (type II–X NB) [15] which are respectively based on type–I and tyep–II ONBs. Accordingly, CVMA technique was also expanded, namely type I–X and II–X CVMAs [21, 15]. Then, these CVMAs can support every pair of p and m such that $4p \nmid m(p-1)$. Type I–X and II–X NBs are special classes of Gauss period Normal Bases (GNBs) [22]. For type I–X and II–X NBs compared to GNBs, there exist some inefficient cases as introduced below. GNB is generally characterized with not only p and m but also a certain positive integer parameter h. Thus, this thesis especially calls it type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$, where h needs to satisfy the following conditions.

1) $r = hm + 1$ is a prime number not equal to p,

2) $\gcd(hm/e, m) = 1$, where e is the multiplicative order of p modulo r.

With the above notations, type I–X NB is, for example, classified to type–$\langle h, m\rangle$ GNB such that $e = hm$. For type–$\langle h, m\rangle$ GNB and its special classes, it is well–known that there exist such infinite positive integers h's, and one can be appropriately chosen; however, from the viewpoint of the computational cost of CVMAs, it is preferred to be small [21, 15]. Thus, the minimal one $h_{\min}$ should be adopted among such positive integers h's in order to prepare type–$\langle h_{\min}, m\rangle$ GNB in $\mathbb{F}_{p^m}$.

1.2 Contribution of Symmetric–key Cryptosystems

Since NIST published Advanced Encryption Standard (AES), namely a special class of Rijndael [7], many hardware implementations of AES algorithm have been reported [26, 27, 28, 29, 30, 31, 32]. This thesis also proposes approaches for more *efficient* hardware implementations, where the "*efficient*" is, in this thesis, meant with as primarily "*high–speed*", and secondly "*compact*".

In the encryption procedure of AES algorithm, the four steps such as SubBytes, ShiftRows, MixColumns and AddRoundKey [23] are iterated in sequence. On the other hand, in the decryption procedure of AES algorithm, 4 steps such as InvSubBytes, InvShiftRows, InvMixColumns, AddRoundKey [23] are iterated in sequence. For software implementations, SubBytes and InvSubBytes are often implemented with the lookup–table [7]. On the other hand, for hardware implementations, SubBytes and InvSubBytes are often implemented with some arithmetic operation circuits in octic binary extension field $\mathbb{F}_{2^8}$. In SubBytes and InvSubBytes, an inversion in $\mathbb{F}_{2^8}$ is carried out, and it plays an important role to prevent *linear cryptanalysis* [24]. Additionally, it is the most complex among the arithmetic operations. On the other hand, in the case of hardware implementations, not only SubBytes and InvSubBytes but also MixColumns and InvMixColumns should be efficient. In MixColumns and InvMixColumns, some multiplications in $\mathbb{F}_{2^8}$ are carried out. Thus, this thesis first considers to implement efficient arithmetic operation circuits in $\mathbb{F}_{2^8}$ by using only some logic gates such as AND, XOR, and XNOR gates.

In the case of the original AES algorithm [7], an element in $\mathbb{F}_{2^8}$ is represented by the polynomial basis, whose modular polynomial is the octic irreducible polynomial $t^8+t^4+t^3+t+1$ over $\mathbb{F}_2$. Therefore, originally, SubBytes and InvSubBytes implementations require inversion circuits in the $\mathbb{F}_{2^8}$. However, by adopting inversion circuits in towering fields (composite fields [25]) isomorphic to the $\mathbb{F}_{2^8}$, some researchers have been provided faster and more compact SubBytes and InvSubBytes circuits. At the beginning, Rudra et al. have shown such implementation with a certain $\mathbb{F}_{(2^4)^2}$ as the isomorphic towering field [26]. On the other hand, Satoh and Morioka et al. have shown that with a certain $\mathbb{F}_{((2^2)^2)^2}$ [27, 28]. After those, some implementations with the other $\mathbb{F}_{(2^4)^2}$ and $\mathbb{F}_{((2^2)^2)^2}$ have been reported [29, 30, 31, 32]. Within the author's knowledge, the implementations with $\mathbb{F}_{(2^4)^2}$ [26, 32] can provide faster inversion circuits than those with

$\mathbb{F}_{((2^2)^2)^2}$ [27, 28, 29, 30, 31]. Thus, this thesis focuses on $\mathbb{F}_{(2^4)^2}$, and proposes *Redundantly Represented Basis* (RRB) which can provide faster inversion circuits in $\mathbb{F}_{(2^4)^2}$ than the bases adopted by [26, 32]. Then, this thesis also considers multiplication circuits in the $\mathbb{F}_{(2^4)^2}$ with RRB. By adopting RRB, an inversion in $\mathbb{F}_{(2^4)^2}$ can be carried out in $4T_{\mathrm{AND}} + 7T_{\mathrm{XOR}}$, where T_{AND} and T_{XOR} respectively denote the critical path delays of AND and XOR gates.

In the case that arithmetic operations in towering field isomorphic to the $\mathbb{F}_{2^8}$ are adopted for the encryption and decryption procedures of AES algorithm, not only arithmetic operations in an isomorphic towering field but also basis conversion from the $\mathbb{F}_{2^8}$ to the isomorphic towering field should be efficient. When many kinds of basis conversion matrices can not be prepared, it is quite difficult to select some efficient conversion matrices. In order to prepare more kinds of basis conversion matrices, Nogami et al. have proposed *Mixed Bases* (MB) technique [31]; however, when using RRB, MB is not enough to provide efficient matrices. Thus, this thesis proposes *More Miscellaneously Mixed Bases* (MMMB), and then shows how to find efficient conversion matrices.

1.3 Outline

This thesis is organized as follows:

Chap. 2 briefly reviews group and finite field theories.

Chap. 3 focuses on two remaining problems: 1) The minimal $h_{\min}$ of type I–X and II–X NBs sometimes become larger than that of type–$\langle h, m\rangle$ GNB, and then this inconvenience causes some inefficient implementations, for example, as shown in [33]. 2) CVMA technique has not been expanded for general GNBs yet. First, this chapter expands CVMA technique for type–$\langle h, m\rangle$ GNBs. As the result, this expansion will improve some inefficient situations because it is possible that $h_{\min}$ becomes smaller by this expansion. After that, in order to *theoretically* obtain the tendency of the computational complexity of CVMA with respect to extension degrees, this chapter proposes an important theorem such that the existence probability of type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$ and the expected value of $h_{\min}$ can be explicitly obtained. Then, this chapter demonstrates the efficiency difference for $h_{\min}$ between type I–X and II–X CVMAs and the CVMA expanded for type–$\langle h, m\rangle$ GNBs.

Chap. 4 holds the following proposals: 1) to make arithmetic operations in $\mathbb{F}_{(2^4)^2}$ more efficient, and 2) to find more efficient basis conversion matrices. As described above, the former proposal is achieved by RRB, and the latter proposal is achieved by MMMB. By utilizing RRB and MMMB, this chapter theoretically shows that the encryption and decryption circuits of AES can be provided by the critical path delay $4T_{\mathrm{AND}} + 13T_{\mathrm{XOR}}$.

Chap. 5 concludes this thesis.

Chapter 2

Fundamentals

This chapter briefly reviews group and field theories.

2.1 Group

Group is an algebraic system defined as follows.

Definition 1 (Group) *A group* $\langle \mathbb{G}, \circ \rangle$ *is a nonempty set with a binary operation* $\circ$ *that satisfies the following group axioms:*

- **G1 : (Closure)** *For* $^\forall a, ^\forall b \in \mathbb{G}$*, the result of* $a \circ b$ *is also in* $\mathbb{G}$*.*
- **G2 : (Associativity)** $(a \circ b) \circ c = a \circ (b \circ c), \quad a, b, c \in \mathbb{G}$*.*
- **G3 : (Unity)** *For* $^\forall a \in \mathbb{G}$*, there exists an element* $e \in \mathbb{G}$ *such that* $a \circ e = e \circ a = a$*, where* e *is called unity (unit element).*
- **G4 : (Inverse Element)** *For* $^\forall a \in \mathbb{G}$*, there exists an element* $x \in \mathbb{G}$ *such that* $a \circ x = x \circ a = e$*, where* x *is called inverse element of* a*.*

Definition 2 (Commutative Group)

- **AG5 : (Commutativity)** *A group* $\mathbb{G}$ *is said to be commutative (or abelian), if* $a \circ b = b \circ a$ *for* $^\forall a, b \in \mathbb{G}$*.*

For example, the algebraic system $\langle \mathbb{Z}, + \rangle$ is an infinite commutative group, where $\mathbb{Z}$ is the integer set and $+$ means the ordinary addition for integers. For a finite group, its order is defined as follows.

Definition 3 (Order of Group) *The order* $|\mathbb{G}|$ *is the number of elements in finite group* $\mathbb{G}$*.*

Let us consider a example of finite group. An algebraic system $\langle \mathbb{Z}_n = \{0, 1, 2, 3, \ldots, n-1\}, + \rangle$ is not a group because it does not satisfy the group axioms. Therefore, in order to construct a group from $\mathbb{S}$, it is necessary to modify the addition. We will define a new sum as

$$a + b \equiv c \pmod{n}, \quad a, b \in \mathbb{Z}_n, \tag{2.1}$$

where the notation "$c \pmod{n}$" means that c is assigned to a remainder on division by n when $a + b = c \notin \mathbb{Z}_n$. Therefore, c certainly belongs to $\mathbb{Z}_n$ and then $\langle \mathbb{Z}_n, + \rangle$ forms a group.

There is a convenient way of presenting a finite group. A table displaying the group operation is referred to as a Cayley table. For example, the group $\mathbb{Z}_4$ is presented as follows.

Example 1 *The Cayley table for the group* $\mathbb{Z}_4$ *is:*

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

In what follows, we will use the notation of ordinary addition such that $a + a = 2a$ and $a + a + a = 3a$ (in multiplicative notation, these are denoted by a^2,a^3).

Cyclic Group

A group $\mathbb{G}$ is said to be *cyclic* if there is an element $g \in \mathbb{G}$ such that for any $a \in \mathbb{G}$ there is some integer j with $a = g^j$. Such an element g is called a generator of the cyclic group.

From the definition, we can see that any elements in cyclic group are generated with iterative operations of generator g. **Fig**. 2.1 shows it schematically.

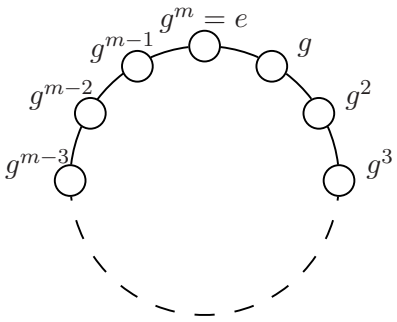


Figure 2.1: Cyclic group

In general, for an element $a \in \mathbb{G}$, the least positive integer m such that $a^m = e$ is called order of a, where e is the unity in $\mathbb{G}$.

Isomorphism

Let us consider a mapping ψ from a set $\mathbf{A}$ to a set $\mathbf{B}$ as

$$\psi : a \in \mathbf{A} \longmapsto b \in \mathbf{B}, \quad b = \psi(a). \tag{2.2}$$

When it satisfies each of the following, ψ is called a *surjection*, an *injection* or a *bijection*.

Surjection : For $^\forall b \in \mathbf{B}$, when there exists $a \in \mathbf{A}$ such that $b = \psi(a)$.

Injection : For $a_1 \neq a_2 \in \mathbf{A}$, when $\psi(a_1) \neq \psi(a_2) \in \mathbf{B}$.

Bijection : ψ is surjection and injection. In other words, it is one–to–one mapping.

When a mapping $\psi : \langle \mathbf{A}, \alpha \rangle \longmapsto \langle \mathbf{B}, \beta \rangle$ satisfies the following relation, it is called *homomorphism*.

$$\psi(a_1 \, \alpha \, a_2) = \psi(a_1) \, \beta \, \psi(a_2) = b_1 \, \beta \, b_2, \quad (a_1, a_2 \in \mathbf{A}, \quad b_1 = \psi(a_1), \; b_2 = \psi(a_2) \in \mathbf{B}). \tag{2.3}$$

If ψ is both bijection, it is called *isomorphism*, then $\langle \mathbf{A}, \alpha \rangle$ and $\langle \mathbf{B}, \beta \rangle$ are said to be isomorphic. Additionally, a homomorphism from a group to itself is called an *endomorphism*, and if it is both bijection then it is called *automorphism*.

Kernel

For a homomorphism $\psi : \langle \mathbf{A}, \alpha \rangle \longmapsto \langle \mathbf{B}, \beta \rangle$, the following $\mathrm{Ker}(\psi)$ is called a *kernel* of ψ.

$$\mathrm{Ker}(\psi) = \{a \in \mathbf{A} \mid \psi(a) = e_{\mathbf{B}}\}, \quad (e_{\mathbf{B}} : \text{the unity in } \langle \mathbf{B}, \beta \rangle) \tag{2.4}$$

Cartesian Product Set

For two sets $\mathbf{A}$ and $\mathbf{B}$, the following set $\mathbf{A} \times \mathbf{B}$ is called *Cartesian product set.*

$$\mathbf{A} \times \mathbf{B} = \{(a, b) \mid a \in \mathbf{A},\ b \in \mathbf{B}\}. \tag{2.5}$$

Note that $(a, b) = (a', b')$ only when $a = a',\ \ b = b'$.

2.2 Field

Field is an algebraic system defined as follows.

Definition 4 (Field) *A field $\langle \mathbb{F}, +, \cdot \rangle$ has two binary operations denoted by $+$ and $\cdot$, such that:*

F1 : (Additive Group) *$\mathbb{F}$ is a commutative group with respect to $+$.*

F2 : (Multiplicative Group) *$\mathbb{F}^*$ is a group with respect to $\cdot$, where $\mathbb{F}^*$ is the set that consists of every element distinct from the unity (zero element) with respect to $+$.*

F3 : (Distributive law) *For all $a, b, c \in \mathbb{F}$ we have $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(b + c) \cdot a = b \cdot a + c \cdot a$.*

In general, the elements 0 and 1 represent the unity regarding to the operation $+$ and regarding to the operation $\cdot$, respectively.

Definition 5 (Order of Field) *The order is the number of elements in $\mathbb{F}$. If the order of $\mathbb{F}$ is finite, $\mathbb{F}$ is called finite field.*

Definition 6 (Characteristic of Field) *The least positive number n such that $n \cdot a = 0$ for every $a \in \mathbb{F}$ is called characteristic.*

This paper treats only finite fields. Finite fields have the following property, which is used often in cryptographic area.

Theorem 1 *For every finite field $\mathbb{F}$, the multiplicative group $\mathbb{F}^*$ is cyclic.*

For example, ElGamal encryption [34] can be defined over multiplicative group of $\mathbb{F}$. Its security depends on the difficulty of a certain problem in $\mathbb{F}$ related to computing *discrete logarithms.*

2.2.1 Prime Field

A subset $\mathbb{K}$ of a field $\mathbb{F}$ that is itself a field under the operations of $\mathbb{F}$ will be called a *subfield* of $\mathbb{F}$. In this case, $\mathbb{F}$ is called an *extension (field)* of $\mathbb{K}$. If $\mathbb{K} \neq \mathbb{F}$, we say that $\mathbb{K}$ is a *proper subfield* of $\mathbb{F}$. Then, prime field is defined as follows.

Definition 7 (Prime Field) *A field containing no proper subfield is called prime field.*

Moreover, the following theorem is given about finite field.

Theorem 2 *Every finite field has a prime field as a subfield.*

Therefore, finite fields are classified into two types, which are prime field and its extension field. Prime field $\mathbb{F}_p$ has a prime number p as the order and characteristic.

Arithmetic Operations in Prime Field

In the same way as Eq. (2.1), we can define fundamental operations of $\mathbb{F}_p = \{0, 1, 2 \cdots, p-1\}$ by using the remainder of an integer as follows.

$$a + b \equiv c \pmod{p}, \qquad a - b = a + (-b) \equiv c \pmod{p}, \tag{2.6a}$$

$$a \cdot b \equiv c \pmod{p}, \qquad a \,/\, b = a \cdot b^{-1} \equiv c \pmod{p}, \qquad a, b \in \mathbb{F}_p. \tag{2.6b}$$

We can obtain b^{-1} easily by using *Fermat's Little Theorem* as follows.

Theorem 3 (Fermat's Little Theorem) *For a non–zero element $a \in \mathbb{F}_p$, we have*

$$a^{p-1} \equiv 1 \pmod{p}. \tag{2.7}$$

On the other hand, using Lehmer's technique and the technique based on binary greatest common divisor (BGCD) , the inversion can be also carried out efficiently.

2.2.2 Extension Field

The order of extension field $\mathbb{F}_{p^m}$ is p^m. Arithmetic operations in $\mathbb{F}_{p^m}$ are realized by using polynomials. Every element in $\mathbb{F}_{p^m}$ is expressed as a polynomial that have m elements in $\mathbb{F}_p$ as coefficients. Then, arithmetic operations in $\mathbb{F}_{p^m}$ are carried out with ordinary addition, subtraction and multiplication for polynomial, and modular polynomial reduction by using a certain irreducible polynomial.

Definition 8 (Irreducibility) *A polynomial $f(x)$ is said to be irreducible over $\mathbb{F}_p$ if there does not exit, except $f(x)$ itself, polynomials of degree more than or equal to 1 those divide $f(x)$.*

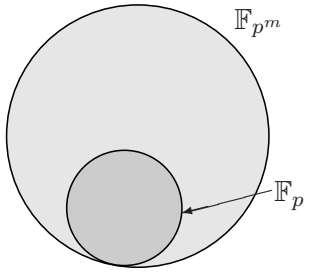


Figure 2.2: Sketch of an m–th extension field $\mathbb{F}_{p^m}$

Addition, Subtraction and Multiplication in Extension Field

As described above, every element in $\mathbb{F}_{p^m}$ is expressed as a polynomial. We denote elements A and B in $\mathbb{F}_{p^m}$ by

$$\begin{aligned} A &= a_0 + a_1\omega + a_2\omega^2 + \cdots + a_{m-1}\omega^{m-1}, \\ B &= b_0 + b_1\omega + b_2\omega^2 + \cdots + b_{m-1}\omega^{m-1}, \end{aligned} \tag{2.8}$$

where ω is a root of a monic irreducible polynomial of degree m

$$f(x) = \sum_{i=0}^{m} f_i x^i = f_0 + f_1 x + f_2 x^2 + \cdots f_{m-1}x^{m-1} + x^m \text{ over } \mathbb{F}_p, \tag{2.9}$$

in short, $f(\omega) = 0$. In this case, ω is a proper element[1] in $\mathbb{F}_{p^m}$ and then $\mathbb{F}_p$ is extended as a m–dimensional vector space over $\mathbb{F}_p$ by a following basis

$$\{1, \omega, \omega^2, \cdots, \omega^{m-1}\}. \tag{2.10}$$

Therefore, any elements in $\mathbb{F}_{p^m}$ is expressed as a linear combination of $\{1, \omega, \omega^2, \cdots, \omega^{m-1}\}$. Then, $\langle \mathbb{F}_{p^m}, + \rangle$ forms a commutative group under addition (and subtraction) as shown in **Fig**. 2.3 (and **Fig**. 2.4) because the basis $\{1, \omega, \omega^2, \cdots, \omega^{m-1}\}$ is linearly independent when $f(x)$ is irreducible.

$$\begin{array}{rl} & A = \quad a_0 + \quad a_1\omega + \quad a_2\omega^2 + \cdots + \quad a_{m-1}\omega^{m-1} \\ +) & B = \quad b_0 + \quad b_1\omega + \quad b_2\omega^2 + \cdots + \quad b_{m-1}\omega^{m-1} \\ \hline & A + B = (a_0{+}b_0) + (a_1{+}b_1)\omega + (a_2 + b_2)\omega^2 + \cdots + (a_{m-1} + b_{m-1})\omega^{m-1} \end{array}$$

Figure 2.3: Addition in extension field

$$\begin{array}{rl} & A = \quad a_0 + \quad a_1\omega + \quad a_2\omega^2 + \cdots + \quad a_{m-1}\omega^{m-1} \\ -) & B = \quad b_0 + \quad b_1\omega + \quad b_2\omega^2 + \cdots + \quad b_{m-1}\omega^{m-1} \\ \hline & A - B = (a_0{-}b_0) + (a_1{-}b_1)\omega + (a_2 - b_2)\omega^2 + \cdots + (a_{m-1} - b_{m-1})\omega^{m-1} \end{array}$$

Figure 2.4: Subtraction in extension field

On the other hand, $\mathbb{F}^*_{p^m}$ is not closed under ordinary multiplication for polynomials as shown in **Fig**. 2.5.

$$\begin{array}{rl} & A = a_0 + a_1\omega + a_2\omega^2 + \cdots + a_{m-1}\omega^{m-1} \\ \cdot\ ) & B = b_0 + b_1\omega + b_2\omega^2 + \cdots + b_{m-1}\omega^{m-1} \\ \hline & A \cdot B = (a_0b_0) + (a_0b_1 + a_1b_0)\omega + \cdots\cdots\cdots\cdots + (a_{m-1}b_{m-1})\omega^{2m-2} \end{array}$$

Figure 2.5: Multiplication in extension field with school book method

In order to make $\mathbb{F}^*_{p^m}$ closed under multiplication, we need modular polynomial reduction with a root ω of $f(x)$. ω holds the following relation,

$$\omega^m = -f_0 - f_1\omega - f_2\omega^2 - f_{m-1}\omega^{m-1}. \tag{2.11}$$

By applying Eq. (2.11) to power of ω repeatedly, we can reduce the degree from $2m-2$ to $m-1$. In other words, we can express a product of A and B with linear combination of the basis $\{1, \omega, \omega^2, \cdots, \omega^{m-1}\}$ as follows,

$$A \cdot B = \quad c_0 + c_1\omega + c_2\omega^2 + \cdots + c_{m-1}\omega^{m-1} \mod \underline{f(\omega)}\,. \tag{2.12}$$

Then, a multiplication in $\mathbb{F}_{p^m}$ is carried out with an ordinary multiplication for polynomial and modular polynomial reduction as described above.

