Spectral representations of discrete functions

Fu, Cheng

2005

Fu, C. (2005). Spectral representations of discrete functions. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46880>

<https://doi.org/10.32657/10356/46880>

Nanyang Technological University

Downloaded on 26 Apr 2023 21:09:16 SGT

402264

Spectral Representations of Discrete Functions

Fu Cheng

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2005

QA
9.59
F949
2005

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research done by me and has not been submitted for a higher degree to any other University or Institute.

18 Dec, 2005
Date

.....
Fu Cheng

*To all members in my family,
for their love and never-ending encouragement.*

Acknowledgements

My first, and most earnest, acknowledgement must go to my thesis advisor, Dr. Bogdan J. Falkowski. In every sense, none of this thesis would have been possible without his invaluable help. Dr. Bogdan J. Falkowski has been instrumental in ensuring my academic, professional, and moral well being ever since. I am extremely grateful for his continuous encouragement, forbearance and especially, for his diligent guidance through the past three years of my research work.

This research receives both the financial and technical support from School of Electrical & Electronic Engineering, Nanyang Technological University, which are greatly acknowledged.

I would also like to thank those anonymous reviewers for their help and comments that improved my various published papers.

Finally, I would like to show my sincere gratitude to Dr. Bogdan J. Falkowski again for the happy and profitable years I worked with him.

Summary

The research work discussed in this thesis concerns spectral representations of discrete functions for multiple-valued logic. The main objectives are the development of novel discrete transforms and fast algorithms for ternary and quaternary switching functions which have wide applications in logic design and optimization, VLSI systems and digital signal processing.

Families of fast linearly independent (LI) transforms based on Galois field (GF) (3) are shown. There are four different families of fast transforms introduced here. Each family of fast transforms is classified into several classes based on their matrices' properties and relations. These new fast transforms are implemented by C program and run on ternary benchmark functions to compare with ternary Reed-Muller transform both on the obtained number of non-zero spectral coefficients and the calculation times.

The concept of fast LI transforms is extended to standard algebra. Different families of transforms so called Linearly Independent Ternary Arithmetic (LITA) transforms are presented as well as the properties and relations among different classes and transforms. Computational complexity considerations for them are also discussed. These transforms show their advantages in experimen-

tal results when compared with ternary arithmetic transforms. Their application in ternary logic function verification is also discussed.

The representations of quaternary switching functions based on $GF(4)$ as well as standard quaternary algebra are presented in the following chapters. A number of Quaternary Linearly Independent (QLI) transforms are derived and classified into different classes. A novel approach to generate QLI transforms from corresponding binary transforms is also shown. The fastest QLI transforms based on $GF(4)$ and standard quaternary algebra are also presented. The computational costs and experimental results of QLI transforms are shown and compared with quaternary Reed-Muller and arithmetic transforms accordingly.

The hardware implementations of the proposed multiple-valued spectral transforms are shown. Both the ternary and quaternary LI transforms can be implemented on systolic processors by applying some of their matrices' properties. A universal logic circuit module is also proposed. It is shown that the universal module can be easily cascaded to realize higher level multiple-valued logic circuits.

Contents

Acknowledgements	i
Summary	ii
List of Figures	ix
List of Tables	xii
1 Introduction	1
1.1 Review of linearly independent and related transforms	4
1.2 Review of theory and applications of arithmetic transforms . . .	8
1.3 List of contributions	11
1.4 Outline of the thesis	13
2 Discrete Functions – Basic Representations and Transforms	17
2.1 Sets	17
2.2 Kronecker product	20
2.3 Union, intersection, and complement	21
2.4 Functions	22

2.5	The linear algebra structure of Galois functions	24
2.6	Reed-Muller and arithmetic transforms	28
3	Families of Linearly Independent Transforms over GF(3)	34
3.1	Basic definitions for LI transforms over GF(3)	35
3.2	F – R type fast LI transforms over GF(3)	40
3.2.1	Classification of fast LI transforms	40
3.2.2	Experimental results	64
3.3	F ² type linearly independent transforms over GF(3)	68
3.3.1	Classification of F ² type fast LI transforms	68
3.3.2	Computational costs	89
3.4	Classification of linearly independent helix transforms	93
3.4.1	Properties of helix transforms	93
3.4.2	Computational complexity	106
3.4.3	Experimental results	110
3.5	Classes of fastest linearly independent transforms over GF(3) and their properties	111
3.5.1	Ternary fastest linearly independent transforms	112
3.5.2	Properties and relations	126
3.5.3	Computational complexity considerations	129
3.5.4	Experimental results	132
4	Linearly Independent Ternary Arithmetic Transforms	137

4.1	Basic definitions of linearly independent ternary arithmetic transforms	138
4.2	F – R type linearly independent ternary arithmetic transforms	143
4.2.1	Basic recursive F – R type LITA transforms	144
4.2.2	F – R type LITA transforms with permutations	159
4.2.3	Experimental results	160
4.3	F ² type linearly independent ternary arithmetic transforms	166
4.3.1	Family of F ² type LITA transforms	166
4.3.2	Computation complexity consideration	180
4.4	Ternary arithmetic helix transforms	181
4.4.1	Properties and relations	182
4.4.2	Computational complexity	197
4.4.3	Experimental results	198
4.5	Fastest linearly independent ternary arithmetic transforms	200
4.5.1	Properties of fastest LITA transform matrices	201
4.5.2	Computational costs	212
4.5.3	Ternary logic function verification	213
4.5.4	Experimental results	215
5	Generation of Linearly Independent Transforms over GF(4)	220
5.1	Basic definition of linearly independent transforms over GF(4)	221
5.2	Generation of linearly independent transforms over GF(4)	223
5.2.1	Classes of quaternary linearly independent transforms	223