Theorem 4 *Let $f(x)$ be an irreducible polynomial of degree m over $\mathbb{F}_p$. Then there exists an extension field $\mathbb{F}_{p^m}$ over $\mathbb{F}_p$ with a root of $f(x)$ as a basis generator.*

[1] In this paper, we call an element that belongs to $\mathbb{F}_{p^m}$ but not to its proper subfield *a proper element* in $\mathbb{F}_{p^m}$.

Basis in Extension Field

There are many bases to express an element in $\mathbb{F}_{p^m}$, and each basis has a different effect on operations in $\mathbb{F}_{p^m}$. For example, the basis $\{1, \omega, \omega^2, \cdots, \omega^{m-1}\}$ as previously described is called *polynomial basis* and efficient for multiplication. On the other hand, when the following conjugates of a generator ω are linearly independent,

$$\{\omega, \omega^p, \omega^{p^2}, \cdots, \omega^{p^{m-1}}\}. \tag{2.13}$$

their set is called *normal basis*, and efficient for *Frobenius mapping* :

$$A \to A^p, \quad A \in \mathbb{F}_{p^m}. \tag{2.14}$$

In general, using a basis $\{\omega_0, \cdots, \omega_{m-1}\}$, an arbitrary element A in $\mathbb{F}_{p^m}$ is represented as

$$A = a_0\omega_0 + a_1\omega_1 + \cdots + a_{m-1}\omega_{m-1}. \tag{2.15}$$

Every basis consists of m linearly independent elements in $\mathbb{F}_{p^m}$.

Inversion in Extension Field

In the same way as prime field $\mathbb{F}_p$, an element in extension field $\mathbb{F}_{p^m}$ has the following property.

Theorem 5 *For a non–zero element A in $\mathbb{F}_{p^m}$, we have equality as*

$$A^{p^m-1} = (A \cdot A^p \cdot A^{p^2} \cdot \cdots \cdot A^{p^{m-1}})^{p-1} = \Big(\prod_{i=0}^{m-1} A^{p^i}\Big)^{p-1} = 1. \tag{2.16}$$

$\prod_{i=0}^{m-1} A^{p^i}$ is the product of conjugates of A with respect to $\mathbb{F}_p$. This production is called *norm*. For this norm, we can easily obtain the following theorem by Eq. (2.16).

Theorem 6 *The norm of A with respect to $\mathbb{F}_p$ becomes a non–zero element in $\mathbb{F}_p$.*

As an inversion algorithm in extension field $\mathbb{F}_{p^m}$ efficiently by using this property, Itoh–Tsujii inversion algorithm (ITA) [35] shown in **Alg**. 1 have been proposed.

Algorithm 1: ITA in $\mathbb{F}^*_{p^m}$

Input: $X \in \mathbb{F}^*_{p^m}, \quad s = m - 1.$

Output: $Z = X^{-1} = \left(X^p \cdots X^{p^{m-1}}\right) / \left(XX^p \cdots X^{p^{m-1}}\right).$

1 $Y \leftarrow X^p, \quad j = 1.$
2 **for** $\lfloor \log_2(s) \rfloor \geq i \geq 1$ **do** $Z \leftarrow Z \cdot Z^{p^j}, \quad j \leftarrow 2j.$
3 **if** $s[i] = 1$ **then** $Z \leftarrow Z \cdot Y^p$, $j \leftarrow j + 1.$
4 $x \leftarrow Z \cdot X.$
5 $Z \leftarrow Z \cdot x^{-1}.$

$x\,(= \prod_{i=0}^{m-1} X^{p^i})$ in **Step** 4 of **Alg**. 1 becomes a non–zero element that belongs to $\mathbb{F}_p$ because it is the norm of X with respect to $\mathbb{F}_p$. Therefore, in order to obtain x in **Step** 4 of **Fig**. 1, we just have to calculate one of the vector coefficients with respect to $\mathbb{F}_p$ of $Z \cdot X \in \mathbb{F}_{p^m}$ when $\mathbb{F}_p$ is constructed by a polynomial basis or type–$\langle k, m \rangle$ Gauss period normal basis (GNB) described below. Additionally, the calculation amount of x^{-1} is I_1 because $x \in \mathbb{F}_p$.

Chapter 3

Multiplications Flexible for Scalable Asymmetric–key Cryptosystems

3.1 Fundamentals of Gauss Period Normal Basis (GNB)

In this paper, the set which contains a normal basis, the extension field constructed by the normal basis, and a certain efficient multiplication algorithm in the extension field is considered as *a series*, and this section briefly reviews several kinds of series. The contents of the series introduced in this section are described in **Table** 3.1.

Table 3.1: The notations of normal bases, extension fields, and efficient multiplication algorithms

Series	Basis	Extension field	Multiplication algorithm
type–I series	type–I ONB [17] (type–$\langle 1, m\rangle$ GNB)	type–I AOPF [16] (type–$\langle 1, m\rangle$ AOPF)	type–I CVMA [16] (type–$\langle 1, m\rangle$ CVMA)
type–II series	type–II ONB [17] (type–$\langle 2, m\rangle$ GNB)	type–II AOPF [20] (type–$\langle 2, m\rangle$ AOPF)	type–II CVMA [20] (type–$\langle 2, m\rangle$ CVMA)
type I–X series	type I–X NB [21]	type I–X AOPF [21]	type I–X CVMA [21]
type II–X series	type II–X NB [15]	type II–X AOPF [15]	type II–X CVMA [15]
type–$\langle h, m\rangle$ series	type–$\langle h, m\rangle$ GNB [22]	type–$\langle h, m\rangle$ AOPF (called in this thesis)	**type–$\langle h, m\rangle$ CVMA** (proposed in this thesis)

3.1.1 Type–I Series

Mullin et al. have proposed type–I *Optimal Normal Basis* (ONB) [17] as a normal basis efficient for extension field multiplication. Type–I ONB exists when the following condition is satisfied.

Condition 1 (the existence of type–I ONB)

1) $r = m + 1$ *is a prime number not equal to* p*.*

2) The order of p *in* $\mathbb{F}_r$ *is* m*.*

Then, the following multiplicative group is obtained.

$$\langle \{ \langle\!\langle p^i \rangle\!\rangle : 0 \le i < m \}, \cdot \rangle = \mathbb{F}_r^*, \tag{3.1}$$

where $\langle\langle t \rangle\rangle$ denotes "$t \pmod r$" for an integer t and the prime number $r = m + 1$. Here, let β be a primitive r–th root of unity in $\mathbb{F}_{p^m}$. In other words, β is a zero of the following *all one polynomial* (AOP) over $\mathbb{F}_p$.

$$f(t) = \frac{t^r - 1}{t - 1} = \sum_{i=0}^{r-1} t^i = \Phi_r(t), \tag{3.2}$$

where Φ_s denotes the s–th cyclotomic polynomial for a positive integer s. Type–I ONB is defined with the above β as follows.

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\} = \{\gamma, \gamma^2, \cdots, \gamma^m\}, \quad \gamma = \beta \in \mathbb{F}_{p^m}. \tag{3.3}$$

Actually, as shown in Eq. (3.3), type–I ONB forms not only normal basis but also pseudo polynomial basis. Since this ONB is prepared by a zero β of the AOP given by Eq. (3.2), Nogami et al. have especially called the extension field constructed by this ONB type–I *All One Polynomial Field* (AOPF) [16].

For this AOPF, Karatsuba multiplication [10, 11] can be applied since type–I ONB is also pseudo polynomial basis as described above. On the other hand, Nogami et al. have proposed type–I *Cyclic Vector Multiplication Algorithm* (CVMA) [16] as the other efficient multiplication algorithm in type–I AOPF. This algorithm is described in **Alg**. 2.

Algorithm 2: Type–I CVMA [16]

Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}, \quad x_i, y_i \in \mathbb{F}_p$.

Output: $Z = X \cdot Y = \sum_{i=0}^{m-1} z_i \gamma^{p^i}, \quad z_i \in \mathbb{F}_p$.

Preparation steps:

1 $\epsilon[0] \leftarrow m$.
2 **for** $i = 0$ **to** $m - 1$ **do** $\epsilon\big[\langle\langle p^i \rangle\rangle\big] \leftarrow i$.
3 **for** $i = 0$ **to** $m - 2$ **do**
4 $\quad$ **for** $j = i + 1$ **to** $m - 1$ **do** $\eta[i, j] \leftarrow \epsilon\big[\langle\langle p^i + p^j \rangle\rangle\big]$.

Evaluation steps:

5 **for** $l = 0$ **to** $m - 1$ **do** $v_l \leftarrow x_l y_l$.
6 $v_m \leftarrow 0$.
7 **for** $i = 0$ **to** $m - 2$ **do**
8 $\quad$ **for** $j = i + 1$ **to** $m - 1$ **do** $v_{\eta[i,j]} \leftarrow v_{\eta[i,j]} + (x_i - x_j)(y_i - y_j)$.
9 **for** $l = 0$ **to** $m - 1$ **do** $z_l \leftarrow v_m - v_l$.

3.1.2 Type–II Series

Mullin et al. have also proposed type–II ONB [17] as another normal basis efficient for extension field multiplication. Type–II ONB exists when **Cond**. 2.1 and either 2.2a or 2.2b are satisfied.

Condition 2 (the existence of type II ONB)

1) $r = 2m + 1$ is a prime number not equal to p.

2a) The order of p in $\mathbb{F}_r$ is $2m$.

2b) The order of p in $\mathbb{F}_r$ is m, and m is odd.

Then, the following multiplicative group is obtained.

$$\left\langle \left\{ \left\langle\!\left\langle \pm p^i \right\rangle\!\right\rangle : 0 \le i < m \right\}, \cdot \right\rangle = \mathbb{F}_r^*, \tag{3.4}$$

where $\langle\!\langle t \rangle\!\rangle$ denotes "$t \pmod r$" for an integer t and the prime number $r = 2m + 1$. Here, let β be a primitive r–th root of unity in $\mathbb{F}_{p^e}^*$, where e is the order of p in $\mathbb{F}_r^*$. In other words, β is a certain zero of the AOP in Eq. (3.2). Type–II ONB is defined with the above β as follows.

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}, \quad \gamma = \beta + \beta^{-1} \in \mathbb{F}_{p^m}. \tag{3.5}$$

Type–II ONB is not polynomial basis differently from type–I ONB.

In the same manner as type–I series, Nogami et al. have especially called the extension field constructed by this ONB type–II AOPF, and have also proposed type–II CVMA [20]. This algorithm is described in **Alg**. 3.

Algorithm 3: Type–II CVMA [20]

Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}, \quad x_i, y_i \in \mathbb{F}_p.$

Output: $Z = X \cdot Y = \sum_{i=0}^{m-1} z_i \gamma^{p^i}, \quad z_i \in \mathbb{F}_p.$

Preparation steps:

1 **for** $i = 0$ **to** $m - 1$ **do** $\epsilon\left[\left\langle\!\left\langle p^i \right\rangle\!\right\rangle\right] \leftarrow i, \quad \epsilon\left[\left\langle\!\left\langle -p^i \right\rangle\!\right\rangle\right] \leftarrow i.$

2 **for** $i = 0$ **to** $m - 2$ **do**

3 $\quad$ **for** $j = i + 1$ **to** $m - 1$ **do** $\eta[i, j, 0] \leftarrow \epsilon\left[\left\langle\!\left\langle p^i + p^j \right\rangle\!\right\rangle\right], \quad \eta[i, j, 1] \leftarrow \epsilon\left[\left\langle\!\left\langle p^i - p^j \right\rangle\!\right\rangle\right].$

Evaluation steps:

4 **for** $l = 0$ **to** $m - 1$ **do** $v_l \leftarrow x_l y_l.$

5 **for** $i = 0$ **to** $m - 2$ **do**

6 $\quad$ **for** $j = i + 1$ **to** $m - 1$ **do**

7 $\quad\quad u \leftarrow (x_i - x_j)(y_i - y_j), \quad v_{\eta[i,j,0]} \leftarrow v_{\eta[i,j,0]} + u, \quad v_{\eta[i,j,1]} \leftarrow v_{\eta[i,j,1]} + u.$

8 **for** $l = 0$ **to** $m - 1$ **do** $z_l \leftarrow -v_l.$

3.1.3 Type–I eXtended (Type I–X) Series

Type–I and type–II ONBs are very efficient normal bases for extension field multiplication; however, these ONBs do not exist for an arbitrary pair of characteristic p and extension degree m. In order to overcome this inconvenience, Kato et al. have proposed type I–X normal basis (NB) [21]. It is prepared with a positive integer h which satisfies the following condition.

Condition 3 (the h of type I–X NB)

1) $r = hm + 1$ is a prime number not equal to p.

2) The order of p in $\mathbb{F}_r$ is hm.

Then, the following multiplicative group is obtained.

$$\left\langle \left\{ \left\langle\!\left\langle p^{i+km} \right\rangle\!\right\rangle : 0 \le i < m, 0 \le k < h \right\}, \cdot \right\rangle = \mathbb{F}_r^*, \tag{3.6}$$

where $\langle\!\langle t \rangle\!\rangle$ denotes "$t \pmod r$" for an integer t and the prime number $r = hm + 1$. Here, let β be a primitive m–th root of unity in $\mathbb{F}_{p^{hm}}^*$. In other words, β is a zero of the AOP in Eq. (3.2).

Type I–X NB is defined with the above h and β as follows.

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}, \quad \gamma = \sum_{k=0}^{h-1} \beta^{p^{km}} \in \mathbb{F}_{p^m}. \tag{3.7}$$

Actually, the set $\{\beta, \beta^p, \ldots, \beta^{p^{hm-1}}\}$ on which type I–X NB is based is type–I ONB in $\mathbb{F}_{p^{hm}}$. Thus, as the name suggests, *type–I eXtended* NB is obtained by extending type–I ONB. Because this NB exists whenever $8p \nmid m(p-1)$ [21], it is available in $\mathbb{F}_{p^m}$ for every pair of characteristic p and extension degree m when $p > m$.

In the same manner as type–I and type–II series, Kato et al. have especially called the extension field constructed by this NB type I–X AOPF, and have also proposed type I–X CVMA [21]. This algorithm is described in **Alg**. 4.

Algorithm 4: Type I–X CVMA [21]

Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}, \quad x_i, y_i \in \mathbb{F}_p$.

Output: $Z = X \cdot Y = \sum_{i=0}^{m-1} z_i \gamma^{p^i}, \quad z_i \in \mathbb{F}_p$.

Preparation steps:

Prepare a positive integer h which satisfies **Cond**. 3.
$\epsilon[0] \leftarrow m$.
for $i = 0$ **to** $m-1$ **do**
  for $k = 0$ **to** $h-1$ **do** $\epsilon\left[\left\langle\!\left\langle p^{i+km} \right\rangle\!\right\rangle\right] \leftarrow i$.
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    for $k = 0$ **to** $h-1$ **do** $\eta[i,j,k] \leftarrow \epsilon\left[\left\langle\!\left\langle p^i + p^{j+km} \right\rangle\!\right\rangle\right]$.

Evaluation steps:

for $l = 0$ **to** $m-1$ **do** $v_l \leftarrow x_l y_l$.
$v_m \leftarrow 0$.
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    $u \leftarrow (x_i - x_j)(y_i - y_j)$.
    for $k = 0$ **to** $h-1$ **do** $v_{\eta[i,j,k]} \leftarrow v_{\eta[i,j,k]} + u$.
if *h is odd* **then**
  $w \leftarrow h v_m$.
  for $l = 0$ **to** $m-1$ **do** $z_l \leftarrow w - v_l$.
else for $l = 0$ **to** $m-1$ **do** $z_l \leftarrow -v_l$.

3.1.4 Type–II eXtended (Type II–X) Series

In the same way as type I–X NB, Kato et al. have also proposed type II–X NB [15] which is obtained by extending type–II ONB. It is prepared with an even positive integer h which satisfies **Cond**. 4.1 and either 4.2a or 4.2b, where h' denotes $h/2$.

Condition 4 (the $h\,(= 2h')$ of type II–X NB)

1) $r = hm + 1$ is a prime number not equal to p.

2a) The order of p in $\mathbb{F}_r$ is hm.

2b) The order of p in $\mathbb{F}_r$ is $h'm$, and m is odd.

Then, the following multiplicative group is obtained.

$$\left\langle \left\{ \left\langle\!\left\langle \pm p^{i+km} \right\rangle\!\right\rangle : 0 \le i < m, 0 \le k < h' \right\}, \cdot \right\rangle = \mathbb{F}_r^*, \tag{3.8}$$

where $\langle\!\langle t \rangle\!\rangle$ denotes "$t \pmod r$" for an integer t and a prime number $r = hm + 1 = 2h'm + 1$. Here, let β be a primitive r–th root of unity in $\mathbb{F}_{p^e}^*$, where e is the order of p in $\mathbb{F}_r^*$. In other words, β is a certain zero of the AOP in Eq. (3.2). Type II–X NB is defined with the above $h'(= h/2)$ and β as follows.

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}, \quad \gamma = \sum_{k=0}^{h'-1} (\beta + \beta^{-1})^{p^{km}} \in \mathbb{F}_{p^m}. \tag{3.9}$$

Actually, the set $\{\beta + \beta^{-1}, (\beta + \beta^{-1})^p, \ldots, (\beta + \beta^{-1})^{p^{h'm-1}}\}$ on which type II–X NB is based is type–II ONB in $\mathbb{F}_{p^{h'm}}$. Because this NB exists whenever $8p \nmid m(p-1)$ or whenever $4p \nmid m(p-1)$ and $2 \nmid m$ [15], it is also available in $\mathbb{F}_{p^m}$ for every pair of characteristic p and extension degree m when $p > m$.

In the same manner as the other series, Kato et al. have especially called the extension field constructed by this NB type II–X AOPF, and have also proposed type II–X CVMA [15]. This algorithm is described in **Alg**. 5.

Algorithm 5: Type II–X CVMA [15]

Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}, \quad x_i, y_i \in \mathbb{F}_p.$

Output: $Z = X \cdot Y = \sum_{i=0}^{m-1} z_i \gamma^{p^i}, \quad z_i \in \mathbb{F}_p.$

Preparation steps:

Prepare a positive integer $h\,(= 2h')$ which satisfies **Cond**. 4.
for $i = 0$ **to** $m-1$ **do**
  for $k = 0$ **to** $h'-1$ **do** $\epsilon\left[\left\langle\!\left\langle p^{i+km} \right\rangle\!\right\rangle\right] \leftarrow i, \quad \epsilon\left[\left\langle\!\left\langle -p^{i+km} \right\rangle\!\right\rangle\right] \leftarrow i.$
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    for $k = 0$ **to** $h'-1$ **do**
      $\eta[i,j,k] \leftarrow \epsilon\left[\left\langle\!\left\langle p^i + p^{j+km} \right\rangle\!\right\rangle\right], \quad \eta[i,j,k+h'] \leftarrow \epsilon\left[\left\langle\!\left\langle p^i - p^{j+km} \right\rangle\!\right\rangle\right].$

Evaluation steps:

for $l = 0$ **to** $m-1$ **do** $v_l \leftarrow x_l y_l.$
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    $u \leftarrow (x_i - x_j)(y_i - y_j).$
    for $k = 0$ **to** $h-1$ **do** $v_{\eta[i,j,k]} \leftarrow v_{\eta[i,j,k]} + u.$
for $l = 0$ **to** $m-1$ **do** $z_l \leftarrow -v_l.$

3.1.5 Type–$\langle h, m \rangle$ Series

This subsection first introduces *Gauss period Normal Basis* (GNB) [22]. It is prepared with a positive integer h which satisfies the following condition.