- 5.2.2 Complete example 231
 - 5.2.3 Experimental results 232
- 5.3 Quaternary linearly independent helix transforms 233
 - 5.3.1 Classification of QLIH transforms 234
 - 5.3.2 Experimental results 247
- 5.4 Classes of fastest quaternary linearly independent transforms . . 250
 - 5.4.1 Properties of FQLI transform 250
 - 5.4.2 Computational costs 258
 - 5.4.3 Experimental results 259
- 5.5 New approach to generate QLI transforms from binary case . . 261
 - 5.5.1 Generation of QLI transforms 261
 - 5.5.2 Experimental results 267
- 6 Quaternary Linearly Independent Arithmetic Transforms 268**
 - 6.1 Basic definitions 269
 - 6.2 Quaternary linearly independent arithmetic transforms 271
 - 6.2.1 Calculation of QLIA spectrum 283
 - 6.3 Quaternary arithmetic helix transforms based on Kronecker product 288
 - 6.3.1 Four basic types of QAH transforms 288
 - 6.3.2 A complete example 296
 - 6.3.3 Computational complexity 296
 - 6.3.4 Experimental results 297

<i>Contents</i>	viii
6.4 Fastest quaternary linearly independent arithmetic transforms .	299
6.4.1 Classification and properties of FQLIA transforms	300
6.4.2 Computational complexity and experimental results . . .	305
7 Hardware Implementations	307
7.1 Implementation of multiple-valued LI transforms on systolic proces- sor	308
7.1.1 F – R type ternary LI transforms	308
7.1.2 Fastest ternary LI transforms	311
7.1.3 Quaternary Kronecker arithmetic helix transforms	314
7.2 Logic circuit module of QLI transforms	318
8 Conclusion and Future Work	326
8.1 Conclusion	327
8.2 Future work	330
Author’s Publications	332
Bibliography	336

List of Figures

2.1	Addition and multiplication over $GF(3)$	25
2.2	Addition and multiplication over $GF(4)$	25
3.1	Forward butterfly diagram for the LI transform from Class A when $n = 3$	53
3.2	Forward butterfly diagram of $M_{n(a)}$	78
3.3	Forward butterfly diagram of $M_{n(b)}$	78
3.4	Logic circuit realization of M_Q	92
3.5	Forward butterfly diagram of Y1.1.	115
3.6	Inverse butterfly diagram of Y1.1.	116
3.7	Butterfly diagram of (3.78).	120
3.8	Butterfly diagram of (3.80).	120
3.9	Forward butterfly diagram of Z1.1.	123
3.10	Inverse butterfly diagram of Z1.1.	126
4.1	Forward butterfly diagram of M_a	147
4.2	Inverse butterfly diagram of M_a	147
4.3	Forward butterfly diagram of $M_{(4,21)}$	168

4.4	Forward and inverse butterfly diagrams of $M_{(4,29)}$	174
4.5	Forward butterfly diagram of $M_{n-(a)}$	177
4.6	Forward butterfly diagram of $M_{n-(b)}$	177
4.7	Forward butterfly diagram of $T_3^{(1)}$	204
4.8	Inverse butterfly diagram of $\left(T_3^{(1)}\right)^{-1}$	205
4.9	Forward butterfly diagram of $T_3^{*(1)}$	208
4.10	Inverse butterfly diagram of $\left(T_3^{*(1)}\right)^{-1}$	209
5.1	Relation between binary and quaternary linearly independent transforms.	262
7.1	Systolic processor implementation of F – R types ternary LI transforms.	310
7.2	Permutation part.	311
7.3	Butterfly diagram of $M_{(7,5)}$ when $n = 2$	312
7.4	Systolic processor implementation of ternary fastest LI transform in (7.5).	313
7.5	Butterfly diagram of $M_{(7,7)}$ when $n = 2$	314
7.6	Systolic processor implementation of ternary fastest LI transforms in (7.7).	315
7.7	Inverse butterfly diagram of RPH transform when $n = 2$	317
7.8	Part X systolic processor of RPH	319
7.9	Part Y systolic processor of RPH	320
7.10	Inverse butterfly diagram of LPH transform when $n = 2$	321

7.11 Inverse butterfly diagrams for QLI transforms $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$	322
7.12 Universal logic module based on QLI transforms.	323
7.13 Logic structure of QLI transform $M_{(a)}^{-1}$	323
7.14 Logic structure of QLI transform $M_{(b)}^{-1}$	324
7.15 Logic structure of QLI transform $M_{(c)}^{-1}$	324
7.16 Logic structure of QLI transform $M_{(d)}^{-1}$	325

List of Tables

2.1	Function table.	23
3.1	Class A of LI transforms over GF(3).	45
3.2	Class B of LI transforms over GF(3).	47
3.3	First part of Class C of LI transforms over GF(3).	48
3.4	Second part of Class C of LI transforms over GF(3).	50
3.5	First part of Class D of LI transforms over GF(3).	52
3.6	Second part of Class D of LI transforms over GF(3).	57
3.7	Class E of LI transforms over GF(3).	61
3.8	Class F of LI transforms over GF(3).	63
3.9	Number of non-zero spectral coefficients for $M_a - M_d$	65
3.10	Calculation times in seconds for $M_a - M_d$	66
3.11	Transform matrices and butterfly diagrams of Class S.	70
3.12	Transform matrices and butterfly diagrams of Class P.	72
3.13	Transform matrices and butterfly diagrams of Class S*.	73
3.14	Transform matrices and butterfly diagrams of Class P*.	74

3.15 Transform matrices and butterfly diagrams of the first half of Class T.	80
3.16 Transform matrices and butterfly diagrams of the second half of Class T.	81
3.17 Transform matrices and butterfly diagrams of the first half of Class Q.	83
3.18 Transform matrices and butterfly diagrams of the second half of Class Q.	84
3.19 Transform matrices and butterfly diagrams of the first half of Class T*.	86
3.20 Transform matrices and butterfly diagrams for the second half of Class T*.	87
3.21 Transform matrices and butterfly diagrams of the first half of Class Q*.	90
3.22 Transform matrices and butterfly diagrams of the second half of Class Q*.	91
3.23 Class LP of helix transforms.	94
3.24 Class RP of helix transforms.	97
3.25 Class LN of helix transforms.	100
3.26 Class RN of helix transforms.	102
3.27 Computational costs of helix transforms.	110
3.28 Number of non-zero spectral coefficients for helix transforms. . .	111
3.29 Fastest LI transforms in Class Y1.	113
3.30 Fastest LI transforms in Class Y2.	114