Condition 5 (the h of GNB)

1) $r = hm + 1$ *is a prime number not equal to* p.

2) $\gcd(hm/e, m) = 1$, *where* e *is the order of* p *in* $\mathbb{F}_r$.

Let d be a primitive h–th root of unity in $\mathbb{F}_r^*$, then the following multiplicative group is obtained.

$$\left\langle \left\{ \left\langle\!\left\langle p^i d^k \right\rangle\!\right\rangle : 0 \le i < m,\, 0 \le k < h \right\}, \cdot \right\rangle = \mathbb{F}_r^*, \tag{3.10}$$

where $\langle\!\langle t \rangle\!\rangle$ denotes "$t \pmod r$" for an integer t and a prime number $r = hm + 1$. Here, let β be a primitive r–th root of unity in $\mathbb{F}_{p^e}^*$. In other words, β is a certain zero of the AOP in Eq. (3.2). GNB is defined with the above h, d and β as follows.

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}, \quad \gamma = \sum_{k=0}^{h-1} \beta^{d^k} \in \mathbb{F}_{p^m}. \tag{3.11}$$

This paper especially calls it type–$\langle h, m \rangle$ GNB. Since type–$\langle h, m \rangle$ GNB exists whenever $4p \nmid m(p-1)$ [22], it is also available in $\mathbb{F}_{p^m}$ for every pair of characteristic p and extension degree m when $p > m$. In the same manner as the other series, this thesis especially calls the extension field constructed by this GNB type–$\langle h, m \rangle$ AOPF.

From the viewpoint of type–$\langle h, m \rangle$ GNB, type–I and type–II ONBs are respectively characterized as type–$\langle h = 1, m \rangle$ and type–$\langle h = 2, m \rangle$ GNBs. They are very efficient normal bases; however, due to the restriction such that $h = 1$ or 2, they do not exist for an arbitrary pair of characteristic p and extension degree m. On the other hand, type I–X and II–X are also special classes of GNBs, in detail, type–$\langle h, m \rangle$ GNB with $d = \langle\!\langle p^m \rangle\!\rangle$ and type–$\langle h, m \rangle$ GNB with $d = \langle\!\langle -p^m \rangle\!\rangle$, respectively. Actually, the areas supported for the parameters p, m and h by the introduced bases are illustrated in **Fig**. 3.1. According to **Fig**. 3.1, there exists type–$\langle h, m \rangle$ GNB also in the area where type I–X and type II–X NBs can not support. Unfortunately, the previously introduced CVMAs are not available in this area.

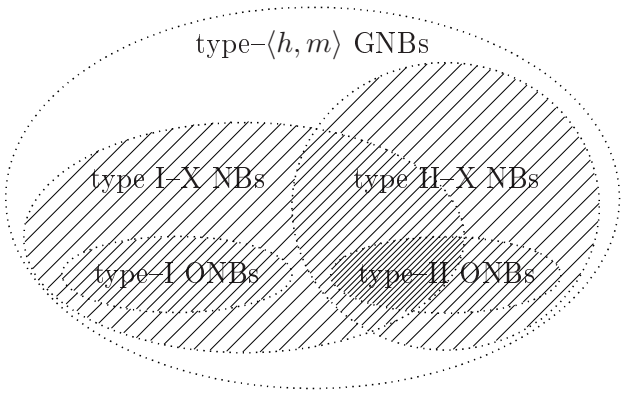


Figure 3.1: The simplified image of the relations among the normal bases in **Table** 3.1

.

3.2 Cyclic Vector Multiplication Algorithm for GNB

As shown in **Table** 3.1, efficient extension field multiplication algorithms, namely type–I, type–II, type I–X, and type II–X CVMAs, have respectively been proposed for type–I and type–II ONBs, and type I–X and type II–X NBs. As introduced in **Sec**. 3.1.5, these bases are special classes of type–$\langle h, m\rangle$ GNBs, that is, they has lower applicability for the *key* parameters p, m and h than type–$\langle h, m\rangle$ GNB. Thus, in order to take this applicability higher, the purpose of this section is to fully expand CVMA for type–$\langle h, m\rangle$ GNB as focused in **Table** 3.1 with **bold face** letters. Below, $\langle\langle t\rangle\rangle$ denotes "$t \pmod r$" for an integer t and a prime number $r = hm + 1$.

3.2.1 Deriving CVMA for type–$\langle h, m\rangle$ GNB

Let X and Y denote elements in $\mathbb{F}_{p^m}$, then they are represented with type–$\langle h, m\rangle$ GNB shown in Eq. (3.11) as follows.

$$X = \sum_{i=0}^{m-1} x_i \gamma^{p^i},\ \ Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i},\ \ x_i, y_i \in \mathbb{F}_p. \tag{3.12}$$

Then, a multiplication $Z = X \cdot Y$ is given by Eq. (3.13).

$$Z = \sum_{l=0}^{m-1} z_l \gamma^{p^l} = \sum_{i=0}^{m-1}\sum_{j=0}^{m-1} x_i y_j \gamma^{p^i+p^j} = \sum_{l=0}^{m-1} x_l y_l \Big(\gamma^{p^l} \sum_{i=0}^{m-1} \gamma^{p^i}\Big) - \sum_{0\le i<j<m}\sum (x_i - x_j)(y_i - y_j)\gamma^{p^i+p^j}. \tag{3.13}$$

According to Eq. (3.10), since the β is a zero of the AOP shown in Eq. (3.2), $\sum_{i=0}^{m-1} \gamma^{p^i}$ in Eq. (3.13) is obtained as follows.

$$\sum_{i=0}^{m-1} \gamma^{p^i} = \sum_{i=0}^{m-1}\sum_{k=0}^{h-1} \beta^{p^i d^k} = \sum_{i=1}^{r-1} \beta^i = -1. \tag{3.14}$$

On the other hand, $\gamma^{p^i+p^j}$ in Eq. (3.13) is obtained as follows.

$$\gamma^{p^i+p^j} = \sum_{k=0}^{h-1}\sum_{l=0}^{h-1} \beta^{p^i d^l + p^j d^k} = \sum_{k=0}^{h-1}\sum_{l=0}^{h-1} \beta^{(p^i + p^j d^{k-l}) d^l} = \sum_{k=0}^{h-1}\sum_{l=0}^{h-1} \beta^{(p^i + p^j d^k) d^l}. \tag{3.15}$$

$\sum_{l=0}^{h-1} \beta^{(p^i+p^j d^k)d^l}$ is derived as follows. Note that **Case 2** does not occur when h is even because $\big\langle\big\langle p^i + p^j d^k\big\rangle\big\rangle = 0$ only when $i = j$ (see **Proof** 1).

Case 1: When $\big\langle\big\langle p^i + p^j d^k\big\rangle\big\rangle \neq 0$ in $\mathbb{F}_r$, the following equation is obtained.

$$\sum_{l=0}^{h-1} \beta^{(p^i+p^j d^k)d^l} = \sum_{l=0}^{h-1} \beta^{p^{\epsilon\left[\left\langle\left\langle p^i+p^j d^k\right\rangle\right\rangle\right]} d^{\theta\left[\left\langle\left\langle p^i+p^j d^k\right\rangle\right\rangle\right]+l}} = \sum_{l=0}^{h-1} \big(\beta^{d^l}\big)^{p^{\epsilon\left[\left\langle\left\langle p^i+p^j d^k\right\rangle\right\rangle\right]}} = \gamma^{p^{\epsilon\left[\left\langle\left\langle p^i+p^j d^k\right\rangle\right\rangle\right]}}, \tag{3.16}$$

where ϵ and θ denote the following functions.

$$\epsilon\Big[\Big\langle\Big\langle p^i d^k\Big\rangle\Big\rangle\Big] = i,\ \ \theta\Big[\Big\langle\Big\langle p^i d^k\Big\rangle\Big\rangle\Big] = k \quad (0 \le i < m,\ 0 \le k < h). \tag{3.17}$$

Case 2: When $\big\langle\big\langle p^i + p^j d^k\big\rangle\big\rangle = 0$ in $\mathbb{F}_r$, according to Eq. (3.14), the following equation is obtained.

$$\sum_{l=0}^{h-1} \beta^{d^l (p^i+p^j d^k)} = \sum_{l=0}^{h-1} 1 = h = -h \sum_{l=0}^{m-1} \gamma^{p^l}. \tag{3.18}$$

Proof 1 (The pair of i and j such that $\langle\!\langle p^i + p^j d^k \rangle\!\rangle = 0$)
When $\langle\!\langle p^i + p^j d^k \rangle\!\rangle = 0$, let $l = j - i$ $(i \leq j)$, then the following relation holds.

$$1 = \left\langle\!\!\left\langle -p^l d^k \right\rangle\!\!\right\rangle. \tag{3.19}$$

Case 1: *When h is odd and thus m is even, the $2h$–th power of Eq. (3.19) becomes*

$$1 = \left\langle\!\!\left\langle p^{2hl} \right\rangle\!\!\right\rangle. \tag{3.20}$$

Since the order of p in $\mathbb{F}_r^*$ is e, there exists a primitive element $\hat{g}$ in $\mathbb{F}_r^*$ such that

$$\langle\!\langle p \rangle\!\rangle = \left\langle\!\!\left\langle \hat{g}^{h\hat{m}/e} \right\rangle\!\!\right\rangle. \tag{3.21}$$

Then, the following equation is obtained.

$$1 = \left\langle\!\!\left\langle p^{2hl} \right\rangle\!\!\right\rangle = \left\langle\!\!\left\langle \hat{g}^{2h\hat{a}l} \right\rangle\!\!\right\rangle \quad (\hat{a} = hm/e). \tag{3.22}$$

Thus, l needs to satisfy $hm \mid 2h\hat{a}l$ since the order of $\mathbb{F}_r^*$ is hm. According to the equations $\hat{a} = hm/e$ and $\gcd(hm/e, m) = 1$, the following equation is obtained.

$$\gcd(2h\hat{a}l, hm) = h \times \gcd(2\hat{a}l, m) = h \times \gcd(2l, m). \tag{3.23}$$

Since $0 \leq l < m$, the following relation holds only when $l = 0$ and $m/2$.

$$\gcd(2h\hat{a}l, hm) = hm. \tag{3.24}$$

When $l = 0$, according to Eq. (3.19), $\langle\!\langle -1 \rangle\!\rangle$ needs to be represented as a certain power of d; however, it does not because the order h of d in $\mathbb{F}_r^*$ is odd. Thus, Eq. (3.20) holds only when $l = i - j = m/2$.

Case 2*: When h is even and thus m is possible to be odd, the h–th power of Eq. (3.19) becomes*

$$1 = \left\langle\!\!\left\langle p^{hl} \right\rangle\!\!\right\rangle. \tag{3.25}$$

Then, in this case, the following equation is obtained.

$$1 = \left\langle\!\!\left\langle p^{hl} \right\rangle\!\!\right\rangle = \left\langle\!\!\left\langle \hat{g}^{h\hat{a}l} \right\rangle\!\!\right\rangle \quad (\hat{a} = hm/e). \tag{3.26}$$

Thus, l needs to satisfy that $hm \mid h\hat{a}l$. According to the equations $\hat{a} = hm/e$ and $\gcd(hm/e, m) = 1$, in this case, the following equation is obtained.

$$\gcd(h\hat{a}l, hm) = h \times \gcd(\hat{a}l, m) = h \times \gcd(l, m). \tag{3.27}$$

Since $0 \leq l < m$, the following relation holds only when $l = 0$.

$$\gcd(h\hat{a}l, hm) = hm. \tag{3.28}$$

Thus, Eq. (3.25) holds when $l = i - j = 0$. □

From Eqs. (3.14), (3.15), (3.16), (3.18), Z in Eq. (3.13) is given by Eq. (3.29), where δ_s denotes the unit impulse function as Eq. (3.30), and $\bar{\delta}_s$ denotes $1-\delta_s$.

$$Z = \sum_{l=0}^{m-1} z_l \gamma^{p^l} = -\sum_{l=0}^{m-1} x_l y_l \gamma^{p^l} - \sum\sum_{0\le i<j<m} (x_i - x_j)(y_i - y_j) \sum_{k=0}^{h-1} \bar{\delta}_0\Big[\Big\langle\!\Big\langle p^i + p^j d^k \Big\rangle\!\Big\rangle\Big] \gamma^{p^{\epsilon}\left[\langle\!\langle p^i + p^j d^k \rangle\!\rangle\right]} + \begin{cases} h \sum\sum_{0\le i<j<m} (x_i - x_j)(y_i - y_j) \sum_{k=0}^{h-1} \delta_0\Big[\Big\langle\!\Big\langle p^i + p^j d^k \Big\rangle\!\Big\rangle\Big] \sum_{l=0}^{m-1} \gamma^{p^l} & \text{(when } h \text{ is odd)}, \\ 0 & \text{(when } h \text{ is even)}. \end{cases} \tag{3.29}$$

$$\delta_s(t) = \begin{cases} 1 & \text{(when } s = t\text{)}, \\ 0 & \text{(otherwise)}, \end{cases} \tag{3.30}$$

Let ϵ in Eq. (3.17) be redefined as a function which also satisfies as Eq. (3.31), and η denotes the following function.

$$\epsilon[0] = m, \quad \eta[i,j,k] = \epsilon\Big[\Big\langle\!\Big\langle p^i + p^j d^k \Big\rangle\!\Big\rangle\Big]. \tag{3.31}$$

Then, z_l in Eq. (3.29) is calculated as follows.

$$z_l = \begin{cases} h v_m - v_l - x_l y_l & \text{(when } h \text{ is odd)}, \\ -v_l - x_l y_l & \text{(when } h \text{ is even)}, \end{cases} \quad v_l = \sum\sum_{0\le i<j<m} (x_i - x_j)(y_i - y_j) \sum_{k=0}^{h-1} \delta_l\big[\eta[i,j,k]\big]. \tag{3.32}$$

Consequently, the CVMA expanded for type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$, namely type–$\langle h, m\rangle$ CVMA, is constructed as shown in **Alg**. 6.

Note that the preparation steps (Step. 1 to 7) in **Alg**. 6 is performed only once when p and m are fixed. Thus, the computational cost of type–$\langle h, m\rangle$ CVMA is explicitly given as follows.

$$M_m(h) = \frac{m(m+1)}{2} M_1 + \begin{cases} \left(\frac{m(m-1)(h+2)}{2} - 1 + m\right) A_1 + H_1 & \text{(when } h \text{ is odd)}, \\ \left(\frac{m(m-1)(h+2)}{2}\right) A_1 & \text{(when } h \text{ is even)}, \end{cases} \tag{3.33}$$

where H_m denotes the calculation cost of a scalar–h multiplication in $\mathbb{F}_{p^m}$. This computational cost is equal to those of type I–X and type II–X CVMAs, and they need more additions in $\mathbb{F}_p$ as h becomes larger. Thus, in order to more efficiently perform these CVMAs, h should be as small as possible, furthermore it is the most desirable that $h = 1$ or $h = 2$. Actually, type–I and type–$\langle h{=}1, m\rangle$ CVMAs are not algorithmically equivalent but they have the same computational cost. In the same, type–II and type–$\langle h{=}2, m\rangle$ CVMAs have the same computational cost.

3.2.2 Experimental Result

The author experimented on a few software implementations of type–$\langle h, m\rangle$ CVMA. As the characteristic p, this experimentation adopted a 256–bit prime number, which is the same scale of the implementation of Ate–type pairing [8]. In this case, since multiple precision arithmetic operations were necessary, the GNU MP (GMP) arithmetic library [37] was utilized. Note that

Algorithm 6: Type–$\langle h, m\rangle$ CVMA

Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}, \quad x_i, y_i \in \mathbb{F}_p.$

Output: $Z = X \cdot Y = \sum_{i=0}^{m-1} z_i \gamma^{p^i}, \quad z_i \in \mathbb{F}_p.$

Preparation steps:

Prepare a positive integer h which satisfies **Cond**. 5,
and a primitive h–th root d of unity in $\mathbb{F}_r$.
$\epsilon[0] \leftarrow m$.
for $i = 0$ **to** $m-1$ **do**
  for $k = 0$ **to** $h-1$ **do** $\epsilon\big[\langle\!\langle p^i d^k \rangle\!\rangle\big] \leftarrow i$.
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    for $k = 0$ **to** $h-1$ **do** $\eta[i,j,k] \leftarrow \epsilon\big[\langle\!\langle p^i + p^j d^k \rangle\!\rangle\big]$.

Evaluation steps:

for $l = 0$ **to** $m-1$ **do** $v_l \leftarrow x_l y_l$.
$v_m \leftarrow 0$.
for $i = 0$ **to** $m-2$ **do**
  for $j = i+1$ **to** $m-1$ **do**
    $u \leftarrow (x_i - x_j)(y_i - y_j)$.
    for $k = 0$ **to** $h-1$ **do** $v_{\eta[i,j,k]} \leftarrow v_{\eta[i,j,k]} + u$.
if h *is odd* **then**
  $w \leftarrow h v_m$.
  for $l = 0$ **to** $m-1$ **do** $z_l \leftarrow w - v_l$.
else for $l = 0$ **to** $m-1$ **do** $z_l \leftarrow -v_l$.

not the so–called *Integer Functions* whose name prefixes are `mpz_` but *Low–level Functions* whose name prefixes are `mpn_` in the library are adopted in order to achieve high–speed performance. This experimentation employed both the 32–bit and 64–bit computation environments described in **Table** 3.2. Then, the computation time of each arithmetic operation in $\mathbb{F}_p$ was obtained as shown in **Table** 3.3. Note that *Montgomery reduction* technique [38] was applied to perform a reduction modulo p for every multiplication in $\mathbb{F}_p$. According to **Table** 3.3, it is found that the ratio $\nu = M_1/A_1$ in the 32–bit environment is larger than that in the 64–bit environment. This suggests that, in the 64–bit environment compared to the 32–bit environment, type–$\langle h, m\rangle$ CVMA is greater influenced by h since it needs more additions in $\mathbb{F}_p$ as h becomes larger.

On the other hand, the computation time of a multiplication in $\mathbb{F}_{p^m}$ with type–$\langle h, m\rangle$ CVMA were obtained as shown in **Tables** 3.4, 3.5, where 2 kinds of times are described for every pair of m and h. One means the actual measured time in the environment shown in **Table** 3.3, and another means the theoretical time obtained by assigning M_1, A_1 and H_1 in **Table** 3.3 to $M_m(h)$ in Eq. (3.33). From the experimental results, it is found that a little difference occurs between the each actual measured time and the corresponding theoretical time; however, the every difference is less than about 5%, that is, negligibly small.

Table 3.2: The computational environment

	32–bit environment	64–bit environment
CPU	Core i3–620 3.06GHz (It has a dual core; however, only 1 core was worked.)	
Cache	2–nd: 512 KB × 2, 3–rd: 4.0 MB	
Memory	2.0 GB × 4 (in dual–channel configuration)	
OS	Ubuntu 10.10 32–bit version	Ubuntu 10.10 64–bit version
Language	C	
Compiler	GCC 4.4.5 32–bit version	GCC 4.4.5 64–bit version
Optimization	"–O3 –m32" compiler option	"–O3 –m64" compiler option
Library	GNU MP 5.0.1 [37]	

Table 3.3: The computation time of each arithmetic operation in $\mathbb{F}_p$

	32–bit environment	64–bit environment	mainly utilized functions
addition (subtraction)	$A_1 = 25$ $n$sec.	$A_1 = 13$ $n$sec.	`mpn_add_n`, `mpn_sub_n`
multiplication	$M_1 = 199$ $n$sec. ($\nu = M_1/A_1 \approx 8.0$)	$M_1 = 71$ $n$sec. ($\nu = M_1/A_1 \approx 5.5$)	`mpn_mul_basecase`†, `mpn_redc_1`†
scalar–h multiplication	$H_1 = 45$ $n$sec. ($\xi = H_1/A_1 \approx 1.8$)	$H_1 = 27$ $n$sec. ($\xi = H_1/A_1 \approx 2.1$)	`mpn_mul_1`, `mpn_sub_n`

† Actually, these functions are declared in the archive file "libgmp.a"; however, there do not exist the prototype declarations of the functions in the header file "gmp.h". Thus, in order to utilize these functions, the author appropriately edited "gmp.h".

Table 3.4: The computation time of a multiplication in $\mathbb{F}_{p^m}$ (the 32–bit environment) [μsec]

h \ m	2	3	4	5	6	7	8	9	10	11	12
1	0.77 0.70	— (—)	2.65 (2.52)	— (—)	5.57 (5.43)	— (—)	— (—)	— (—)	14.8 (14.5)	— (—)	21.0 (20.7)
2	0.76 (0.70)	1.56 (1.49)	— (—)	4.14 (3.99)	5.94 (5.68)	— (—)	10.3 (9.96)	12.8 (12.6)	— (—)	19.2 (18.6)	— (—)
3	0.81 (0.79)	— (—)	2.93 (2.86)	— (—)	6.35 (6.22)	— (—)	— (—)	— (—)	17.0 (16.8)	— (—)	24.2 (24.1)
4	— (—)	1.72 (1.64)	2.99 (2.89)	— (—)	— (—)	8.90 (8.72)	— (—)	14.5 (14.4)	18.1 (17.7)	— (—)	— (—)
5	0.87 (0.84)	— (—)	— (—)	— (—)	7.24 (6.97)	— (—)	12.5 (12.3)	— (—)	— (—)	— (—)	27.6 (27.4)
6	0.85 (0.80)	1.85 (1.79)	— (—)	5.07 (4.99)	7.65 (7.18)	10.3 (9.77)	— (—)	— (—)	20.6 (19.9)	24.8 (24.1)	29.3 (28.7)
7	— (—)	— (—)	3.49 (3.46)	— (—)	8.22 (7.72)	— (—)	— (—)	— (—)	22.1 21.3	— (—)	— (—)
8	0.88 (0.85)	— (—)	— (—)	5.53 (5.49)	— (—)	— (—)	— (—)	19.0 (18.0)	— (—)	28.2 (26.9)	33.5 (32.0)
9	0.95 (0.94)	— (—)	3.79 (3.76)	— (—)	— (—)	— (—)	15.3 (15.1)	— (—)	— (—)	— (—)	35.0 (34.0)
10	— (—)	2.15 (2.09)	3.83 (3.79)	— (—)	9.07 (8.68)	12.5 (11.9)	— (—)	— (—)	25.2 (24.4)	— (—)	— (—)
11	1.01 (0.99)	— (—)	— (—)	— (—)	9.44 (9.22)	— (—)	17.4 (16.5)	— (—)	— (—)	— (—)	— (—)
12	— (—)	2.26 (2.24)	— (—)	6.61 (6.49)	9.60 (9.43)	— (—)	17.5 (17.0)	23.2 (21.6)	— (—)	— (—)	— (—)
13	— (—)	— (—)	4.36 (4.36)	— (—)	10.0 (9.97)	— (—)	— (—)	— (—)	29.1 (28.1)	— (—)	42.3 (40.6)
14	1.02 (1.00)	2.43 (2.39)	— (—)	6.99 (6.99)	— (—)	— (—)	18.6 (18.4)	25.0 (23.4)	— (—)	— (—)	— (—)
15	1.10 1.09	— (—)	4.60 (4.60)	— (—)	— (—)	— (—)	— (—)	— (—)	31.6 (30.3)	— (—)	46.3 (43.9)
16	— (—)	— (—)	— (—)	— (—)	10.9 (10.9)	15.2 (15.0)	— (—)	— (—)	— (—)	— (—)	47.4 (45.2)
17	— (—)	— (—)	— (—)	— (—)	11.5 (11.5)	— (—)	20.8 (20.7)	— (—)	— (—)	— (—)	— (—)
18	1.13 1.10	— (—)	4.99 (4.99)	— (—)	11.7 (11.7)	16.1 (16.1)	— (—)	27.1 (27.0)	33.4 (33.4)	41.9 (40.6)	— (—)
19	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	35.9 (34.8)	— (—)	49.2 (50.5)
20	1.17 (1.15)	2.84 (2.84)	— (—)	8.49 (8.49)	— (—)	— (—)	— (—)	28.9 (28.8)	— (—)	— (—)	50.5 51.8

† The time outside each paranthesis () is the actual measured time, and the time inside each paranthesis () is the theoretical time.

Table 3.5: The computation time of a multiplication in $\mathbb{F}_{p^m}$ (the 64–bit environment) [μsec]

h \ m	2	3	4	5	6	7	8	9	10	11	12
1	0.29 (0.27)	— (—)	1.02 (0.98)	— (—)	2.26 (2.14)	— (—)	— (—)	— (—)	6.03 (5.78)	— (—)	8.57 (8.26)
2	0.29 (0.27)	0.59 (0.58)	— (—)	1.70 (1.59)	2.45 (2.27)	— (—)	4.26 (4.01)	5.35 (5.07)	— (—)	7.92 (7.55)	— (—)
3	0.32 (0.32)	— (—)	1.23 (1.17)	— (—)	2.71 (2.56)	— (—)	— (—)	— (—)	7.39 (6.97)	— (—)	10.4 (10.0)
4	— (—)	0.69 (0.66)	1.24 (1.18)	— (—)	— (—)	3.84 (3.63)	— (—)	6.53 (6.00)	7.77 (7.42)	— (—)	— (—)
5	0.34 (0.34)	— (—)	— (—)	— (—)	3.09 (2.95)	— (—)	5.68 (5.22)	— (—)	— (—)	— (—)	12.1 (11.7)
6	0.33 (0.32)	0.77 (0.74)	— (—)	2.21 (2.11)	3.20 (3.05)	4.37 (4.17)	— (—)	— (—)	8.93 (8.59)	10.8 (10.4)	12.8 (12.4)
7	— (—)	— (—)	1.54 (1.48)	— (—)	2.50 (3.34)	— (—)	— (—)	— (—)	9.59 (9.31)	— (—)	— (—)
8	0.35 (0.34)	— (—)	— (—)	2.47 (2.37)	— (—)	— (—)	— (—)	8.13 (7.88)	— (—)	12.2 (11.8)	14.5 (14.1)
9	0.40 (0.40)	— (—)	1.69 (1.63)	— (—)	— (—)	— (—)	6.87 (6.68)	— (—)	— (—)	— (—)	15.5 (15.1)
10	— (—)	0.94 (0.89)	1.71 (1.65)	— (—)	3.98 (3.83)	5.45 (5.26)	— (—)	— (—)	11.2 (10.9)	— (—)	— (—)
11	0.42 (0.42)	— (—)	— (—)	— (—)	4.26 (4.12)	— (—)	7.60 (7.41)	— (—)	— (—)	— (—)	— (—)
12	— (—)	1.01 (0.97)	— (—)	2.99 (2.89)	4.36 (4.22)	— (—)	7.88 (7.65)	9.97 (9.75)	— (—)	— (—)	— (—)
13	— (—)	— (—)	2.00 (1.95)	— (—)	4.64 (4.51)	— (—)	— (—)	— (—)	13.0 (12.8)	— (—)	19.1 (18.6)
14	0.42 (0.42	1.09 (1.05)	— (—)	3.25 (3.15)	— (—)	— (—)	8.60 (8.38)	10.9 (10.7)	— (—)	— (—)	— (—)
15	0.48 (0.47)	— (—)	2.15 (2.10)	— (—)	— (—)	— (—)	— (—)	— (—)	14.2 (14.0)	— (—)	21.0 (20.3)
16	— (—)	— (—)	— (—)	— (—)	5.36 (5.00)	7.06 (6.90)	— (—)	— (—)	— (—)	— (—)	22.3 (21.0)
17	— (—)	— (—)	— (—)	— (—)	5.43 (5.29)	— (—)	9.74 (9.59)	— (—)	— (—)	— (—)	— (—)
18	0.49 (0.47)	— (—)	2.32 (2.27)	— (—)	5.50 (5.39)	7.61 (7.45)	— (—)	13.0 (12.6)	15.8 (15.6)	19.9 (19.0)	— (—)
19	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	— (—)	16.6 (16.3)	— (—)	25.4 (23.7)
20	0.51 (0.50)	1.33 (1.28)	— (—)	4.01 (3.93)	— (—)	— (—)	— (—)	13.7 (13.5)	— (—)	— (—)	26.1 (24.4)

† The time outside each paranthesis () is the actual measured time, and the time inside each paranthesis () is the theoretical time.

3.3 Existence Probability of GNB

This section mainly provides an important theorem to theoretically evaluate the efficiency of type–$\langle h, m\rangle$ CMVA proposed in the previous section. Below, "$\sim$" and "$\approx$" respectively denote the theoretical approximation and truncation operators.

3.3.1 Theorem to Derive Existence Probability

The existence probability of type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$ is given by **Theo**. 7 [36].

Theorem 7 *The Euler's totient function for a positive integer t is denoted by $\varphi(t)$. It indicates the number of positive integers less than t which are coprime to t. Let $\mathcal{P}(h, m)$ be the possibility that the parameter h such that* **Cond**. *5.1 is satisfied also satisfies* **Cond**. *5.2, namely the existence probability of type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$. Then, $\mathcal{P}(h, m)$ is given by*

$$\mathcal{P}(h, m) \sim \varphi(m)/m. \tag{3.34}$$

□

For example, consider when $m = 6$. In this case, the possible positive integers h's which satisfy **Cond**. 5.1 are obtained as follows.

$$h = 1, 2, 3, 5, 6, 7, 10, 11, 12, 13, \ldots \tag{3.35}$$

Among such positive integers h's, consider the probability that the minimal positive integer $h_{\min}$ additionally satisfies **Cond**. 5.2 but the others smaller than $h_{\min}$ do not. The existence probability of type–$\langle h_{\min}, m\rangle$ GNB in $\mathbb{F}_{p^m}$ is denoted by $\tilde{\mathcal{P}}(h_{\min}, m)$, where p is a variable prime number. For example, in the case that $h_{\min} = 3$, $\tilde{\mathcal{P}}(3, 6)$ can be approximately calculated with **Theo**. 7 as follows.

$$\tilde{\mathcal{P}}(3,6) = \{1 - \mathcal{P}(1,6)\} \times \{1 - \mathcal{P}(2,6)\} \times \mathcal{P}(3,6) \sim \Big(1 - \frac{\varphi(6)}{6}\Big)^2 \Big(\frac{\varphi(6)}{6}\Big) \approx 0.148. \tag{3.36}$$

When $m=6$, the expected value of $h_{\min}$ can be approximately obtained with **Theo**. 7 as follows.

$$\begin{aligned} &1\cdot\tilde{\mathcal{P}}(1,6) + 2\cdot\tilde{\mathcal{P}}(2,6) + 3\cdot\tilde{\mathcal{P}}(3,6) + \cdots \\ &\qquad \sim 1\Big(\frac{\varphi(6)}{6}\Big) + 2\Big(1-\frac{\varphi(6)}{6}\Big)\Big(\frac{\varphi(6)}{6}\Big) + 3\Big(1-\frac{\varphi(6)}{6}\Big)^2\Big(\frac{\varphi(6)}{6}\Big) + \cdots \approx 3.53. \end{aligned} \tag{3.37}$$

As described above, by utilizing **Theo**. 7, $\tilde{\mathcal{P}}(h_{\min}, m)$ for every pair $(h_{\min}, m)$ and the expected value of $h_{\min}$ for every m can be approximately calculated.

Actually, the following two theorems are important for a proof of **Theo**. 7.

Theorem 8 (Prime Number Theorem (PNT) [39])
For a positive integer t, let $\pi(t)$ be the number of prime numbers less than t. Then, $\pi(t)$ is obtained as follows, where $\mathrm{Li}(t)$ *denotes the offset logarithmic integral function.*

$$\pi(t) \sim \mathrm{Li}(t) = \int_2^t \frac{1}{\ln(s)} \mathrm{d}s, \tag{3.38}$$

□

Theorem 9 (PNT for arithmetic progressions [39])
For a positive integer t, let $\pi_{n,c}(t)$ be the number of prime numbers less than t in the arithmetic progression with first term c and the common difference n such that $\gcd(c,n)=1$. Then, $\pi_{n,c}(t)$ is given by

$$\pi_{n,c}(t) \sim \mathrm{Li}(t)/\varphi(n). \tag{3.39}$$

□

Below, **Theo**. 7 is theoretically proven with **Theos**. 8, 9.

Proof 2 *Suppose that* **Cond**. *5.1 is satisfied, in other words, $r(=hm+1)$ is a prime number not equal to p. Let p and d, namely the characteristic and a primitive h–th root of unity in $\mathbb{F}_r^*$, be respectively represented as the following powers of primitive elements g_1 and g_2 in $\mathbb{F}_r^*$.*

$$\langle\langle p \rangle\rangle = \langle\langle {g_1}^{a_1} \rangle\rangle \quad (0 \le a_1 < r-1 = hm), \qquad \langle\langle d \rangle\rangle = \langle\langle {g_2}^{a_2} \rangle\rangle \quad (0 \le a_2 < r-1 = hm), \tag{3.40}$$

where $\langle\langle t \rangle\rangle$ denotes "$t \pmod r$" for a positive integer t. Consider the following set of the products.

$$\left\{ \left\langle\left\langle p^i d^k \right\rangle\right\rangle : 0 \le i < m,\ 0 \le k < h \right\}. \tag{3.41}$$

Actually, the set in Eq. (3.41) becomes the same of Eq. (3.10) when a_1 and a_2 in Eq. (3.40) satisfy the following relation.

$$\gcd\left(\gcd(a_1,a_2), hm\right) = 1. \tag{3.42}$$

Thus, this thesis gives **Cond**. *5.2 as one of the existence conditions of type-$\langle h,m\rangle$ GNB in $\mathbb{F}_{p^m}$ because the smallest a_1 and a_2 which satisfy Eq. (3.40) are respectively given as hm/e and m, where e denotes the order of p in $\mathbb{F}_r^*$.*

Here, let a_2 in Eq. (3.40) be fixed at the minimal, namely m. Then, in order to guarantee that type-$\langle h,m\rangle$ GNB exists in $\mathbb{F}_{p^m}$, a_1 in Eq. (3.40) needs to satisfy the following relation.

$$\gcd(\gcd(a_1,m), hm) = \gcd(a_1, \gcd(m, hm)) = \gcd(a_1, m) = 1. \tag{3.43}$$

Let a_1 be represented by

$$a_1 = bm + a_1' \quad (0 \le b < h,\ 0 \le a_1' < m), \tag{3.44}$$

then Eq. (3.43) is reduced with the Euclidean algorithm as follows.

$$\gcd(a_1, m) = \gcd(a_1', m) = 1, \tag{3.45}$$

The number of a_1''s which satisfy Eq. (3.45) is given by $\varphi(m)$. Thus, the number of a_1's which satisfies Eq. (3.43) is obtained as follows.

$$h\varphi(m). \tag{3.46}$$

On the other hand, reconsider a variable prime number p less than a positive integer t, then $\langle\langle p \rangle\rangle$ in Eq. (3.40) is again given by

$$\langle\langle p \rangle\rangle = \langle\langle {g_1}^{a_1} \rangle\rangle = i \quad (0 \le a_1 < r-1 = hm,\ 1 \le i \le r-1 = hm). \tag{3.47}$$

According to Eq. (3.48), a_1 and i correspond one–to–one with each other because the following relation holds.

$$\{\langle\langle g_1^{a_1} \rangle\rangle :\ 0 \le a_1 < r-1 = hm\} = \{i : 1 \le i \le r-1 = hm\} = \mathbb{F}_r^*. \tag{3.48}$$

Thus, from Eq. (3.48) and **Theos**. *8, 9, the ratio of the number of p's with one of a_1's is obtained as follows.*

$$\pi_{r,i}(t)/\pi(t) \sim (\mathrm{Li}(t)/\varphi(r))/\mathrm{Li}(t) \sim 1/\varphi(r) = 1/hm. \tag{3.49}$$

Thus, based on Eqs. (3.46), (3.49), the probability that Eq. (3.45) is satisfied, in other words, the existence probability of type-$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$ is given by

$$\mathcal{P}(h, m) \sim (h\varphi(m))/(hm) = \varphi(m)/m. \tag{3.50}$$

□

In the same way, let $\mathcal{P}_{\text{I–X}}(h, m)$ and $\mathcal{P}_{\text{II–X}}(h, m)$ respectively denote the existence probabilities of type I–X and II–X NBs. Then, they are obtained as follows, where h' is $h/2$.

$$\mathcal{P}_{\text{I–X}}(h, m) \sim \quad \varphi(hm)/(hm), \tag{3.51a}$$

$$\mathcal{P}_{\text{II–X}}(h, m) \sim \begin{cases} \varphi(hm)/(hm) + \varphi(h'm)/(h'm) & (\text{when } m \text{ is odd}), \\ \varphi(hm)/(hm) & (\text{when } m \text{ is even}), \end{cases} \tag{3.51b}$$

It is obvious that $\varphi(m)/m$ is larger than $\varphi(hm)/(hm)$ and $\varphi(hm)/(hm) + \varphi(h'm)/(h'm)$. It means that type-$\langle h, m\rangle$ GNB will help to keep the minimal $h_{\min}$ small, compared to type I–X NB and type II–X NB.

3.3.2 Evaluation with Existence Probabilities

The existence probability of each normal basis can be calculated with **Theo**. 7. The calculation result is illustrated in **Fig**. 3.2, where the size of each h of type I–X, type II–X NBs and type-$\langle h, m\rangle$ GNB is not restricted. As shown in **Fig**. 3.2, only type–I NB and type–I NB can not support a lot of pairs of characteristic p and m. Thus, if the efficiencies of their CVMAs are not strictly evaluated, it will be maintained that type I–X, type II–X NBs and type-$\langle h, m\rangle$ GNB are quite useful since they are available for most of pairs of p and m. Then, as indicated in **Fig**. 3.2, there is no large difference among the existence probabilities of type I–X, type II–X NBs and type-$\langle h, m\rangle$ GNB.

However, actually type I–X, type II–X NBs and type-$\langle h, m\rangle$ GNB have different efficiencies corresponding to h. Hence, suppose that ν and ξ respectively denote M_1/A_1 and H_1/A_1, then let us first confirm the relation among the efficiency of type-$\langle h, m\rangle$ CVMA and the parameters ν, ξ and h in each case, namely when h is odd and when h is even.

Case 1: Consider the case that h is odd. Type-$\langle h{=}1, m\rangle$ CVMA, one of the most efficient type-$\langle h, m\rangle$ CVMAs, has the following computatinal cost.

$$M_m(1) = \frac{m(m{+}1)}{2}M_1 + \left(\frac{(m{-}1)(3m{+}2)}{2}\right)A_1 = \left(\frac{m(m{+}1)}{2}\nu + \frac{(m{-}1)(3m{+}2)}{2}\right)A_1. \tag{3.52}$$

The increased amount of the computational cost of type-$\langle h, m\rangle$ CVMA compared to type-$\langle h{=}1, m\rangle$ CVMA is given by

$$M_m(h) - M_m(1) = \left(\frac{m(m{-}1)(h{-}1)}{2} + \xi\right)A_1. \tag{3.53}$$

Thus, the ratio $R_{\mathrm{odd}}(h,m)$ of the increase is obtained as follows

$$R_{\mathrm{odd}}(h,m) = \frac{M_m(h) - M_m(1)}{M_m(1)} = \frac{h - 1 + \frac{1}{m(m-1)}}{\frac{m+1}{m-1}\nu + 3 + \frac{2}{m}}. \tag{3.54}$$

Case 2: Consider the case that h is even. Type–$\langle h{=}2, m\rangle$ CVMA, one of the most efficient type–$\langle h, m\rangle$ CVMAs, has the following computational cost.

$$M_m(2) = \frac{m(m+1)}{2}M_1 + \left(\frac{4m(m-1)}{2}\right)A_1 = \left(\frac{m(m+1)}{2}\nu + \frac{4m(m-1)}{2}\right)A_1. \tag{3.55}$$

When h is even, the increased amount of the computational cost of type–$\langle h, m\rangle$ CVMA compared to type–$\langle h{=}2, m\rangle$ CVMA is given by

$$M_m(h) - M_m(2) = \left(\frac{m(m-1)}{2}h - \frac{2m(m-1)}{2}\right)A_1. \tag{3.56}$$

Thus, the ratio $R_{\mathrm{odd}}(h,m)$ of the increase is obtained as follows.

$$R_{\mathrm{even}}(h,m) = \frac{M_m(h) - M_m(2)}{M_m(2)} = \frac{h-2}{\frac{m+1}{m-1}\nu + 4}. \tag{3.57}$$

As described above, through the increased amount for m of the computational cost of type–$\langle h, m\rangle$ CVMA compared to type–$\langle h{=}1, m\rangle$ or type–$\langle h{=}2, m\rangle$ CVMA, it is found that type–$\langle h, m\rangle$ CVMA fatally becomes inefficient when h is large. Thus, this thesis recommends the h's such that both $R_{\mathrm{odd}}(h,m)$ and $R_{\mathrm{even}}(h,m)$ are somewhat small as *practical* parameters. According to **Sec**. 3.2.2 (**Table** 3.3), an experimental result was obtained such that $\nu \approx 8.0$ and $\xi \approx 1.8$ in the 32–bit environment, and another result was obtained such that $\nu \approx 5.5$ and $\xi \approx 2.1$ in the 64–bit environment. Then, the existence probability of each normal basis such that $R_{\mathrm{odd}}(h,m)$ and $R_{\mathrm{even}}(h,m)$ are less than a certain threshold, namely from 0.1 to 0.5, becomes as **Figs**. 3.4, 3.5. These tables guarantee that the number of useful type–$\langle h, m\rangle$ GNBs is larger than that of type I–X and type II–X NBs. In other words, on the practical side, type–$\langle h, m\rangle$ CVMA works faster than type I–X and II–X CVMA. Here, let the existence probabilities in the 32–bit and 64–bit environments be compared, then according to **Figs**. 3.4, 3.5, it is found that the latters form a more sharp zigzag shape than the formers. As the cause for this result, it is considered that type–$\langle h, m\rangle$ CVMA is greater influenced by h in the 64–bit environment compared to the 32–bit environment as describe in **Sec**. 3.2.2.