3.31	Fastest LI transforms in Class Z1.	124
3.32	Fastest LI transforms in Class Z2.	125
3.33	Number of non-zero spectral coefficients for Class Y1.	133
3.34	Number of non-zero spectral coefficients for Class Z1.	134
3.35	Calculation times for Class Y1.	135
3.36	Calculation times for Class Z1.	135
4.1	LITA transforms of Class A	145
4.2	LITA transforms of Class B	149
4.3	LITA transforms of Class C	150
4.4	LITA transforms of Class D	151
4.5	LITA transforms of Class E	153
4.6	LITA transforms of Class F	154
4.7	Number of non-zero spectral coefficients for LITA and TRM transforms.	162
4.8	Calculation times in seconds for LITA transforms.	163
4.9	Number of non-zero spectral coefficients for LITA transforms.	165
4.10	Transform matrices and butterfly diagrams for Class S.	170
4.11	Transform matrices and butterfly diagrams for Class P.	172
4.12	Class LP of ternary arithmetic helix transforms.	183
4.13	Class RP of ternary arithmetic helix transforms.	185
4.14	Class LN of ternary arithmetic helix transforms.	187
4.15	Class RN of ternary arithmetic helix transforms.	189

4.16	Number of non-zero spectral coefficients for ternary arithmetic helix transforms.	199
4.17	Calculation times for ternary arithmetic helix transforms.	200
4.18	Computational costs of fastest LITA transforms.	214
4.19	Number of non-zero spectral coefficients for T_n	216
4.20	Number of non-zero spectral coefficients for S_n	218
4.21	Calculation times in seconds for T_n and S_n	219
5.1	Number of non-zero spectral coefficients for QLI transforms.	233
5.2	Calculation times in seconds for QLI transforms.	234
5.3	Sub-class A of QLIH transforms.	237
5.4	Sub-class B of QLIH transforms.	238
5.5	Sub-class C of QLIH transforms.	240
5.6	Sub-class D1 of QLIH transforms.	241
5.7	Sub-class D2 of QLIH transforms.	243
5.8	Sub-class D3 of QLIH transforms.	244
5.9	Number of non-zero spectral coefficients for sub-class A.	248
5.10	Number of non-zero spectral coefficients for sub-class B.	249
5.11	Number of non-zero spectral coefficients for sub-class C.	249
5.12	Calculation times for sub-class A.	250
5.13	FQLI transforms in Class 1.	252
5.14	FQLI transforms in Class 2.	253
5.15	FQLI transforms in Class 3.	255
5.16	FQLI transforms in Class 4.	257

5.17	Number of non-zero spectral coefficients for FQLI.	260
5.18	Calculation times for FQLI.	260
5.19	Forward and inverse QLI transforms in Class 1.	263
5.20	Forward butterfly diagrams for QLI transforms 1.b – 4.b.	266
5.21	Number of non-zero spectral coefficients for QLI transforms in Class 1.	267
6.1	Four types of QLIA transforms.	274
6.2	QLIA transforms in Class A.	278
6.3	QAH transforms and their corresponding butterfly diagrams when $n = 2$	298
6.4	Computational costs for QAH transforms.	298
6.5	Number of non-zero spectral coefficients for RPH and RNH transforms.	299
6.6	Calculation times for RPH and RNH transforms.	300
6.7	Number of non-zero spectral coefficients for ${}^aM^{-1}$ and ${}^bM^{-1}$. . .	305

Chapter 1

Introduction

The spectral or polynomial representation of a switching function may be obtained by expressing it in terms of an orthogonal basis [AAE95, AAV00, Bea84, DDT78, ER82, Fal99, FSY97, GES87, Kar76, Kar85, KSY91, KSZ90, SF96, Shm89, SSJ98, SSS96, TDM01]. The most popular spectral representation of switching function is derived from discrete Walsh functions [AAE95, Bea84, ER82, FR96, FSP92, GES87, Kar76, Kar85, KSM01, KSY91, KSZ90, LM98, SA03, SF96, SSJ98, SSS96, TDM01]. The discrete orthogonal functions are referred to as the basis functions when the spectral or abstract harmonic terminology is used and as monomials when the polynomial terminology is used. Such polynomial representations have extensive applications in coding theory and cryptography as well as design of nonlinear filters [AAE95, Bea84, GES87, Kar85, KEA96, LM98, RF89]. Spectral techniques based on Walsh, Haar, Arithmetic and Reed-Muller transforms have been used in digital logic design for more than thirty years [AAE95, AKS88, Bea84, BP81, CH82, DDT78, DM94,

ER82, Fal94, GES87, Gre86, HS96, HSB02, Hur92, Kar76, Kar85, KB81, KS93, KSA00, KSM01, KSY91, KSZ90, Mil87, PM75, Red72, Sas93, Sas99, SF96, SSJ98, SSS96, Sch89, Shm89, Sta95, TDM01, Wol98]. These techniques have been used for switching function classification, disjoint decomposition, identification of symmetries, parallel and serial linear decomposition, spectral translation synthesis (extraction of linear pre- and post- filters), multiplexer synthesis, threshold logic synthesis, state assignment and testing, and evaluation of logic complexity [AAE95, HSB02, Kar76, Kar85, KSA03, RZ00, RZ02, SSS96, TDM01, TM97]. The renewed interest in applications of spectral methods in design of VLSI digital circuits is caused by their efficiency in linear sifting of decision diagrams [MST00, SA02] and in technology mapping problems [CMZ⁺93, DM94].