On the other hand, the expected value of h_{min} for every m of each normal basis can be also calculated with **Theo**. 7. The calculation result is illustrated in **Fig**. 3.3. According to **Fig**. 3.3, the expected value of h_{min} for each m of type–$\langle h, m\rangle$ GNB is smaller than those of type I–X and type II–X NBs. Thus, it is said that type–$\langle h, m\rangle$ CVMA will be averagely faster than type I–X and type II–X CVMAs. However, the gained speed–up is averagely a few percents.

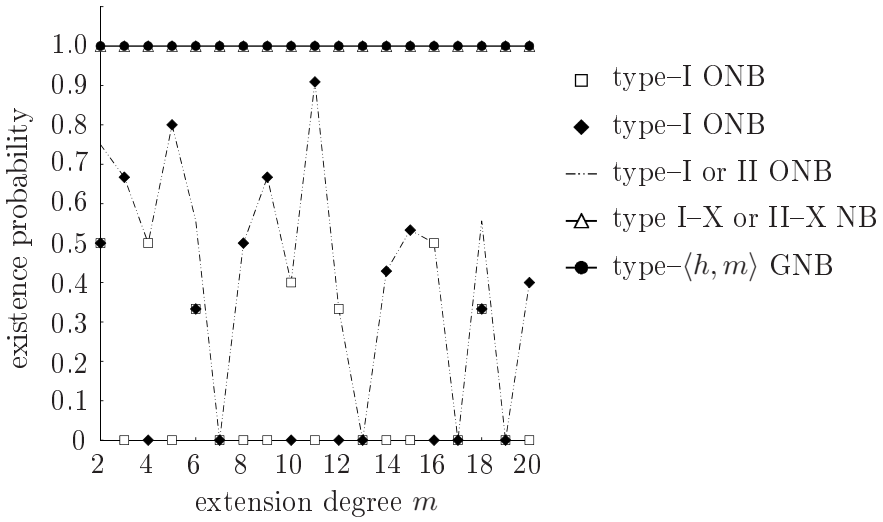


Figure 3.2: The existence probability of each normal basis

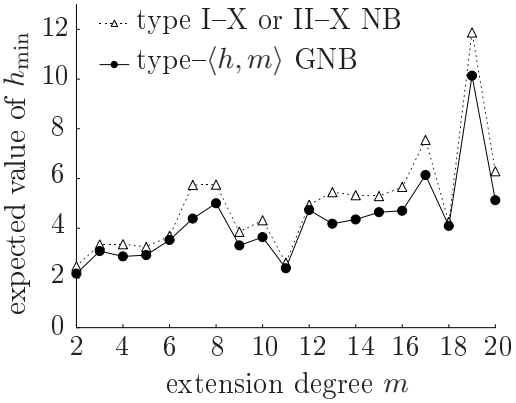
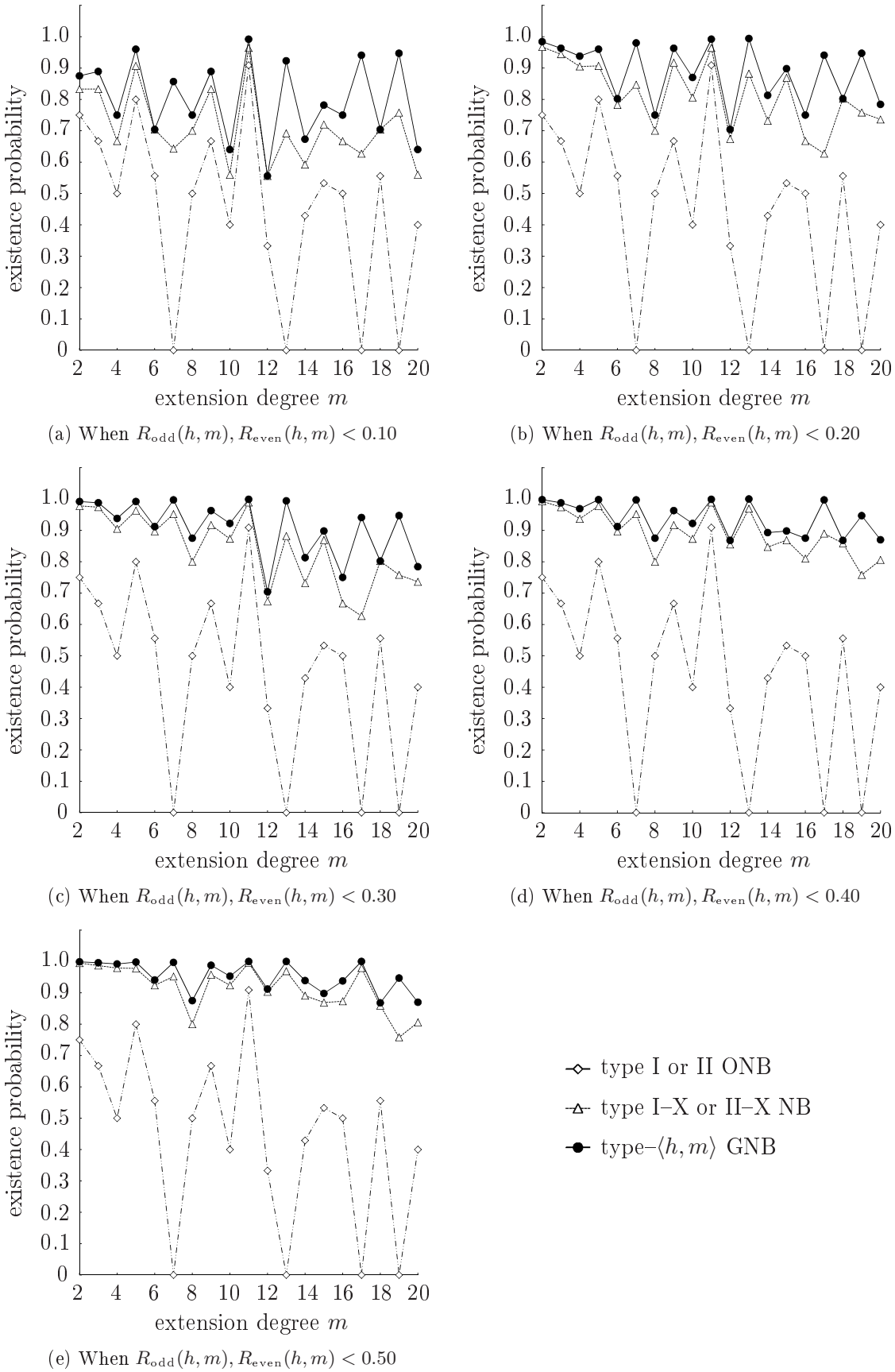


Figure 3.3: The expected value of $h_{\min}$



(a) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.10$

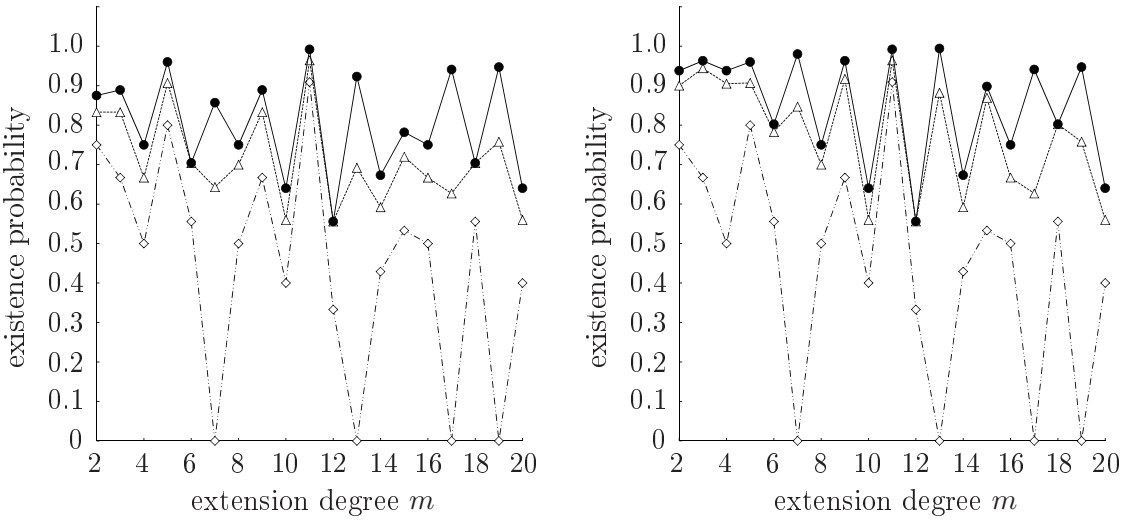
(b) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.20$

(c) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.30$

(d) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.40$

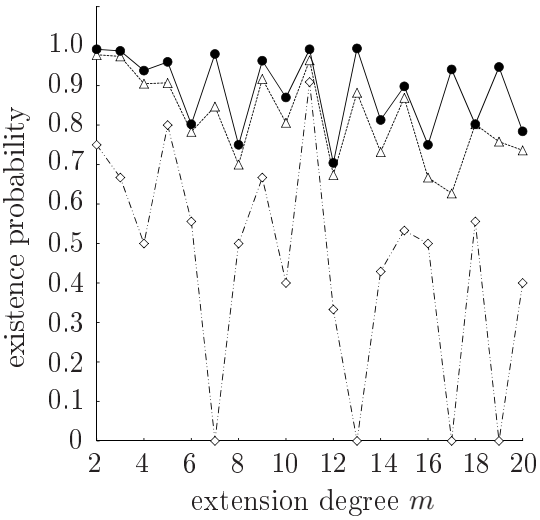
(e) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.50$

Figure 3.4: The existence probability of each practical normal basis in the 32–bit environment

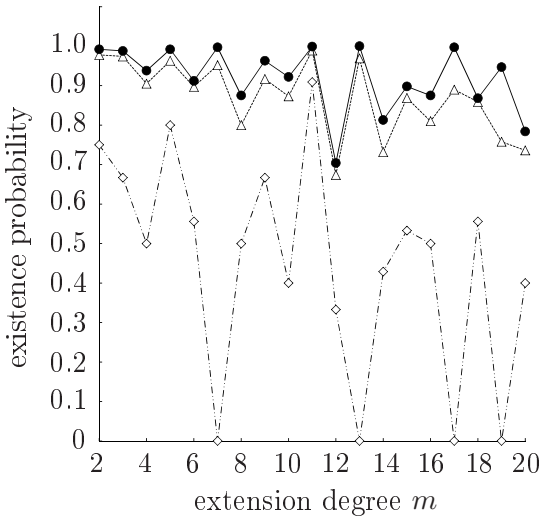


(a) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.10$

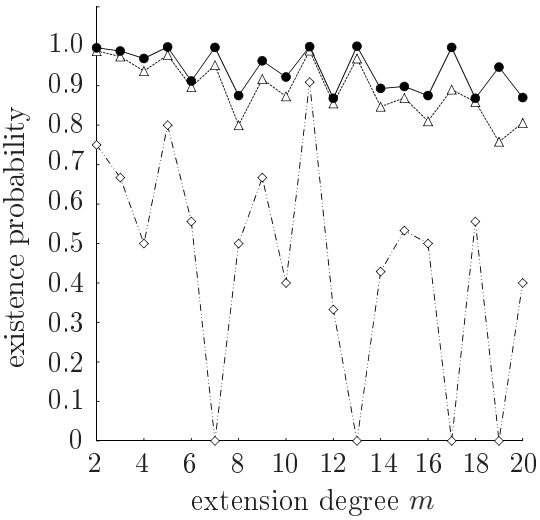
(b) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.20$



(c) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.30$



(d) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.40$



(e) When $R_{\mathrm{odd}}(h,m), R_{\mathrm{even}}(h,m) < 0.50$

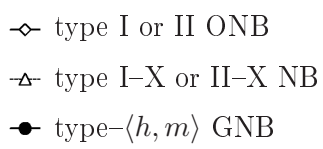


Figure 3.5: The existence probability of each practical normal basis in the 64–bit environment

Chapter 4

Arithmetic Operations to Provide Fast Symmetric–key Cryptosystems

4.1 AES Algorithm Applied Basis Conversion

In encryption and decryption procedures of AES algorithm, a plaintext is split into 128–bit blocks. Every block is described as the following 4×4 matrix, whose each element is dealt with as an element in the $\mathbb{F}_{2^8}$.

$$\begin{bmatrix} H_{0,0} & H_{0,0} & H_{0,2} & H_{0,3} \\ H_{1,0} & H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,0} & H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,0} & H_{3,1} & H_{3,2} & H_{3,3} \end{bmatrix} \quad (H_{j,l} \in \mathbb{F}_{2^8}). \tag{4.1}$$

The original AES algorithm [7] represents an element in $\mathbb{F}_{2^8}$ with the polynomial basis $\{1, \alpha, \alpha^2, \ldots, \alpha^6, \alpha^7\}$, where α is a zero of the irreducible polynomial $f_0(t) = t^8 + t^4 + t^2 + t + 1$ over $\mathbb{F}_2$. Let H denote an element in the $\mathbb{F}_{2^8}$, then this chapter arbitrarily represents H as **Table** 4.1.

This section introduces the encryption and decryption procedures of AES algorithm applied *basis conversion* from the $\mathbb{F}_{2^8}$ to its isomorphic towering field. Although the thesis fundamentally follows the approach in [26], some parts of the procedures are improved. In what follows, the improved parts are clarified.

Table 4.1: Representation styles of an element in the $\mathbb{F}_{2^8}$

Style	Representation $(h_j \in \{0,1\})$
basis in $\mathbb{F}_{2^8}$	$h_0 + h_1\alpha + h_2\alpha^2 + \cdots + h_6\alpha^6 + h_7\alpha^7$
vector	$[\,h_0 \;\; h_1 \;\; h_2 \;\; \cdots \;\; h_6 \;\; h_7\,]$
integer	'h' $(h = h_0 + h_1 2 + h_2 2^2 + \cdots + h_6 2^6 + h_7 2^7)$

4.1.1 Encryption Procedure Applied Basis Conversion

0–th round: Only AddRoundKey is carried out. Then, each element of the 4×4 matrix is processed as

$$C_{0,j,l} = \Big(H_{j,l} + K_{0,j,l}\Big)\mathbf{B} \quad (0 \le j,\, l < 4), \tag{4.2}$$

where $K_{0,j,l}$ is the j–th row and l–th column element of the 0–th round key (4×4 matrix), and $\mathbf{B}$ denotes a basis conversion matrix from the $\mathbb{F}_{2^8}$ to its isomorphic towering field. $C_{0,j,l}$ in Eq. (4.2) becomes an element in the isomorphic towering field. From there to the last round, each element of the 4×4 matrix is dealt with as an element in the isomorphic towering field.

From 1–st to 2–nd last round: First, SubBytes is carried out. Then, each element of the 4×4 matrix is processed as

$$G_{r,j,l} = \Big(C_{r-1,j,l}\Big)^{-1}\bar{\mathbf{B}}\mathbf{A}\mathbf{B} \quad (0 \leq j,\, l < 4), \tag{4.3}$$

where r is the ordinal number of the round, $\bar{\mathbf{B}}$ denotes the inverse matrix of $\mathbf{B}$, and $\mathbf{A}$ denotes the Affine transformation matrix [23]. $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ in Eq. (4.3) can be preliminarily calculated. Additionally, $(C_{r-1,j,l})^{-1}$ in Eq. (4.3) is the inverse element in the isomorphic towering field, and it should be efficiently calculated.

Next, ShiftRows, MixColumns, and AddRoundKey are carried out. In order to perform these steps faster, this thesis applies a new approach different from that in [26]. Actually, each element of the 4×4 matrix can be processed as Eq. (4.4a) or (4.4b).

$$\begin{aligned} C_{r,j,l} = &\left(\Big(G_{r,\langle j+1\rangle,\langle l+j\rangle} + G_{r,\langle j+2\rangle,\langle l+j\rangle}\Big) + \Big(G_{r,\langle j+3\rangle,\langle l+j\rangle} + (K_{r,j,l} + L)\mathbf{B}\Big)\right) \\ &+ \left((\text{`2'}\mathbf{B})\Big(G_{r,j,\langle l+j\rangle} + G_{r,\langle j+1\rangle,\langle l+j\rangle}\Big)\right) \quad (0 \leq j,\, l < 4), \end{aligned} \tag{4.4a}$$

$$\begin{aligned} C_{r,j,l} = &\left(\Big(G_{r,j,\langle l+j\rangle} + G_{r,\langle j+2\rangle,\langle l+j\rangle}\Big) + \Big(G_{r,\langle j+3\rangle,\langle l+j\rangle} + (K_{r,j,l} + L)\mathbf{B}\Big)\right) \\ &+ \left((\text{`3'}\mathbf{B})\Big(G_{r,j,\langle l+j\rangle} + G_{r,\langle j+1\rangle,\langle l+j\rangle}\Big)\right) \quad (0 \leq j,\, l < 4), \end{aligned} \tag{4.4b}$$

where $\langle j\rangle$ means "$j \mod 4$", $K_{r,j,l}$ is the j–th row and l–th column element of the r–th round key (4×4 matrix), and L denotes the Affine transformation vector [23]. In Eq. (4.4), '02'$\mathbf{B}$ and '03'$\mathbf{B}$ can be preliminarily calculated, and $(K_{r,j,l} + L)\mathbf{B}$ can be calculated when the round key is generated.

Last round: First, SubBytes is carried out. Then, each element of the 4×4 matrix is processed as Eq. (4.3).

Next, ShiftRows and AddRoundKey are carried out. Then, each element of the 4×4 matrix is processed as

$$\tilde{C}_{j,l} = G_{r,j,\langle l+j\rangle}\bar{\mathbf{B}} + (K_{r,j,l} + L) \quad (0 \leq j,\, l < 4). \tag{4.5}$$

$K_{r,j,l} + L$ in Eq. (4.5) can be calculated when the round key is generated. $\tilde{C}_{j,l}$ in Eq. (4.5) is dealt with in the same way as $H_{j,l}$, namely as an element in the $\mathbb{F}_{2^8}$. The 4×4 matrix which consists of $\tilde{C}_{j,l}$ in Eq. (4.5) forms a 128–bit block of the cipher text. This 128–bit block is the same of that not applied basis conversion, namely that in the original AES algorithm.

4.1.2 Decryption Procedure Applied Basis Conversion

0–th round: Only AddRoundKey is carried out. Then, each element of the 4×4 matrix is processed as

$$C_{r-1,j,l} = \Big(\tilde{C}_{j,l} + (K_{r,j,l} + L)\Big)\mathbf{B} \quad (0 \leq j,\, l < 4). \tag{4.6}$$

$K_{r,j,l} + L$ in Eq. (4.6) can be calculated when the round key is generated. $C_{r-1,j,l}$ in Eq. (4.6) is an element in the isomorphic towering field. From there to the last round, each element of the 4×4 matrix is dealt with as an element in the isomorphic towering field.