Besides binary functions represented as disjunctive or conjunctive normal forms, some other types of discrete representations, such as multiple-valued functions, cube sets, EXOR-based, spectral and probabilistic formulas have to be dealt with and the number of different spectral transforms and corresponding decision diagrams has been introduced for those various representations [AIS95, CFZ96, CMZ⁺93, CRC94, DDT78, DM94, DMSY02, Fal99, FC98, FSP92, FSY97, HM92, HS96, ISM04, JSD02, Ker01, KS75, KSM01, KSY91, KSZ90, LPV94, MSS02, PRG⁺95, PSB93, SA02, SA03, Sas93, Sas99, SF96]. In recent years there have been major advances in integrated circuit theory that have made feasible implementation of complex systems [CCH⁺99, Wol98]. They include the following two aspects. Design reuse and flexibility is essential to map an ever-growing functionality onto an ever-evolving hardware. Designs have to be captured at the highest level of abstraction to be able to exploit all the degrees of freedom that are available. While the majority of appli-

cations of spectral transforms are still for binary case, these transforms can also be modified for multiple-valued input multiple-valued output discrete functions [Hur84, Muk86, MW86, Per92, Smi81]. Another important application of multiple-valued logic and corresponding spectral transforms is in the process of formal verification by representation of binary sub-circuits at a higher level of abstraction [Dre00]. The following reasons motivate an ever-increasing use of multiple-valued logic combined with spectral methods to overcome the main two problems of current VLSI technology: interconnections and chip-pin-out problems [CCH⁺99, Wol98]. Multiple-valued logic can represent the problem much more concisely than it could be in binary logic and appear as an adequate tool for coping with problems of higher size of complexity [DDT78, KSZ90, MW86]. Multiple-valued approach increases the information density over the interconnections and thus their efficiency. Multiple-valued logic can be represented through reduced representations such as decision diagrams and arrays of cubes - properties of these discrete representations have influence on final circuit design [HSB02, MST00, MT98, RK95, SF96, TH79].

Decision diagrams are a data structure permitting efficient representation of discrete functions and Ordered Binary Decision Diagrams (OBDDs) have become popular representations of binary functions due to their canonicity and efficient algorithms to operate on them [Bry86, Bry91, HSB02, KSA00, MT98, SF96, WFMW00]. However there exist some binary functions for which OBDDs are exponential in terms of number of their variables [Bry91, MT98, WFMW00]. A number of different variants of decision diagrams based on modification of Shannon expansion introduced bit-level and word-level decision diagrams that are useful in logic synthesis, verification and testing [HSB02, Mil87, SF96, Sas99]. A lot of research work was also done on reduction of decision dia-

grams using various techniques combining frequently binary logic and its Walsh spectral domain such as shifting [MST00, MT98], spectral linear transforms with variable reordering techniques [MST00] and even more general spectral autocorrelation approach [KSA03]. It should be also noticed that all the advantages in reduction of decision diagrams through spectral techniques can be also used in design of minimal trellises that are graphical representations of error-correcting codes as there exist direct correspondence between these two representations as shown in [LV99]. Moreover as the trellises are closely related to Reed-Muller expansions all the modifications and decision diagrams based on Shannon expansions will be important also for this research too. Recently various generalizations of Shannon expansions have been studied to develop new decision diagrams based on Linearly Independent Logic and function driven expansions as well as their combination [Ker01, KP03].

1.1 Review of linearly independent and related transforms

The idea to represent switching functions in different algebra has evolved for many years. The Shannon, Reed-Muller and related spectral expansions used in logic synthesis, analysis, classification, and testing were discussed by many authors [AAE95, AAV00, Al-04, BP81, Bro90, BS91, DDT78, Dre00, GES87, GT74, Gre86, Gre89, HM92, HSB02, Kar76, KSA00, KSZ90, MF97, Muk86, MW86, Per92, Per93, RF99a, RK95, SA02, SA03, Sas93, Sas99, SF96, Sha38, SM92, SSJ98, SSS96, Sta00, Sta01, TM97, Tos72, WFMW00, Yan98, YDS02, YTS02]. For many important functions that are non-unate (e.g., parity check-

ers, adders and multipliers), Reed-Muller and related polynomial expansions are advantageous when area, speed and testability are of main concern [HSB02, Sas93, Sas99, SF96, WFMW00]. For such linear functions, they allow surprisingly complex designs to be implemented using very few product terms. It is important to notice that Shannon's expansion is a special case of an expansion using an orthonormal basis based on generalized cofactors that may not be unique [Bro90]. When such orthonormal basis is performed over $GF(2)$ the concept of Linearly Independent (LI) logic has been proposed [Per93, PSB93]. Family of LI transforms that possess fast forward and inverse butterfly diagrams in different classes and their basic properties have been obtained [FR97]. LI transforms are also used as the underlying transforms to develop new types of word decision diagrams [KP03, PFCD02]. The binary Reed-Muller transform itself is only one type of LI transforms presented in [FR97]. Generalization of LI transforms for multiple-valued input binary functions was proposed in [FR01]. Properties of LI transform matrices and their complexity analysis for multiple-valued input binary output functions with appropriate coding and new matrix functions were discussed in [FR00, FR01]. Since the number of all possible LI polynomial expansions even for binary functions with small number of variables is very large, finding efficient ways of generating the expansions is of great importance. In [FR97], classes of binary LI transforms that possess fast forward and inverse butterfly diagrams were identified and their recursive definitions were given together with the theory that allows one to find and calculate optional LI expansions by using fast transforms. Similar classification and fast transform methods for ternary [FF03, FF] and quaternary [FF05c, FF05a, FF05b] LI logic have also been developed.

Reed-Muller transform has been successfully applied in many areas such as

signal and image processing, fault detection and coding techniques, especially those concerned with group or block codes for error control [AAE95, Fal04, Kar85, KSZ90, LM98, RF89]. Logic synthesis and testing using the Reed-Muller based spectral expansions were also discussed by many authors [Bro90, DDT78, Gre86, Mil87, Red72, SA03, Sas99, SF96, SSJ98, SSS96, KSZ90]. They are of great interest because the Reed-Muller based circuits have many desirable properties, such as simple testability vectors [Red72, SF96]. The binary Reed-Muller transform itself is only one type of LI transforms that are the special case of more general orthonormal expansions discussed in [Bro90, DM94, KS75]. Determination of compact representations for ternary functions is an important task, since complexity, in terms of space and time, of any usage of them strongly depends and is determined by the complexity of the representations used. This is a rather complicated task due to exponential complexity in the number of variables count. The problem is even more complex in the case of ternary and multiple-valued logic functions due to the increased number of possible values for their variables [SA02, SSS96]. Binary fixed polarity Reed-Muller expansions have been extended to a prime field $GF(q)$ [DDT78, Gre89, GT74, KSZ90, SSS96].

There are different strategies in the analysis, design and testing of binary logic circuits. Although the binary values of 0 and 1 are normally accepted to define binary logic, there are also other mathematical relationships, which can be used. These alternatives have been widely researched [Che92, DDT78, KSZ90, KSY91, RZ00, RZ02, SF96, Sas99, Sch89, Shm89, Sta01, SA03] and they offer certain advantages for digital design and test in comparison with binary representation. One of them is to combine spectral techniques based on Walsh and other transforms with theory of logic functions using known concepts like Chow

parameters, Boolean derivatives to obtain new tools [KEA96]. Two closely related transforms are frequently used in logic design and testing. They are the Reed-Muller transform based on $GF(2)$ and the arithmetic transform performed in standard arithmetic [AAE95, AIS95, AKS88, DDT78, Fal99, Gre86, Hei91, Hur92, KSZ90, KSY91, KB81, Mal97, PM75, Shm89, Sta95, SSJ98, SA03]. The inverse of arithmetic transform when applied directly to the functional data is called adding transform [FC97]. The detailed properties of transform kernels for both arithmetic and adding transforms and analysis of the complexity to calculate both transforms was presented in [FC97]. It has been shown [AAE95, DDT78, Fal99, Hei91, Hur92, KSZ90, KSY91, KB81, Mal97, PM75, Sch89, Shm89] that, in many applications where logic functions need to be analyzed, it is useful to transform a Boolean function into the equivalent arithmetic expansion. In particular, such arithmetic expansions are used in probabilistic verification of a pair of functions [AAE95, KB81, Sch89] and to describe stochastic behavior of a logical circuit under random values of input variables [Che92, Sch89]. Arithmetic expansion was also used in testing [Che92, Hei91, Hur92, RZ00, RZ02], in probabilistic analysis of logical circuits [KB81, PM75] in reliability analysis [Fal99, Hei91, Mal97] and in the analysis of logic circuits and systems with the help of arithmetic transform [AIS95, AKS88, DMSY02, KS93, KSZ90, KSY91, LPV94, MSS02, SA03]. The adding transform has been recently proposed as a signature-based library search mapper which is able to distinguish the nonequivalent functions with very low aliasing [CRC94].

1.2 Review of theory and applications of arithmetic transforms

The polynomial form of Boolean functions discussed by many authors is nothing else than the orthogonal expansions of the corresponding Boolean functions by the set of basis functions for an arithmetic transform. It should be noticed that Boole himself used the standard algebra in his description of Boolean functions [Boo54]. As the arithmetic transform matrix is Kronecker based then it may be easily computed by means of fast transform [AAE95, FC97, FSY97, KSZ90, KSY91, SSJ98, TDM01, Yan98]. While more efficient than the standard transform matrix multiplication, this method requires the full vector of the original Boolean function to be operated on. However, frequently in modern CAD VLSI systems these functions are represented in reduced forms such as array of cubes or Binary Decision Diagrams (BDDs) which would have to be expanded to full minterm truth vectors in order to apply fast algorithms. Hence methods have been developed that allow calculating the coefficients directly from arrays of cubes or different decision diagrams [FC97, FC98, LPV94, SF96, SA03]. Moreover, these methods allow also calculating only some chosen arithmetic coefficient without generation of the whole spectrum what is useful in some applications of arithmetic spectra, for example in testing. Link of arithmetic transforms with different computationally efficient graphical representations of Boolean functions, such as OBDDs, FBDDs, EVBDDs and MTBDDs have also been shown [CFZ96, FC98, LPV94, MSS02, SF96, Sta95, SSJ98, SA03]. The detailed discussion of status and achievements for arithmetic transform was done in recent review papers [FSY97, Fal99, SM94] and the interested reader is referred to them.

If all the variables in arithmetic expansion are nonnegated, such an expansion is in zero polarity. When some of the variables in arithmetic expansion are uniformly negated in all the terms, the expansion is in some polarity not equal to zero. There are arithmetic expansions corresponding to all possible assignments of all the variables in different polarities and such a family of arithmetic expansions is known under the name of generalized arithmetic expansion [AAE95, FC97, FC98, KSZ90, KSY91, Mal97, SSJ98]. It is obvious that, for some Boolean function, an optimal arithmetic expansion having the smallest number of non-zero coefficients may happen for some polarity not equal to zero. Therefore, when searching for an efficient representation of Boolean functions in standard arithmetic domain, the generalized arithmetic expansions should be considered rather than more known arithmetic expansion in zero polarity. Efficient methods to calculate best polarity have been developed [KSZ90, KSY91, Mal97, SSJ98].

As shown in [AKS88, Mal97], parallelization of logical computations is possible and under certain conditions effective when we convert an arithmetic representation of a system of functions into a single form of an arithmetic polynomial. This approach has been extensively developed in [AKS88, Mal97, Shm89]. In [KS93] a canonical structure is proposed that assure realization of Boolean functions in terms of linear arithmetic polynomials. Matrix relations for computation of the coefficients of the linear polynomials are derived. The linearity conditions of different orders are presented which need to be satisfied in order to obtain simplification of the final algorithmic structure. It is shown that the proposed approach is more efficient than the realization of Boolean functions by threshold devices [KS93].

Arithmetic polynomial forms for multiple-valued functions were considered in [DMSY02, Tos72, Yan98]. The theories of logic algebra for a large number of analytical expressions for multiple-valued functions in terms of the discrete Fourier transform were developed in the book [KSZ90, Yan98]. The synthesized expressions take into account diverse required modes of operations such as arithmetic, Galois Fields or modular algebra. Hence, arithmetic polynomial expansions for multiple-valued functions are considered together with expansions in different algebraic structures known in the literature under the name of multiple-valued Reed-Muller transforms.

In [AIS95] another algorithm for representing a system of m -valued functions by linear arithmetic polynomials was shown. The approach used in this work was to represent the system of multiple-valued functions through the decomposition and a discrete orthogonal transform by a system of linear arithmetic polynomials. Also, by using this representation it is possible to restore completely the original system of multiple-valued functions. The same and related work presented the possibility of implementing considered algorithms on linear systolic arrays [Yan94]. Linear and planar models of VLSI circuits used in optimization and reduction of interconnections based on multiple valued representations of arithmetic polynomials were discussed in [DMSY02, YDS02, YTS02].