From 1–st to 2–nd last round: First, InvShiftRows and InvSubBytes are carried out. Then, each element of the 4×4 matrix is processed as

$$G_{r,j,l} = \left(C_{r,j,\langle l-j \rangle}\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}\right)^{-1} \quad (0 \leq j,\, l < 4), \tag{4.7}$$

where $\bar{\mathbf{A}}$ denotes the inverse Affine transformation matrix [23]. $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$ in Eq. (4.7) is preliminarily calculated. Additionally, $(C_{r,j,l}\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B})^{-1}$ in Eq. (4.7) is the inverse element in the isomorphic towering field, and it should be efficiently calculated.

Next, AddRoundKey and InvMixColumns are carried out. In order to perform these steps faster, this thesis applies a new approach different from that in [26]. For example, each element of the 4×4 matrix can be processed as

$$C_{r-1,j,l} = \Bigg((\text{`14'}\mathbf{B})G_{r,j,l} + (\text{`11'}\mathbf{B})G_{r,\langle j+1 \rangle,l} \Bigg) + \Bigg((\text{`13'})\mathbf{B}G_{r,\langle j+2 \rangle,l} + \Big( (\text{`9'}\mathbf{B})G_{r,\langle j+3 \rangle,l} + J_{r,j,l} \Big) \Bigg), \tag{4.8a}$$

$$J_{r,j,l} = (\text{`14'}K_{r,j,l} + \text{`11'}K_{r,\langle j+1 \rangle,l} + \text{`13'}K_{r,\langle j+2 \rangle,l} + \text{`9'}K_{r,\langle j+3 \rangle,l} + L)\mathbf{B}, \tag{4.8b}$$

where `14'$\mathbf{B}$, `11'$\mathbf{B}$, `13'$\mathbf{B}$, and `9'$\mathbf{B}$ can be preliminarily calculated, and $J_{r,j,l}$ can be calculated when the round key is generated.

Last round: First, InvShiftRows and InvSubBytes are carried out. Then, each element of the 4×4 matrix is processed as Eq. (4.7).

Next, AddRoundKey is carried out. Then, each element of the 4×4 matrix is processed as

$$H_{j,l} = G_{1,j,l}\bar{\mathbf{B}} + K_{0,j,l} \quad (0 \leq j,\, l < 4). \tag{4.9}$$

4.2 Arithmetic Operations in Towering Field $\mathbb{F}_{(2^4)^2}$

In the AES algorithm applied basis conversion from the $\mathbb{F}_{2^8}$ to $\mathbb{F}_{(2^4)^2}$, inversions and multiplications in $\mathbb{F}_{(2^4)^2}$ are required as described in Eqs. (4.3), (4.4), (4.7) and (4.8). Thus, this section introduces how to prepare $\mathbb{F}_{(2^4)^2}$ and its subfield $\mathbb{F}_{2^4}$, and efficient arithmetic operations in these extension fields.

In the case of $\mathbb{F}_{(2^4)^2}$, first construct $\mathbb{F}_{2^4}$, then 2–nd tower over the $\mathbb{F}_{2^4}$. Most of researchers [26, 27, 28, 29, 30, 31, 32] use normal bases and polynomial bases to prepare extension fields and towering fields. This thesis also adopts normal bases to achieve 2–nd towering over $\mathbb{F}_{2^4}$. On the other hand, this thesis adopts an innovative basis to construct $\mathbb{F}_{2^4}$. This section introduces the detail of the adopted bases and the arithmetic operations.

4.2.1 Quartic Extension Field $\mathbb{F}_{2^4}$

Irreducible polynomial and an innovative basis: There exist 3 kinds of quartic irreducible polynomials over $\mathbb{F}_2$ as follows.

$$f_1(t) = t^4+t+1, \quad f_2(t) = t^4+t^3+1, \quad f_3(t) = t^4+t^3+t^2+t+1. \tag{4.10}$$

Normal bases and polynomial bases in $\mathbb{F}_{2^4}$ can be distinguished from a zero of these polynomials. For a zero β of $f_1(t)$, the set $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$ does not form normal bases; however, $\{1, \beta, \beta^2, \beta^3\}$ forms a polynomial basis. Rudra et al. [26] and Joen et al. [32] have shown that the polynomial basis efficiently carries out arithmetic operations, especially inversion, in $\mathbb{F}_{2^4}$.

On the other hand, for a zero β of $f_3(t)$, the sets $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$ and $\{1, \beta, \beta^2, \beta^3\}$ respectively form a normal basis and a polynomial basis. The normal basis is especially called type–I optimal normal basis (ONB) [17], and it carries out arithmetic operations in $\mathbb{F}_{2^4}$ as efficiently as in Rudra et al.'s and Jeon et al.'s implementations. However, this thesis adopts an innovative basis instead of type–I ONB and the polynomial basis. The basis is the union $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}, 1\}$ of type–I ONB $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$ and $\{1\}$, and it can provide faster arithmetic operations than the type–I ONB and the polynomial basis. This thesis especially calls it *Redundantly Represented Basis* (RRB). In what follows, the properties of RRB is described.

β which is a zero of $f_3(t)$ has the following relations.

$$f_3(\beta) = \beta^4 + \beta^3 + \beta^2 + \beta + 1 = 0, \Leftrightarrow \ f_3(\beta) = \beta + \beta^2 + \beta^{2^2} + \beta^{2^3} + 1 = 0, \tag{4.11a}$$

$$\because \ (\beta + 1) f_3(\beta) + 1 = \beta^5 = 1. \tag{4.11b}$$

According to Eq. (4.11b), type–I ONB $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$ is described as follows.

$$\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\} = \{\beta, \beta^2, \beta^3, \beta^4\}. \tag{4.12}$$

Because β, β^2, β^{2^2}, β^{2^3} are conjugate zeros of $f_3(t)$, 4 kinds of polynomial bases are considered according to Eq. (4.11b) as follows.

$$\{1, \beta, \beta^2, \beta^3\} = \{1, \beta, \beta^2, \beta^3 \quad\}, \tag{4.13a}$$

$$\{1, \beta^2, (\beta^2)^2, (\beta^2)^3\} = \{1, \beta, \beta^2, \quad \beta^4\}, \tag{4.13b}$$

$$\{1, \beta^{2^2}, (\beta^{2^2})^2, (\beta^{2^2})^3\} = \{1, \quad \beta^2, \beta^3, \beta^4\}, \tag{4.13c}$$

$$\{1, \beta^{2^3}, (\beta^{2^3})^2, (\beta^{2^3})^3\} = \{1, \beta, \quad \beta^3, \beta^4\}. \tag{4.13d}$$

According to Eqs. (4.12), (4.13), a basis is obtained by removing some one element from the set $\{1, \beta, \beta^2, \beta^3, \beta^4\}$. On the other hand, RRB $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}, 1\} = \{1, \beta, \beta^2, \beta^3, \beta^4\}$ uses all. Thus, the conversion from RRB to the bases in Eqs. (4.12), (4.13) can be easily achieved from Eq. (4.11a).

Let D denote an element in $\mathbb{F}_{2^4}$, then D is represented with RRB as Eq. (4.14a).

$$D = d_0\beta + d_1\beta^2 + d_2\beta^{2^2} + d_3\beta^{2^3} + d_4 \quad (d_j \in \mathbb{F}_2). \tag{4.14a}$$

$$= (d_0 + d_4)\beta + (d_1 + d_4)\beta^2 + (d_2 + d_4)\beta^{2^2} + (d_3 + d_4)\beta^{2^3} \tag{4.14b}$$

$$= (d_4 + d_2) + (d_0 + d_2)\beta + (d_1 + d_2)\beta^2 + (d_3 + d_2)\beta^3. \tag{4.14c}$$

As described above, according to Eq. (4.11a), D represented with RRB can be easily converted to that represented with type–I ONB and the polynomial bases in Eqs. (4.12), (4.13a) as Eqs. (4.14b), (4.14c).

In principle, RRB in $\mathbb{F}_{2^4}$ can not uniquely represent an element in $\mathbb{F}_{2^4}$. For example, $D = \beta + \beta^2$ is also described as $D = \beta^{2^2} + \beta^{2^3} + 1$ according to Eq. (4.11a). However, D is uniquely represented when the Hamming weight of D is restricted to be equal to or less than 2. On the other hand, the Hamming weight of D can be easily reduced to be equal to or less than 2 according to Eq. (4.11a) when it is more than 2.

Arithmetic Operations: Let E denote an element in $\mathbb{F}_{2^4}$, then E is represented with RRB as follows.

$$E = e_0\beta + e_1\beta^2 + e_2\beta^{2^2} + e_3\beta^{2^3} + e_4 \quad (e_j \in \mathbb{F}_2). \tag{4.15}$$

A multiplication $M = D \times E$ is given as follows. Note that it is derived from type–I *Cyclic Vector Multiplication Algorithm* (CVMA) [16] and Eq. (4.14).

$$\begin{aligned} M &= m_0\beta + m_1\beta^2 + m_2\beta^{2^2} + m_3\beta^{2^3} + m_4 \quad (m_j \in \mathbb{F}_2) \\ &= (d_4e_0 + d_2e_1 + d_1e_2 + d_3e_3 + d_0e_4)\beta + (d_0e_0 + d_4e_1 + d_3e_2 + d_2e_3 + d_1e_4)\beta^2 \\ &\quad + (d_3e_0 + d_1e_1 + d_4e_2 + d_0e_3 + d_2e_4)\beta^{2^2} + (d_1e_0 + d_0e_1 + d_2e_2 + d_4e_3 + d_3e_4)\beta^{2^3} \\ &\quad + (d_2e_0 + d_3e_1 + d_0e_2 + d_1e_3 + d_4e_4) \qquad (4.16a) \\ &= (a_{1,2}b_{1,2} + a_{0,4}b_{0,4})\beta + (a_{2,3}b_{2,3} + a_{1,4}b_{1,4})\beta^2 + (a_{0,3}b_{0,3} + a_{2,4}b_{2,4})\beta^{2^2} \\ &\quad + (a_{0,1}b_{0,1} + a_{3,4}b_{3,4})\beta^{2^3} + (a_{0,2}b_{0,2} + a_{1,3}b_{1,3}), \qquad (4.16b) \\ a_{j,l} &= d_j + d_l, \quad b_{j,l} = e_j + e_l \quad (0 \le j < l \le 4). \qquad (4.16c) \end{aligned}$$

The critical path delay of the multiplication circuit given by Eq. (4.16b) is $1T_{\mathrm{AND}} + 2T_{\mathrm{XOR}}$. On the other hand, that given by Eq. (4.16a) is $1T_{\mathrm{AND}} + 3T_{\mathrm{XOR}}$. Thus, in principle, a multiplication in $\mathbb{F}_{2^4}$ should be calculated as Eq. (4.16b) (**Fig**. 4.2).

From here on, suppose that E is a non–zero constant element in $\mathbb{F}_{2^4}$, then this subsection considers a multiplication by the constant element E. When the Hamming weight of E is restricted to be equal to or less than 2, namely 1 or 2, E can be classified as **Table** 4.2. According to Eq. (4.16a), a multiplication $N = D \times E$ can be carried out with theoretically no delay when E belongs to the class (I) of **Table** 4.2, that is, the Hamming weight of E is 1. On the other hand, it can be calculated with $1T_{\mathrm{XOR}}$ when E belongs to the class (II) of **Table** 4.2, that is, the Hamming weight of E is 2. For example, multiplications $N_0 = D \times (1,0,0,0,0)$ and $N_1 = D \times (1,1,0,0,0)$ are respectively given from Eq. (4.16a) as follows.

$$N_0 = d_4\beta + d_0\beta^2 + d_3\beta^{2^2} + d_1\beta^{2^3} + d_2, \tag{4.17a}$$

$$N_1 = (d_2 + d_4)\beta + (d_0 + d_4)\beta^2 + (d_3 + d_1)\beta^{2^2} + (d_0 + d_1)\beta^{2^3} + (d_2 + d_3). \tag{4.17b}$$

A squaring $S = D^2$ can be carried out with theoretically no delay as follows.

$$S = d_3\beta + d_0\beta^2 + d_1\beta^{2^2} + d_2\beta^{2^3} + d_4. \tag{4.18}$$

From here on, suppose that D is a non–zero element in $\mathbb{F}_{2^4}$, then an inversion $I = D^{-1}$ is given as follows (**Fig**. 4.3). See **Sec**. 4.4 about how to derive it.

$$\begin{aligned} I &= i_0\beta + i_1\beta^2 + i_2\beta^{2^2} + i_3\beta^{2^3} + i_4 \quad (i_j \in \mathbb{F}_2) \\ &= (a_{2,4} + a_{0,4}a_{1,4}a_{1,3})\beta + (a_{3,4} + a_{1,4}a_{2,4}a_{0,2})\beta^2 + (a_{0,4} + a_{2,4}a_{3,4}a_{1,3})\beta^{2^2} \\ &\quad + (a_{1,4} + a_{3,4}a_{0,4}a_{0,2})\beta^{2^3} + (a_{0,4}a_{2,4}\overline{a_{1,3}} + a_{1,4}a_{3,4}\overline{a_{0,2}}), \qquad (4.19a) \\ a_{j,l} &= (d_j + d_l) \quad (0 \le j < l \le 4), \qquad (4.19b) \end{aligned}$$

where $\overline{d}$ $(d \in \mathbb{F}_2)$ means "NOT d".

The critical path delay of each arithmetic operation circuit with RRB is given as **Table** 4.3. As shown in **Table** 4.3, compared to Rudra et al.'s [26] and Jeon et al.'s [32] implementations, RRB can reduce each critical path delay of a multiplication circuit and a squaring circuit in $\mathbb{F}_{2^4}$ by $1T_{\mathrm{XOR}}$.

Table 4.2: Classification of non–zero elements in $\mathbb{F}_{2^4}$

Class	(I)	(II)
Element in $\mathbb{F}_{2^4}^*$ †	$(1,0,0,0,0)$	$(1,1,0,0,0)$, $(1,0,1,0,0)$
	$(0,1,0,0,0)$	$(0,1,1,0,0)$, $(0,1,0,1,0)$
	$(0,0,1,0,0)$	$(0,0,1,1,0)$, $(0,0,1,0,1)$
	$(0,0,0,1,0)$	$(0,0,0,1,1)$, $(1,0,0,1,0)$
	$(0,0,0,0,1)$	$(1,0,0,0,1)$, $(0,1,0,0,1)$
Hamming weight	1	2

† $(e_0, e_1, e_2, e_3, e_4)$ denotes an element E in Eq. (4.15).

Table 4.3: The critical path delay of each arithmetic operation circuit in $\mathbb{F}_{2^4}$

Implementation	Multiplication	Squaring	Inversion	Multiplication by the class (I) element	Multiplication by the class (II) element
Rudra al.'s [26]	$(1,3)^\dagger$	$(0,1)^\dagger$	$(2,2)^\dagger$	—	—
Jeon al.'s [32]					
With RRB	$(1,\mathbf{2})^\dagger$	$(0,\mathbf{0})^\dagger$		$(0,0)^\dagger$	$(0,1)^\dagger$

† (j,l) means $jT_{\mathrm{AND}} + lT_{\mathrm{XOR}}$.

‡ The delay when $T_{\mathrm{AND}} \geq T_{\mathrm{XOR}}$ is shown. That when $T_{\mathrm{AND}} \leq T_{\mathrm{XOR}}$ is given as $(1,3)$.

4.2.2 2–nd Towering Field $\mathbb{F}_{(2^4)^2}$

Irreducible Polynomial and Normal Basis: In the same way as **Sec**. 4.2.1, this subsection first considers the setting of irreducible polynomial. Let a quadr- atic polynomial over $\mathbb{F}_{2^4}$ be described as follows.

$$g(t) = t^2 + \mu t + \nu \quad (\mu, \nu \in (\mathbb{F}_{2^4} - \{0\})). \tag{4.20}$$

In order that $g(t)$ is irreducible over $\mathbb{F}_{2^4}$, $g(t)$ needs to satisfy that $\mu^2/\nu \notin \mathbb{F}_{2^2}$. Suppose that γ is a zero of $g(t)$, then the sets $\{\gamma, \gamma^{16}\}$ and $\{1, \gamma\}$ respectively form a normal basis and a polynomial basis in $\mathbb{F}_{(2^4)^2}$. Among these bases, this subsection focuses on the normal basis only.

Arithmetic Operations: Let C denote an element in $\mathbb{F}_{(2^4)^2}$, $\mathbf{B}$ denote a basis conversion matrix from the $\mathbb{F}_{2^8}$ to its isomorphic towering field $\mathbb{F}_{(2^4)^2}$, and 'j' $(0 \leq j < 256)$ denote an element in $\mathbb{F}_{2^8}$ described by the integer style of **Table** 4.1. Then, C and 'j'$\mathbf{B}$ is represented with the normal basis $\{\gamma, \gamma^{16}\}$ as follows.

$$C = D\gamma + E\gamma^{16} \ (D, E \in \mathbb{F}_{2^4}), \qquad \text{'}j\text{'}\mathbf{B} = Q_j\gamma + R_j\gamma^{16} \ (Q_j, R_j \in \mathbb{F}_{2^4}), \tag{4.21}$$

where D, E, Q_j, and R_j are represented with RRB in $\mathbb{F}_{2^4}$. Then, a multiplication $W = C \times$ 'j'$\mathbf{B}$ is given as follows. See **Sec**. 4.4 about how to derive it.

$$W = Y\gamma + Z\gamma^{16} \ (Y, Z \in \mathbb{F}_{2^4}) = \{D\delta_j + E\epsilon_j\}\gamma + \{D\epsilon_j + E\eta_j\}\gamma^{16}, \tag{4.22a}$$

$$\delta_j = Q_j(\mu + \frac{\nu}{\mu}) + R_j \cdot \frac{\nu}{\mu}, \quad \epsilon_j = (Q_j + R_j) \cdot \frac{\nu}{\mu}, \quad \eta_j = Q_j \cdot \frac{\nu}{\mu} + R_j(\mu + \frac{\nu}{\mu}), \tag{4.22b}$$

where δ_j, ϵ_j, and η_j can be preliminarily calculated. According to **Tables** 4.2, 4.3, the critical path delay of the multiplication circuit given by Eq. (4.22a) is at most $\mathbf{2}T_{\mathrm{XOR}}$ even if δ_j, ϵ_j, and η_j are assigned with arbitrary elements.

From here on, suppose that C is a non–zero element in $\mathbb{F}_{(2^4)^2}$, then with *Itoh–Tsujii inversion Algorithm* (ITA) [35], an inversion $X = C^{-1} = (C \cdot C^{16})^{-1} C^{16}$ is given as follows (**Fig**. 4.6(a)). Note that it is derived by generalizing the approach in [30], in detail, by appending a μ^2–multiplication in $\mathbb{F}_{2^4}$.

$$X = Y\gamma + Z\gamma^{16} \ (Y, Z \in \mathbb{F}_{2^4}) \ = \{E\gamma + D\gamma^{16}\}/\{DE\mu^2 + (D+E)^2\nu\}, \tag{4.23}$$

where each multiplication by μ^2 and ν can be carried out with theoretically no delay according to **Table** 4.3 when the following condition is satisfied.

Condition 6 *Both μ^2 and ν belong to the class (I) of* **Table** *4.1.*

Thus, this thesis considers that both μ^2 and ν are assigned with the class (I) elements. Then, there exist **20** irreducible polynomials over $\mathbb{F}_{2^4}$ which satisfies **Cond**. 6, and the critical path delay of the inversion circuit in $\mathbb{F}_{(2^4)^2}$ is given as $\mathbf{4}T_{\mathrm{AND}} + \mathbf{7}T_{\mathrm{XOR}}$ from **Table** 4.3 and **Fig**. 4.6(a). As shown in **Table** 4.4, the circuit of this work can carry out an inversion in the towering field isomorphic to the $\mathbb{F}_{2^8}$ faster than those of the others. On the other hand, the circuit size is given as **Table** 4.5 (before downsizing). As shown in **Table** 4.5, the inversion circuit in $\mathbb{F}_{(2^4)^2}$ of this work (before downsizing) uses more XOR gates than that of Jeon et al. Thus, the next subsection considers how to downsize the inversion circuit in $\mathbb{F}_{(2^4)^2}$.