The broadest approach in further generalization of arithmetic expansion is to allow one to formulate such an expansion in terms of standard addition and subtraction operating on an arbitrary logic function or logic expression. It is obvious that, in the special case of the truth vector for AND function, such an expansion becomes mixed arithmetic where each variable can have an arbitrary polarity throughout all terms in the expansion or generalized arithmetic expansion.

sion. In the majority of cases, the new expansion has its own properties and may be advantageous in different applications of arithmetic-type expansions in logic design. The concept of such generalization known as Linearly Independent Arithmetic (LIA) transforms based on the inverse matrices from LI logic transforms [FR97, Per93] and forward matrices calculated in standard arithmetic has recently been introduced [RF99b]. Various classes of LIA transforms and their basic properties have been discussed [RF99b]. Arithmetic transform falls into the broad definition of LIA logic. Comparisons among various LIA transforms show that for many classes of logical functions some LIA transforms outperform arithmetic transform when applied to fault detection [RF99a]. LIA transforms are also used as the underlying transforms to develop new types of word decision diagrams [SSJ98, Sta01, SA03]. Recently Arithmetic Ternary Decision Diagrams with the sign alternating factor have been developed for generation of arithmetic expressions and compared with other known Ternary Decision Diagrams [SA03, Sta00].

1.3 List of contributions

The following contributions have been made during the completion of the thesis:

1. The concept of linearly independent transforms is extended to multiple-valued logic based on Galois field (3) and (4).
2. The extensions of arithmetic transforms are investigated for ternary and quaternary-valued logic.
3. F – R type linearly independent transforms are classified based on ternary and quaternary-valued logic. Their fast transforms and butterfly diagrams

are shown.

4. Ternary linearly independent transforms of F^2 type are presented. Their properties and fast transforms are discovered.
5. Novel helix transforms are proposed. Four basic types of helix transforms are derived into four complete classes of linearly independent transforms over both Galois field and standard algebra.
6. Multiple-valued fastest linearly independent transforms are shown with their comparing results with corresponding ternary and quaternary Reed-Muller and arithmetic transforms.
7. An efficient matrix decomposition method is using in the thesis to calculating the spectrum of the novel transforms.
8. Computational complexities and costs of different linearly independent transforms are discussed.
9. The revised MCNC, IWLS'93 and ISCAS'95 benchmark functions are used to compare the experimental results of the novel linearly independent transforms with the results of multiple-valued Reed-Muller transform or arithmetic transform, correspondingly.
10. The application of using ternary arithmetic transforms for ternary logic functions verification is discussed
11. Implementations of different multiple-valued linearly independent transforms on systolic processors are shown.
12. A novel universal logic module of quaternary linearly independent transforms is proposed.

While part of the contributions listed in this section has been published in international journals and proceedings of conferences, others are currently under review. The complete list is shown in "**Author's Publications**".

1.4 Outline of the thesis

The research work discussed in this thesis concerns spectral representations of discrete functions for multiple-valued logic. Its focus is on novel linearly independent transforms, efficient representations, calculation and applications of ternary and quaternary for switching functions.

Some common definitions needed in the discussion of the thesis are presented in Chapter 2. Some basic mathematic ideas and concepts are also surveyed. The definitions and properties of Reed-Muller expansion and arithmetic transforms are shown.

The concept of Linearly Independent (LI) transforms is initially extended to Galois Field (GF) (3) in Chapter 3. Three families of novel introduced fast LI transforms, namely $F - R$ type, F^2 type, and helix transforms are discussed in details for their properties, mutual relations and efficient computational methods. Each type of the fast LI transforms contains hundreds of transforms, which are categorized into several classes based on their common relations and properties. Fast calculation methods are also shown to obtain their inverse transform matrices and butterfly diagrams. The family of fastest LI transforms over $GF(3)$ is also investigated in this chapter. These fastest LI transforms contain the minimal number of non-zero elements and can be easily implemented in terms of reversible logic gates. The presented ternary LI transforms show their advan-

tages in obtained spectrum and calculation speed when they are implemented and tested by revised ternary benchmark functions when compared with ternary Reed-Muller transform.

In Chapter 4, the extension of LI transforms based on standard ternary algebra is discussed. Three types of fast Linearly Independent Ternary Arithmetic (LITA) transforms are classified and shown with their properties and mutual relations. Each family is corresponding to their Galois field expressions given in previous chapter. Following the proposed concept of fastest LI transforms over $GF(3)$, the fastest LITA transforms are also proposed. One of its application in ternary logic function verification is discussed in detail. The considerations of the computational complexity for the novel LITA transforms are presented. The experimental results of these LITA transforms are compared with ternary arithmetic transform in terms of non-zero spectral coefficients and calculation speed. The novel expansion can have applications in spectral representations of ternary logic functions as well as calculation of their stochastic behavior [Che92, Sch89]. The different butterfly structures introduced should be considered as more important than simple as fast algorithms to calculate LITA transforms and expansions.

Chapter 5 shows the Quaternary Linearly Independent (QLI) transforms over $GF(4)$. The family of F – R type QLI transforms is introduced. Quaternary Linearly Independent Helix (QLIH) transforms and Fastest Quaternary Linearly Independent (FQLI) transforms are shown in consequence. Their recursive equations defining the forward and inverse transforms are presented. The butterfly diagrams are also derived based on matrix decomposition method to provide efficient calculation of the new transforms. Their computational

complexities are discussed and the experimental results are presented. Another special family of QLI transforms is also presented, which shows their tight relations with binary LI transforms so that they can be generated directly from binary LI transforms.

In Chapter 6, Quaternary Linearly Independent Arithmetic (QLIA) transforms are introduced. Similar to the relation between the algebra over $GF(3)$ and ternary standard algebra, linearly independent logic based on $GF(4)$ can also be transferred into standard quaternary algebra. In this chapter, Quaternary Arithmetic Helix (QAH) transforms and Fastest Quaternary Linearly Independent Arithmetic (FQLIA) transforms are also proposed and classified into different classes. Their properties and relations have been investigated. A novel fast calculation method is shown in this chapter for the calculation of QLIA spectrum. The computational complexities and experimental results of QAH and FQLIA transforms on quaternary benchmark functions are shown and compared with Quaternary Arithmetic (QA) transform. Their applications in logic synthesis and circuit design are also discussed. These new expansions can have applications to describe probabilistic and stochastic behavior of quaternary and related binary functions as well as in statistical analysis of nonlinear filters providing more flexibility than the current methods based on binary case.

Two kinds of hardware implementations using the new spectral transforms are discussed in Chapter 7. First is the implementations of multiple-valued LI transforms on systolic processor. The F – R type and fastest ternary LI transforms are given as examples. It is shown that the new LI transform can be easily implemented using matrix decomposition method and they have advantages in terms of saving the number of the computing and storage cells of systolic proces-

sors. When the QAH transforms are implemented, they need to be separated into two parts of systolic structures that are connected in series. The non-zero elements in the helix transforms are always located along the two sides of the diagonal/reverse-diagonal. It will cause conflicts when the inputs are sent to a FIFO system such as systolic processor. It is shown in this chapter that two sequentially connected systolic structures can solve this problem and give compact hardware structures of QAH transforms. This idea can apply to all kinds of helix transforms. Second, a universal logical module based on the first stage QLI transforms is proposed. For the reason that all the QLI transform matrices contain only two kinds of non-zero submatrices, the presented universal logical module can be easily cascaded to realize higher level quaternary logic circuits.

The final chapter comments on the future work that can be undertaken and has a conclusion of the work presented in this thesis.

Chapter 2

Discrete Functions – Basic Representations and Transforms

In this chapter, some basic mathematic ideas and concepts will be surveyed. Some common definitions needed in the following chapters will also be presented and can be found in [AAE95, AAV00, Al-04, Bea84, Boo54, BP81, Bro90, BS91, BS95, Che92, DDT78, DM94, ER82, GE78, GES87, Gra81, Gre86, Gre89, Gre90, GT74, GT76, Hei91, Kar76, KSY91, KSZ90, Mal97, MW86, RF89, Sas93, Sas99, Sch89, SF96, Sha38, SSJ98, SSS96, TDM01, Tos72, Yar85].

2.1 Sets

A *set* is a collection of objects without repetitions; the objects in the collection are called its *elements*. Having said that, in order to affirm the view of a set doubtless shared by the reader, we back away and declare the words "set"

and "element" to be primitive. They can be described but not defined, being analogous in this respect to the words "line" and "point" in geometry. Our intent therefore is not to say what sets are but to discuss a few of the things that many done with them legally, inasmuch as they provide us with great notational convenience. In this thesis, only *finite sets*, i.e., sets possessing a finite number of elements, are considered.

We write

$$x \in A \tag{2.1}$$

to signify that x is an element (or member) of a set named A .

A set maybe described by *enumeration*, i.e., by an explicit listing of its elements, written within curly braces. A set with three elements A , B , and C may be written by

$$\{ A, B, C \}$$

A second way to describe a set is by means of a *membership-property*. To specify a set S in this way, we write

$$S = \{x | P(x)\}$$

where $P(x)$ is a predicate that is true if and only if $x \in S$.

Some sets are described conveniently either by enumeration or by a membership-property; the choice between the two equivalent specifications

$$S = \{ 1, -1, i, -i \}$$

and

$$S = \{x \mid x^4 = 1\},$$

for example, is clearly a matter of taste.

A third way to describe a set is by means of a *recursive rule*. Consider the set

$$B = \left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \dots \right\}$$

of binary codes. This set is described by the following statements:

- 1. $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is an element of B .

- 2. If $[X]$ is an element of B , then so is $\left[\begin{array}{c|c} 0 & X \\ \hline 1 & \\ \vdots & \\ 1 & X \end{array} \right]$.

- 3. Nothing else is an element of B .

2.2 Kronecker product

The *Kronecker (cartesian) product* of sets S and T , written $S \otimes T$, is the set defined by

$$S \otimes T = \{(x, y) \mid x \in S \text{ and } y \in T\} \quad (2.2)$$

When

$$S = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad T = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix},$$

then

$$S \otimes T = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{bmatrix}.$$

The Kronecker product is not commutative, i.e., $S \otimes T \neq T \otimes S$ except for special S, T pairs. The cardinality of $S \otimes T$ is related to that of the individual sets S and T by

$$\#(S \otimes T) = (\#S) \cdot (\#T). \quad (2.3)$$

Thus the sets $S \otimes T$ and $T \otimes S$ have the same cardinality.

The Kronecker product is a set of ordered pairs. We define a three-fold Kronecker product as a set of ordered triples, i.e.,

$$R \otimes S \otimes T = \left\{ (x, y, z) \mid x \in R, y \in S, z \in T \right\}. \quad (2.4)$$

Higher-order Kronecker products are defined by obvious extension. We write S^n to signify the n -fold Kronecker product of S with itself, i.e.,

$$S^n = \overbrace{S \otimes S \otimes \dots \otimes S}^{n \text{ times}}. \quad (2.5)$$

2.3 Union, intersection, and complement

Let S and T be sets. Then the *union* of S and T , written $S \cup T$, is defined by

$$S \cup T = \{x \mid x \in S \text{ or } x \in T \text{ or both } \}. \quad (2.6)$$

Thus $\{a, b, c\} \cup \{a, b, d\} = \{a, b, c, d\}$.

The *intersection* of S and T , written $S \cap T$, is defined by

$$S \cap T = \{x \mid x \in S \text{ and } x \in T\}. \quad (2.7)$$

Thus $\{a, b, c\} \cap \{a, b, d\} = \{a, b\}$. Two sets S and T are said to be *disjoint* if $S \cap T = \emptyset$, i.e., if they have no elements in common.

The *relative complement* of T with respect to S , written $S - T$, or $S \setminus T$, is defined by

$$S - T = \{x \mid x \in S \text{ and } x \notin T\}. \quad (2.8)$$

Thus $\{a, b, c\} - \{a, b, d\} = \{c\}$. In many situations it is natural to think of a *universal set*, U , such that every set of interest is a subset of U ; in such

situations we define the *absolute complement* of T , written $\overline{T}$, by

$$\overline{T} = U - T = \{x \mid x \in U \text{ and } x \notin T\}. \quad (2.9)$$

The statement ' $x \in U$ ' is redundant, however, in the given context, thus we may define $\overline{T}$ by the simpler statement

$$\overline{T} = \{x \mid x \notin T\}. \quad (2.10)$$

2.4 Functions

A *function* f from a set S into a set T , written

$$f : S \rightarrow T, \quad (2.11)$$

assigns to every element $x \in S$ an element $f(x) \in T$ called the *image* of x . The set S is called the *domain* of f ; the set T is called the *co-domain* of f . The *range* of f is the set of images of elements of S under f . The range of f is clearly a subset of its co-domain; if the range of f is equal to T , we say that the function (2.11) is *onto* T .