4.2.3 Theoretical Downsizing the Inversion Circuit in $\mathbb{F}_{(2^4)^2}$

Focus on **Fig**. 4.6(a), then it is seeable that the wire (i) directly connects to the multiplication circuit (I) and (II), the wire (ii) connects through the μ^2–multiplication circuit to the multiplication circuit (I) and directly connects to the multiplication circuit (III), and the wire (iii) directly connects to the multiplication circuit (II) and (III). Thus, for the inversion circuit in $\mathbb{F}_{(2^4)^2}$, a part, namely 1–st part shown in **Fig**. 4.2(a), of each multiplication circuit in $\mathbb{F}_{2^4}$ can be shared with each other as **Fig**. 4.6(b). Then, the circuit size can be reduced by **30**XOR gates according to **Table** 4.5. As a result, the inversion circuit in $\mathbb{F}_{(2^4)^2}$ of this work (after downsizing) uses less logic gates than that of Jeon et al.

Table 4.4: The critical path delay of an inversion circuit in towering field

Towering field	Implementation	Critical path delay
$\mathbb{F}_{((2^2)^2)^2}$	Satoh and Morioka et al.'s [27, 28]	$4T_{\mathrm{AND}} + 17T_{\mathrm{XOR}}$
	Mentens's et al. [29]	
	Canright's [30]	$4T_{\mathrm{AND}} + 15T_{\mathrm{XOR}}$
	Nogami et al.'s [31]	$4T_{\mathrm{AND}} + 14T_{\mathrm{XOR}}$
$\mathbb{F}_{(2^4)^2}$	Rudra et al.'s [26]	$4T_{\mathrm{AND}} + 10T_{\mathrm{XOR}}$
	Jeon et al.'s [32]	
	This work	$4T_{\mathrm{AND}} + \mathbf{7}T_{\mathrm{XOR}}$

Table 4.5: The number of logic gates for an inversion circuit in $\mathbb{F}_{(2^4)^2}$

Implementation	Before downsizing	After downsizing
Rudra et al.'s [26]	60AND + 72XOR	
Jeon et al.'s [32]	58AND + 67XOR + 2OR	
This work	42AND + 98XOR + 2XNOR	**42**AND + **68**XOR + **2**XNOR

4.3 Basis Conversion between $\mathbb{F}_{2^8}$ and $\mathbb{F}_{(2^4)^2}$

This section evaluates the calculation efficiencies given by basis conversion matrices for Eq. (4.3) (namely, SubBytes), Eq. (4.4) (namely, ShiftRows, MixColumns, and AddRoundKey), Eq. (4.7) (namely, InvShiftRows and InvSubBytes), and Eq. (4.8) (namely, InvMixColumns and AddRoundKey).

4.3.1 Calculation Efficiency of Eqs. (4.3) and (4.7)

This subsection considers each multiplication by $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ and $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$ in Eqs. (4.3) and (4.7), where $\mathbf{B}$, $\bar{\mathbf{B}}$, $\mathbf{A}$, and $\bar{\mathbf{A}}$ respectively denote a basis conversion matrix from the $\mathbb{F}_{2^8}$ to its isomorphic towering field $\mathbb{F}_{(2^4)^2}$, its inverse matrix, Affine transformation matrix, and the inverse Affine transformation matrix. In the case of adopting RRB described in **Sec**. 4.2.1, both conversion matrices $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ and $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$ from $\mathbb{F}_{(2^4)^2}$ over the $\mathbb{F}_{2^4}$ constructed by RRB to the same $\mathbb{F}_{(2^4)^2}$ are required. Actually, these conversion matrices are given by a basis conversion matrix $\mathbf{B}$ from the $\mathbb{F}_{2^8}$ to $\mathbb{F}_{(2^4)^2}$ over the $\mathbb{F}_{2^4}$ constructed by type–I ONB of Eq. (4.12) or the polynomial bases of Eq. (4.13) according to Eq. (4.14).

In order to show an example, suppose an extension field $\mathbb{F}_{2^4}$ constructed by type–I ONB $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$, a field $\mathbb{F}_{(2^4)^2}$ which 2–nd towers over the $\mathbb{F}_{2^4}$ with the normal basis $\{\gamma, \gamma^{16}\}$, and a basis conversion matrix $\mathbf{B}$ from the $\mathbb{F}_{2^8}$ to the $\mathbb{F}_{(2^4)^2}$. Then, $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ in Eq. (4.3) is represented as the left–hand equation in Eq. (4.24), and an example of the $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ is given as the right–hand equation in Eq. (4.24).

$$\bar{\mathbf{B}}\mathbf{A}\mathbf{B} = \left[\begin{array}{cccccc} u_{0,0} & u_{0,1} & u_{0,2} & \cdots & u_{0,6} & u_{0,7} \\ \vdots & & \ddots & & \vdots & \\ u_{3,0} & u_{3,1} & u_{3,2} & \cdots & u_{3,6} & u_{3,7} \\ \hdashline v_{0,0} & v_{0,1} & v_{0,2} & \cdots & v_{0,6} & v_{0,7} \\ \vdots & & \ddots & & \vdots & \\ v_{3,0} & v_{3,1} & v_{3,2} & \cdots & v_{3,6} & v_{3,7} \end{array}\right], \quad \bar{\mathbf{B}}\mathbf{A}\mathbf{B} = \left[\begin{array}{c:c:c:c:c:c:c:c} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]. \tag{4.24}$$

Let $C_{r-1,j,l}$ in Eq. (4.3) be corresponding to a non–zero element $C = D\gamma + E\gamma^{16}$ $(D, E \in \mathbb{F}_{2^4})$ which is the input of the inversion circuit of **Fig**. 4.6(b), and let $(C_{r-1,j,l})^{-1}$ in Eq. (4.3) be corresponding to $X = C^{-1} = Y\gamma + Z\gamma^{16}$ $(Y, Z \in \mathbb{F}_{2^4})$ which is the output of the inversion circuit of **Fig**. 4.6(b). In the case that the elements Y and Z in $\mathbb{F}_{2^4}$ are represented with RRB as shown in

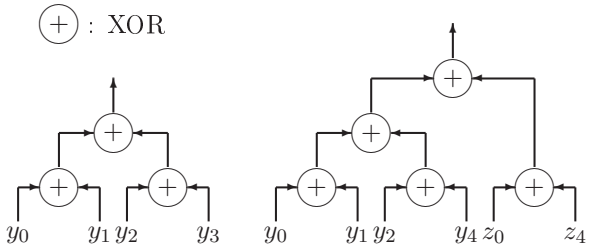
Fig. 4.6(b), converting the representations from RRB to type–I ONB is easy from Eq. (4.11a) as

$$\begin{aligned} Y &= y_0\beta + y_1\beta^2 + y_2\beta^{2^2} + y_3\beta^{2^3} + y_4 \\ &= (y_0+y_4)\beta + (y_1+y_4)\beta^2 + (y_2+y_4)\beta^{2^2} + (y_3+y_4)\beta^{2^3}, \end{aligned} \tag{4.25a}$$

$$\begin{aligned} Z &= z_0\beta + z_1\beta^2 + z_2\beta^{2^2} + z_3\beta^{2^3} + z_4 \\ &= (z_0+z_4)\beta + (z_1+z_4)\beta^2 + (z_2+z_4)\beta^{2^2} + (z_3+z_4)\beta^{2^3}. \end{aligned} \tag{4.25b}$$

Then, a multiplication by $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ is given as Eq. (4.26), and the circuits of Eq. (4.26) is drawn as **Fig**. 4.1.

$$X\bar{\mathbf{B}}\mathbf{A}\mathbf{B} = \begin{bmatrix} y_0+y_4 \\ y_0+y_1 \\ (y_0+y_4)+(z_0+z_4) \\ (y_0+y_1)+(y_2+y_4) \\ (y_0+y_1)+(z_0+z_4) \\ (y_0+y_1)+(y_2+y_3) \\ \Big((y_0+y_1)+(y_2+y_4)\Big)+(z_0+z_4) \\ (y_0+y_1)+(z_0+z_1) \end{bmatrix}^{\mathcal{T}}. \tag{4.26}$$



(a) 6–th row (b) 7–th row

Figure 4.1: Example images of circuits for Eq. (4.26)

According to the above consideration, a conversion matrix $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ from the $\mathbb{F}_{(2^4)^2}$ over the $\mathbb{F}_{2^4}$ constructed by RRB to the same $\mathbb{F}_{(2^4)^2}$ over the $\mathbb{F}_{2^4}$ constructed by RRB (actually, type–I ONB) is obtained.

A row of $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ can be represented with the following 2 vectors from Eq. (4.24).

$$U_j = [\,u_{j,0}\ u_{j,1}\ u_{j,2}\ u_{j,3}\,]^{\mathcal{T}}, \qquad V_j = [\,v_{j,0}\ v_{j,1}\ v_{j,2}\ v_{j,3}\,]^{\mathcal{T}}. \tag{4.27}$$

Let $\mathrm{Hw}(U)$ denote the number of "1" in the vector U, namely the Hamming weight of U. According to Eq. (4.26) and **Fig**. 4.1, the critical path delay of the circuit multiplying $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ is equal to or less than $2T_{\mathrm{XOR}}$ when all vectors U_j and V_j $(0 \le j < 8)$ satisfy that $\mathrm{Hw}(U_j){:}\mathrm{Hw}(V_j) \neq 3{:}1$, $1{:}3$, and $\mathrm{Hw}(U_j)+\mathrm{Hw}(V_j) \le 4$; otherwise, it is $3T_{\mathrm{XOR}}$. The probability when all column vectors of $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$ satisfy that $\mathrm{Hw}(U_j){:}\mathrm{Hw}(V_j) \neq 3{:}1$, $1{:}3$, and $\mathrm{Hw}(U_j)+\mathrm{Hw}(V_j) \le 4$ is given as

$$({}_8\mathrm{C}_0 + {}_8\mathrm{C}_1 + {}_8\mathrm{C}_2 + {}_8\mathrm{C}_3 + {}_4\mathrm{C}_4 \cdot {}_4\mathrm{C}_0 + {}_4\mathrm{C}_2 \cdot {}_4\mathrm{C}_2 + {}_4\mathrm{C}_0 \cdot {}_4\mathrm{C}_4)^8/2^{8\times 8} \approx 0.47\%. \tag{4.28}$$

Note that the above probability is not strictly accurate because a basis conversion matrix must be a regular matrix.

On the other hand , the above consideration of a multiplication by $\bar{\mathbf{B}}\mathbf{AB}$ in Eq. (4.3) is also available for a multiplication by $\bar{\mathbf{B}}\bar{\mathbf{A}}\bar{\mathbf{B}}$ in Eq. (4.7).

4.3.2 Calculation Efficiency of Eqs. (4.4) and (4.8)

The calculation circuit of Eq. (4.4a) is shown in **Fig**. 4.4. Naturally, the calculation circuit of Eq. (4.4b) can be drawn in the same way as **Fig**. 4.4. According to **Fig**. 4.4, the calculation efficiency of Eq. (4.4) depends on the element '2'$\mathbf{B}$ or '3'$\mathbf{B}$ in $\mathbb{F}_{(2^4)^2}$. In more detail, when a multiplication by either '2'$\mathbf{B}$ or '3'$\mathbf{B}$ can be carried out in $1T_{\rm XOR}$, the critical path delay of the calculation circuit of Eq. (4.4) is $3T_{\rm XOR}$; otherwise, it is $4T_{\rm XOR}$ since each multiplication by '2'$\mathbf{B}$ and '3'$\mathbf{B}$ needs at most $2T_{\rm XOR}$ according to **Sec**. 4.2.2.

On the other hand, the calculation efficiency of Eq. (4.8) depends on the elements '14'$\mathbf{B}$, '11'$\mathbf{B}$, '13'$\mathbf{B}$, and '9'$\mathbf{B}$ in $\mathbb{F}_{(2^4)^2}$. This section proposes how to find the $\mathbf{B}$ such that the critical path delay of the calculation circuit of Eq. (4.8) is $4T_{\rm XOR}$. In order to achieve the above proposal, according to Eq. (4.22a), both an element among δ_{14}, δ_{11}, δ_{13}, δ_9, ϵ_{14}, ϵ_{11}, ϵ_{13} and ϵ_9 of Eq. (4.22b), and an element among ϵ_{14}, ϵ_{11}, ϵ_{13}, ϵ_9, η_{14}, η_{11}, η_{13} and η_9 of Eq. (4.22b) must be a zero element or the class (I) element of **Table** 4.2. For example, when ϵ_9 is a zero element or the class (I) element, the calculation of Eq. (4.8) can be carried out as **Fig**. 4.5, where $D_{j,l}$ and $E_{j,l}$ denote elements in $\mathbb{F}_{2^4}$ which satisfy that $G_{r,j,l} = D_{j,l}\gamma + E_{j,l}\gamma^{16}$, $Y_{j,l}$ and $Z_{j,l}$ denote elements in $\mathbb{F}_{2^4}$ which satisfy that $C_{r-1,j,l} = Y_{j,l}\gamma + Z_{j,l}\gamma^{16}$, and $U_{j,l}$ and $V_{j,l}$ denote elements in $\mathbb{F}_{2^4}$ which satisfy that $J_{r,j,l} = U_{j,l}\gamma + V_{j,l}\gamma^{16}$.

4.3.3 More Miscellaneously Mixed Basis (MMMB)

This thesis tries for the following goals.

Goal 1: Each multiplication by $\bar{\mathbf{B}}\mathbf{AB}$ and $\bar{\mathbf{B}}\bar{\mathbf{A}}\bar{\mathbf{B}}$ in Eqs. (4.3) and (4.7) is carried out in $2T_{\rm XOR}$.

Goal 2: The calculation of either Eq. (4.4a) or Eq. (4.4b) is carried out in $3T_{\rm XOR}$.

Goal 3: The calculation of Eq. (4.8) is carried out in $4T_{\rm XOR}$.

In order to achieve the above goals, it is important that an efficient basis conversion matrix $\mathbf{B}$ among a lot of prepared basis conversion matrices $\mathbf{B}$s is selectable. As an efficient technique to prepare more $\mathbf{B}$s, Nogami et al. have proposed *Mixed Bases* (MB) [31], which is applied to an implementation with $\mathbb{F}_{((2^2)^2)^2}$ in [31]. This subsection first considers to apply MB to an implementation with $\mathbb{F}_{(2^4)^2}$.

For a multiplication in $\mathbb{F}_{(2^4)^2}$ in Eq. (4.8), consider the following multiplication instead of Eq. (4.22a). See **Sec**. 4.4 about how to derive it.

$$W = Y + Z\gamma \ (Y, Z \in \mathbb{F}_{2^4}) \ = \{D\delta_j + E\epsilon_j\}\gamma + \{D\zeta_j + E\eta_j\}\gamma^{16}, \tag{4.29a}$$

$$\delta_j = (Q_j + R_j)\nu, \quad \epsilon_j = Q_j\nu + R_j(\mu^2 + \nu), \quad \zeta_j = Q_j\mu, \quad \eta_j = R_j\mu, \tag{4.29b}$$

where δ_j, ϵ_j, ζ_j, and η_j can be preliminarily calculated. In Eq. (4.29a), the normal basis $\{\gamma, \gamma^{16}\}$ is adopted for the input in the same way of Eq. (4.22a). On the other hand, the polynomial basis $\{1, \gamma\}$ is adopted for the output instead of the normal basis $\{\gamma, \gamma^{16}\}$. The critical path delay of this multiplication circuit in $\mathbb{F}_{(2^4)^2}$ is considered in the same way of that of Eq. (4.22a) (See **Sec**. 4.3.2). This multiplication circuit in $\mathbb{F}_{(2^4)^2}$ can provide conversion matrices $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$s from $\mathbb{F}_{(2^4)^2}$ 2–nd towering with not only the normal basis $\{\gamma, \gamma^{16}\}$ but also the polynomial basis

$\{1, \gamma\}$. However, the number of $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$s prepared by this technique is not enough to perfectly achieve the above goals. Thus, this section improves MB.

As described in **Sec**. 4.3.1, in the case that $\mathbb{F}_{2^4}$ is constructed by RRB, the basis conversion matrices $\mathbf{B}$s when $\mathbb{F}_{2^4}$ are constructed by type–I ONB of Eq. (4.12) and the polynomial bases of Eq. (4.13) are available. In more detail, a combination of two bases among the bases of Eqs. (4.12), (4.13) can be used to represent an element in $\mathbb{F}_{(2^4)^2}$. Let C denote an element in $\mathbb{F}_{(2^4)^2}$. For example, consider the combination of the normal basis $\{\beta, \beta^2, \beta^{2^2}, \beta^{2^3}\}$ and the polynomial basis $\{1, \beta, \beta^2, \beta^3\}$, then C is represented with the combination as

$$C = (d_0\beta + d_1\beta^2 + d_2\beta^{2^2} + d_3\beta^{2^3})\gamma + (e_0 + e_1\beta + e_2\beta^2 + e_3\beta^3)\gamma^{16} \ (d_j, e_j \in \mathbb{F}_2). \tag{4.30}$$

By only adopting the combinations as above, $20 \times 5 \times 5 \times 5 \times 5 = \mathbf{12,500}$ kinds of $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$s and $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$s can be respectively prepared. In this thesis, the technique to adopt different bases for the input and output of arithmetic operation in $\mathbb{F}_{(2^4)^2}$ and to use a combination of different bases in $\mathbb{F}_{2^4}$ is especially called More Miscellaneously Mixed Bases (MMMB).

Actually, by using MMMB, some $\bar{\mathbf{B}}\mathbf{A}\mathbf{B}$s and $\bar{\mathbf{B}}\bar{\mathbf{A}}\mathbf{B}$s to achieve **Goal 1**, and some $\mathbf{B}$s to achieve **Goal 3** can be found; however, no '2'$\mathbf{B}$s and '3'$\mathbf{B}$s to achieve **Goal 2** can be found. Thus, in this case, the calculation delay of Eq. (4.4) becomes $4T_{\mathrm{XOR}}$, not $3T_{\mathrm{XOR}}$. This issue will be kept as a future work.

By adopting RRB and MMMB as described in this chapter, the critical path delays of the encryption and decryption procedures of AES algorithm are shown as **Tables** 4.6, 4.7. Then, each round of the encryption procedure can be carried out in $\mathbf{4}T_{\mathrm{AND}} + \mathbf{13}T_{\mathrm{XOR}}$. On the other hand, each round of the decryption procedure also can be carried out in $\mathbf{4}T_{\mathrm{AND}} + \mathbf{13}T_{\mathrm{XOR}}$.

Table 4.6: The critical path delay of the encryption procedure of AES

Implementaion	SubBytes		MixColumns	AddRoundKey
	Inversion	Others		
Rudra et al.'s [26]	$(4, 10)^\dagger$	no data	$(0, 7)^\dagger$	$(0, 1)^\dagger$
Satoh and Morioka et al.'s [27, 28]	$(4, 17)^\dagger$			
Jeon et al.'s [32]	$(4, 10)^\dagger$	$(0, 11)^\dagger$		
This work	$(4, \mathbf{7})^\dagger$	$(0, \mathbf{2})^\dagger$	$(0, \mathbf{4})^\dagger$	

† (j, l) means $jT_{\mathrm{AND}} + lT_{\mathrm{XOR}}$.

Table 4.7: The critical path delay of the decryption procedure of AES

Implementaion	SubBytes		MixColumns	AddRoundKey
	Inversion	Others		
Jeon et al.'s [32]	$(4, 10)^\dagger$	$(0, 10)^\dagger$	$(0, 7)^\dagger$	$(0, 1)^\dagger$
This work	$(4, \mathbf{7})^\dagger$	$(0, \mathbf{2})^\dagger$	$(0, \mathbf{4})^\dagger$	

† (j, l) means $jT_{\mathrm{AND}} + lT_{\mathrm{XOR}}$.

4.4 Derivation of Eqs. (4.19), (4.22), and (4.29)

Eq. (4.19) is derived with ITA [35] as

$$
\begin{aligned}
I = D^{-1} &= (D\cdot D^4)^{-1}D^4 = (D\cdot D^{2^2})^{-1}D^{2^2} \\
&= \{(d_0\beta + d_1\beta^2 + d_2\beta^{2^2} + d_3\beta^{2^3} + d_4)(d_2\beta + d_3\beta^2 + d_0\beta^{2^2} + d_1\beta^{2^3} + d_4)\}^2 \\
&\quad \times (d_2\beta + d_3\beta^2 + d_0\beta^{2^2} + d_1\beta^{2^3} + d_4) \qquad (\because \text{Eq. (4.18)},\quad D\cdot D^{2^2} \in \mathbb{F}_{2^2}) \\
&= \{(d_1d_2 + d_2d_0 + d_0d_3 + d_3d_4 + d_4d_1)(\beta + \beta^{2^2}) \\
&\qquad + (d_0d_1 + d_1d_3 + d_3d_2 + d_2d_4 + d_4d_0)(\beta^2 + \beta^{2^3}) + (d_0 + d_1 + d_2 + d_3 + d_4)\} \\
&\quad \times (d_2\beta + d_3\beta^2 + d_0\beta^{2^2} + d_1\beta^{2^3} + d_4) \qquad (\because \text{Eqs. (4.16a), (4.18)}) \\
&= (a_{2,4} + a_{0,4}a_{1,4}a_{1,3})\beta + (a_{3,4} + a_{1,4}a_{2,4}a_{0,2})\beta^2 + (a_{0,4} + a_{2,4}a_{3,4}a_{1,3})\beta^{2^2} \\
&\quad + (a_{1,4} + a_{3,4}a_{0,4}a_{0,2})\beta^{2^3} + (a_{0,4}a_{2,4}\overline{a_{1,3}} + a_{1,4}a_{3,4}\overline{a_{0,2}}) \ (\because \text{Eqs. (4.16a)}), \qquad (4.31a) \\
a_{j,l} &= (d_j + d_l) \ \ (0 \le j < l \le 4), \qquad (4.31b)
\end{aligned}
$$

On the other hand, because γ and γ^{16} in Eq. (4.22a) are zeros of $g(t)$ in Eq. (4.20), the following relations are obtained with the *Vieta's formulas*.

$$
\gamma + \gamma^{16} = \mu, \qquad \gamma\cdot\gamma^{16} = \nu = \frac{\nu}{\mu}\cdot(\gamma + \gamma^{16}). \qquad (4.32)
$$

Thus, Eq. (4.22) is derived as

$$
\begin{aligned}
W = C \times \text{`}j\text{'}\mathbf{B} &= (D\gamma + E\gamma^{16})(Q_j\gamma + R_j\gamma^{16}) \\
&= (D+E)(Q_j+R_j)(\gamma\cdot\gamma^{16}) + DQ_j(\gamma+\gamma^{16})\gamma + ER_j(\gamma+\gamma^{16})\gamma^{16} \\
&= (D+E)(Q_j+R_j)\cdot\frac{\nu}{\mu}\cdot(\gamma+\gamma^{16}) + DQ_j\cdot\mu\cdot\gamma + ER_j\cdot\mu\cdot\gamma^{16} \quad (\because \text{Eq. (4.32)}) \\
&= \{D\delta_j + E\epsilon_j\}\gamma + \{D\epsilon_j + E\eta_j\}\gamma^{16}, \qquad (4.33a) \\
\delta_j = Q_j(\mu + \frac{\nu}{\mu}) + R_j\cdot\frac{\nu}{\mu},&\quad \epsilon_j = (Q_j+R_j)\cdot\frac{\nu}{\mu},\quad \eta_j = Q_j\cdot\frac{\nu}{\mu} + R_j(\mu+\frac{\nu}{\mu}). \qquad (4.33b)
\end{aligned}
$$

On the other hand, Eq. (4.29) is derived in the same way.

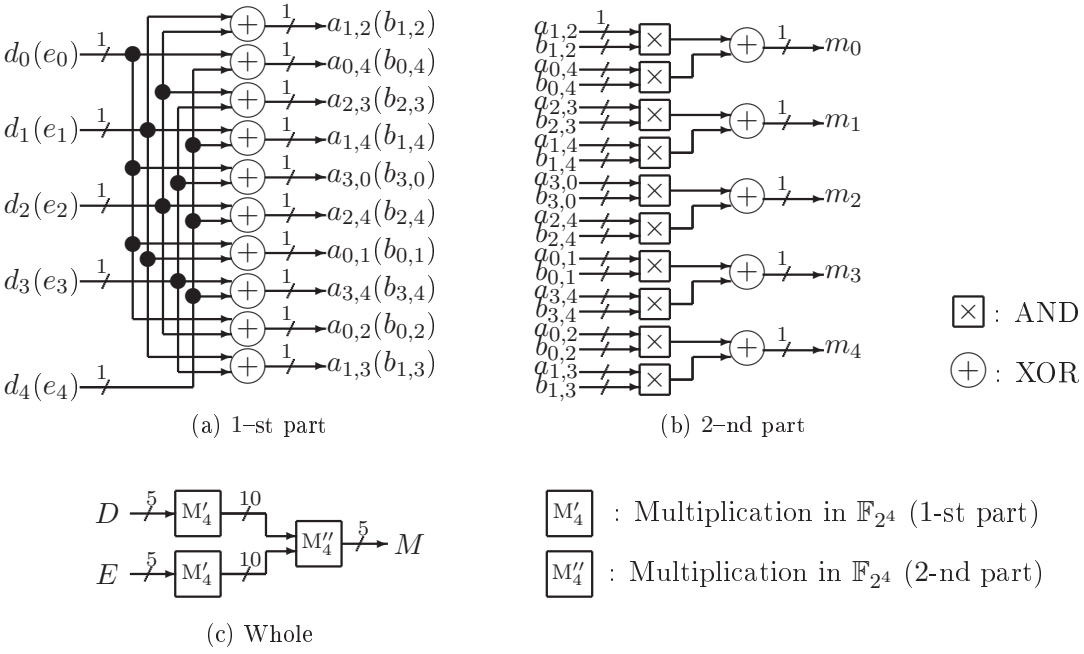


Figure 4.2: The multiplication circuit adopting RRB in $\mathbb{F}_{2^4}$

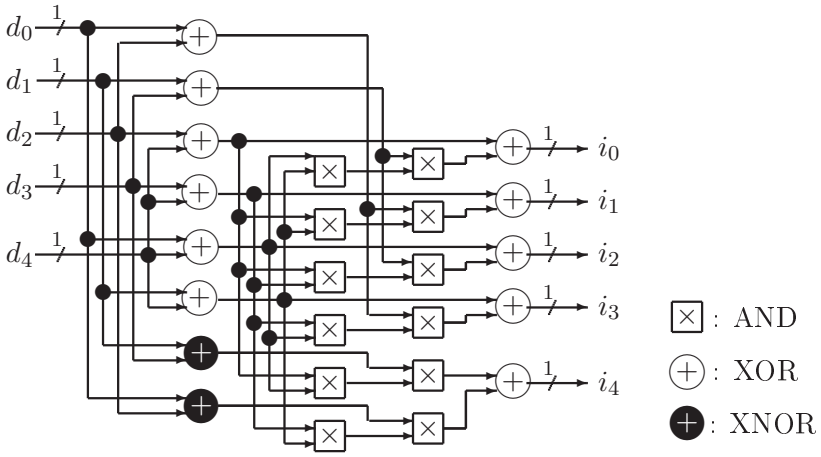


Figure 4.3: The inversion circuit adopting RRB in $\mathbb{F}_{2^4}$

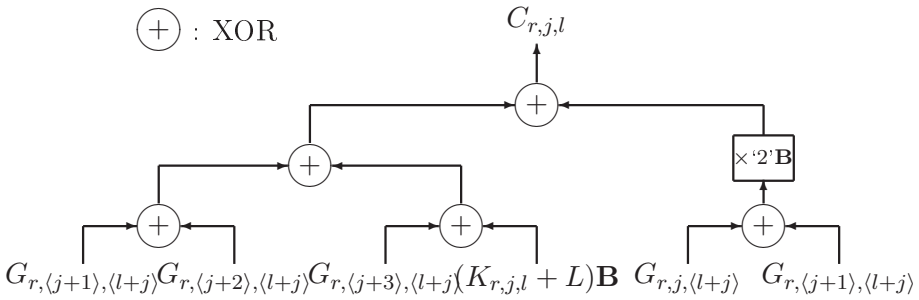


Figure 4.4: The calculation circuit of Eq. (4.4a)

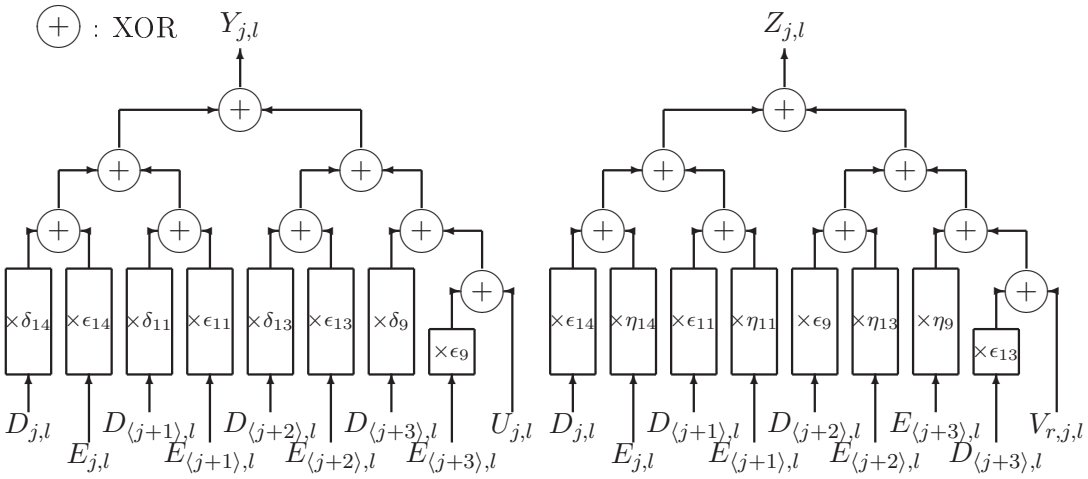
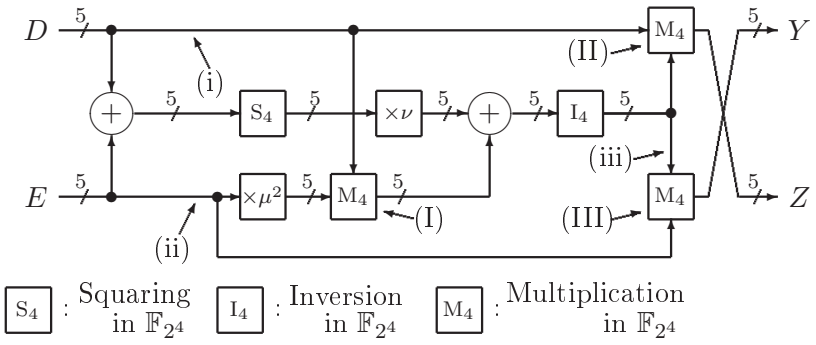
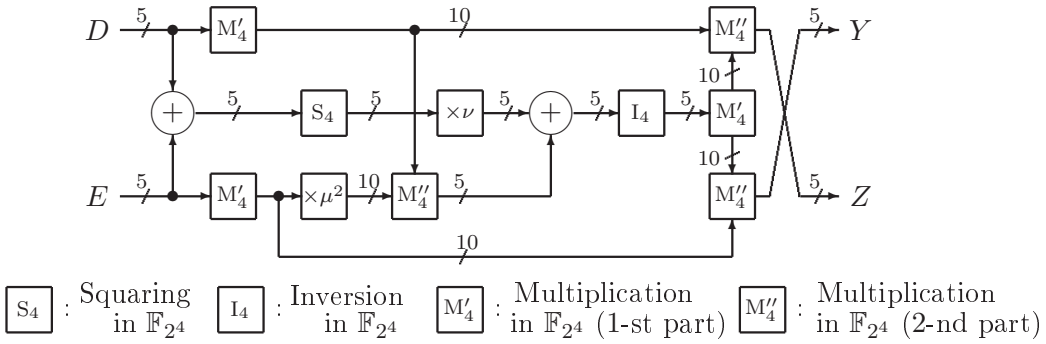


Figure 4.5: An example of the calculation circuit of Eq. (4.8)



(a) Before downsizing



(b) After downsizing

Figure 4.6: The inversion circuit adopting the normal basis in $\mathbb{F}_{(2^4)^2}$

Chapter 5

Conclusion

This thesis described as follows:

Chap. 2 briefly reviewed group and finite field theories.

Chap. 3 exteneded CVMA technique for type–$\langle h, m\rangle$ GNB. As the result, this extension improved some inefficient situations because it is possible that $h_{\min}$ becomes smaller by this expansion. After that, in order to *theoretically* obtain the tendency of the computational complexity of CVMA with respect to extension degrees, this chapter proposed an important theorem such that the existence probability of type–$\langle h, m\rangle$ GNB in $\mathbb{F}_{p^m}$ and the expected value of $h_{\min}$ can be explicitly obtained. Then, this chapter demonstrated the efficiency difference for $h_{\min}$ between type I–X and II–X CVMAs and the CVMA expanded for type–$\langle h, m\rangle$ GNBs.

Chap. 4 proposed RRB as how to make arithmetic operations in $\mathbb{F}_{(2^4)^2}$ more efficient, and MMMB as how to find more efficient basis conversion matrices. By utilizing RRB and MMMB, this chapter theoretically showed that the encryption and decryption circuits of AES can be provided by the critical path delay $4T_{\mathrm{AND}} + 13T_{\mathrm{XOR}}$.

Bibliography

[1] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), "ISO/IEC 27002: Information technology – Security techniques – Code of practice for information security management," 2005, available at "`http://webstore.iec.ch/webstore/webstore.nsf/Artnum_PK/41561`".

[2] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems Based on Pairings," The 17th Symposium on Cryptography and Information Security (SCIS 2000), No. C20, 2000.

[3] D. Boneh and M. Franklin, "Identity–based Encryption from the Weil Pairing," The 21st Annual International Cryptology Conference (CRYPTO 2001), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 2139, pp. 213–229, 2001.

[4] D. Boneh, A. Sahai, and B. Waters, "Functional Encryption: Definitions and Challenges," 8–th Theory of Cryptography Conference (TCC 2011), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 6597, pp 253–273, 2011.

[5] D. Boneh, X. Boyan, and H. Shacham, "Short Group Signatures," The 24th Annual International Cryptology Conference (CRYPTO 2004), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 3152, pp. 41–55, 2004.

[6] National Institute of Standards and Technology (NIST), "Data Encryption Standard (DES)," FIPS publication 46–3, available at "`http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`", 1999.

[7] National Institute of Standards and Technology (NIST), "Advanced Encryption Standard (AES)," FIPS publication 197, available at "`http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`", 2001.

[8] S. Takeuchi, Y. Sakemi, T. Okimoto, Y. Nogami, T. Nakanishi, J. Furukawa, and K. Sako, "How to Implement Furukawa–Imai Group Signature Scheme with Barreto–Naehrig Curve," Proceeding of the 4th International Workshop on SECurity (IWSEC2009), pp. 31–47, 2009.

[9] D. Bailey and C. Paar, "Optimal Extension Fields for Fast Arithmetic in Public–Key Algorithms," Asiacrypt 2000, Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 1976, pp. 248–258, 2000.

[10] A. A. Karatsuba and Y. Ofman, "Multiplication of Multidigit Numbers on Automata," Soviet Physics Doklady, Vol. 7, pp. 595–596, 1963.

[11] A. Weimerskirch and C. Paar, "Generalizations of the Karatsuba Algorithm for Efficient Implementations," Cryptology ePrint Archive Report, No. 224, 2006.

[12] A. L. Toom, "The Complexity of a Scheme of Functional Elements realizing the Multiplication of Integers," Soviet Mathematics, Vol. 3, pp. 714–716, 1963.

[13] S. A. Cook, "On the Minimum Computation Time of Functions," PhD Thesis, Harvard University Department of Mathematics, 1966.

[14] J. Chung and M. A. Hasan, "Asymmetric Squaring Formulae," Technical Report CACR 2006–24, Waterloo University Department of Electrical and Computer Engineering, 2006.

[15] H. Kato, Y. Nogami, T. Yoshida, and Y. Morikawa, "A Multiplication Algorithm in $\mathbb{F}_{p^m}$ Such That $p > m$ with a Special Class of Gauss Period Normal Bases," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E92–A, No. 1, pp. 173–181, 2009.

[16] Y. Nogami, A. Saito, and Y. Morikawa, "Finite Extension Field with Modulus of All–One Polynomial and Representation of Its Elements for Fast Arithmetic Operations," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E86–A, No. 9, pp. 2376–2387, 2003.

[17] R. Mullin, I. Onyszchuk, S. Vanstone, and R.Wilson, "Optimal Normal Bases in GF(p^n)," Discrete Applied Mathematics, Vol. 22, Issue 2, pp. 149–161, 1988.

[18] R. Granger, A. Moss, and N. P. Smart, "Efficient Arightmetic Modulo Minimal Redundancy Cyclotomic Primes," CSTR–09–004, Claude Shannon Institute, Ireland and University of Bristol, Aug. 2009.

[19] B. Baldwin, W. P. Marnane, and R. Granger, "Reconfigurable Hardware Implementation of Arithmetic Modulo Minimal Redundancy Cyclotomic Primes for ECC," Proc. of 2009 International Conference on Reconfigurable Computing and FPGAs, pp. 255–260, 2009.

[20] Y. Nogami, S. Shinonaga, and Y. Morikawa, "Fast Implementation of Extension Fields with TypeII ONB and Cyclic Vector Multiplication Algorithm," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E88–A, No. 5, pp. 1200–1208, 2005.

[21] H. Kato, Y. Nogami, T. Yoshida, and Y. Morikawa, "Cyclic Vector Multiplication Algorithm Based on a Special Class of Gauss Period Normal Basis," ETRI Journal, Vol. 29, No. 6, Dec. 2007.

[22] S. Gao, "Abelian Groups, Gauss Periods and Normal Bases," Finite Fields Application, Vol. 7, No. 1, pp.148–164, 2001.

[23] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," AES Algorithm (Rijndael) Information, "http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf", 1999.

[24] M. Matsui, "Linear Cryptanalysis Method for DES Cipher," EUROCRYPT 1993, Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 765, pp. 386–397, 1994.

[25] C. Paar, "Efficient VLSI Architectures for Bit–Parallel Computation in Galois Fields," PhD thesis, Institute for Experimental Mathematics, University of Essen, Germany, 1994.

[26] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi, "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic," Workshop on Cryptographic Hardware and Embedded Systems 2001 (CHES 2001), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 2162, pp. 171–184, 2001.

[27] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A Compact Rijndael Hardware Architecture with S–Box Optimization," ASIACRYPT 2001, Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 2248, pp. 239–254, 2001.

[28] S. Morioka and A. Satoh, "An Optimized S–box Circuit Architecture for Low Power AES Design," Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 2523, pp. 172–186, 2003.

[29] N. Mentens, L. Batina, B. Preneel, and I. Verbauwhede, "A Systematic Evaluation of Compact Hardware Implementations for the Rijndael S-box" CT–RSA 2005 Proceedings of the 2005 International Conference on Topics in Cryptology, pp. 323–333, 2005.

[30] D. Canright, "A Very Compact S-Box for AES," Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 3659, pp. 441–455, 2005.

[31] Y. Nogami, K. Nekado, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed Bases for Efficient Inversion in $\mathbb{F}_{((2^2)^2)^2}$ and Conversion Matrices of SubBytes of AES," Workshop on Cryptographic Hardware and Embedded Systems (CHES 2010), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 6225, pp. 234–247, 2010.

[32] Y. Jeon, Y. Kim, and D. Lee, "A Compact Memory-free Architecture for the AES Algorithm Using Resource Sharing Methods," Journal of Circuits, Systems, and Computers, Vol. 19, No. 5, pp. 1109–1130, 2010.

[33] Y. Nogami, H. Kato, K. Nekado, S. Uehara, and Y. Morikawa, "Finding a Basis Conversion Matrix Using a Polynomial Basis Derived by a Small Multiplicative Cyclic Group," IEEE Transactions on Information Theory, Vol. 58, No. 7, pp. 4936–4947, 2012.

[34] T. Elgamal, "A Public–Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," IEEE Transactions on Information Theory, Vol. 31, Issue 4, pp. 469–472, 1985.

[35] T. Itoh and S. Tsujii, "A Fast Algorithm for Computing Multiplicative Inverse in GF(2^m) Using Normal Basis," Information and Computation, Vol. 78, Issue 3, pp. 171–177, 1988.

[36] K. Nekado, Y. Nogami, H. Kato, and Y. Morikawa, "Cyclic Vector Multiplication Algorithm and Existence Probability of Gauss Period Normal Basis," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E94–A, No. 1, pp. 172–179, Jan., 2011.

[37] GNU Multiple Precision Arithmetic Library, available at "http://gmplib.org".

[38] P. L. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, Vol. 44, No. 170, pp. 519–521, 1985.

[39] J. Sándor, D. S. Mitrinovic, and B. Crstici, "Handbook of Number Theory I," Springer–Netherlands, 1995.

[40] D. Canright and L. Batina, "A Very Compact "Perfectly Masked" S-Box for AES (corrected)," The 6–th International Conference on Applied Cryptography and Network Security (ACNS 2008), Springer–Verlag, Lecture Notes in Computer Science (LNCS), Vol. 5037, pp. 446–459, 2009.