A function is a specialized relation. We recall that a relation from S to T is a subset of $S \otimes T$, i.e., a collection of ordered pairs each of which takes its first element from S and its second element from T . accordingly, we define a function f from S into T as a relation from S to T having the property that each element of S appears as a first element in exactly one of the ordered pairs in f . Thus the formulas ' $f(x) = y$ ' and ' $(x, y) \in f$ ' are equivalent predicates.

Table 2.1: Function table.

x	$f(x)$
a	a
b	d
c	a

Example 2.4.1. Suppose $S = \{a, b, c\}$, $T = \{a, c, d\}$, and define a function f from S into T by the statements

$$\begin{aligned} f(a) &= a \\ f(b) &= d \\ f(c) &= a. \end{aligned} \tag{2.12}$$

Then f is the relation

$$f = \left\{ (a, a), (b, d), (c, a) \right\}. \tag{2.13}$$

An alternative way to specify a function is by means of a function-table. Such a table for the present example is shown in Table 2.1. □

n-variable functions. A function

$$f : S^n \rightarrow T \tag{2.14}$$

is called an *n-variable function* from S into T . Thus the temperature-values in a $10' \times 10' \times 10'$ room are described by a 3-variable function from $S = \{x \mid x \text{ is a real number between 0 and 10}\}$ into $T = \{x \mid x \text{ is a real number}\}$.

2.5 The linear algebra structure of Galois functions

Galois field and functions. A *finite field* is a field having a finite number of elements. Such a field must have some (finite) prime characteristic p . Hence, its order must be some power $q = p^m$ of the prime p ; every finite field have prime-power order. Finite fields are also called *Galois Fields*. In this thesis, we suppose that $\text{GF}(p^m)$ is a given field of prime-power order $q = p^m$.

The addition $f + g$ and the multiplication $f \cdot g$ of two Galois functions f and g :

$$\begin{aligned} f &: \{\text{GF}(q)\}^n \rightarrow \text{GF}(q) \\ g &: \{\text{GF}(q)\}^n \rightarrow \text{GF}(q) \end{aligned}$$

are defined to be the componentwise extensions of the addition and of the multiplication in $\text{GF}(q)$ respectively, i.e.:

$$\begin{aligned} [(f + g)_e] &= [f(e) + g(e)], \\ [(f \cdot g)_e] &= [f(e) \cdot g(e)]. \end{aligned}$$

Let $l \in \text{GF}(q)$; the addition $l + f$ and the multiplication $l \cdot f$ of the Galois function f and of the field element l is defined by the above equalities when the function g reduces to a constant function l over $\text{GF}(q)$. Figs. 2.1 and 2.2 present the addition and multiplication tables for $\text{GF}(3)$ and $\text{GF}(4)$.

With these definitions, the set of Galois functions $\{\text{GF}(q)\}^n \rightarrow \text{GF}(q)$ is a

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

×	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Figure 2.1: Addition and multiplication over GF(3).

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

×	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	3	1
3	0	3	1	2

Figure 2.2: Addition and multiplication over GF(4).

linear algebra of functions.

Orthogonal basis functions. The basis functions are used to represent a specified function by a sum of elementary functions. Such representations make possible straightforward mathematical operations and often result in a better understanding of the system behavior.

Let $x_i, i = \left(1, 2, 3 \right)$, known as a basis set, denote the three orthogonal unit vectors along the Cartesian coordinate axes. An arbitrary vector $\vec{F}$ can be expanded in the form

$$\vec{F} = \vec{F}_1x_1 + \vec{F}_2x_2 + \vec{F}_3x_3 = \sum_{i=1}^3 \vec{F}_ix_i \tag{2.15}$$

where the $\vec{F}_i$'s are the projections of $\vec{F}$ on each coordinate axis. By taking into consideration the orthogonality of the vector set x_i , which is defined by

$$x_i \cdot x_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad (2.16)$$

the components of $\vec{F}$ are specified by

$$\vec{F}_i = \vec{F} \cdot x_i \quad (2.17)$$

where $\vec{F}_i$ is the projection of the vector $\vec{F}$ on the x_i axis.

Linearly independent algebra. One of the most fundamental ideas of linearly independent algebra is that of a linearly independent transform. A *linearly independent transform* or *linear map* $T : R^n \rightarrow R^m$ is a map that preserves vector addition and scalar multiplication.

An $m \times n$ matrix with entries in R represents a linearly independent transform from R^n to R^m . If we write vectors $x = \begin{bmatrix} x_1, \dots, x_n \end{bmatrix}^T$ and $y = \begin{bmatrix} y_1, \dots, y_n \end{bmatrix}^T$ in R^n as column matrices, then an $m \times n$ matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

maps the vectors to R^m linearly by matrix multiplication. If λ is a real number,

$$\begin{aligned} A(x + y) &= Ax + Ay \\ \lambda Ax &= A(\lambda x) \end{aligned}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

Conversely, if $T : R^n \rightarrow R^m$ is a linear map, we can associate a matrix A with T by considering what T does to the vectors

$$\begin{aligned} e_1 &= \begin{bmatrix} 1, & 0, & \dots, & 0 \end{bmatrix}^T \\ e_2 &= \begin{bmatrix} 0, & 1, & \dots, & 0 \end{bmatrix}^T \\ &\vdots \\ e_n &= \begin{bmatrix} 0, & 0, & \dots, & 1 \end{bmatrix}^T. \end{aligned}$$

We can write any vector $x = \begin{bmatrix} x_1, & \dots, & x_n \end{bmatrix}^T$ as

$$x_1 e_1 + x_2 e_2 + \dots + x_n e_n.$$

Consequently, if

$$\begin{aligned} T(e_1) &= \begin{bmatrix} a_{11}, & a_{21}, & \dots, & a_{m1}, \end{bmatrix}^T \\ T(e_2) &= \begin{bmatrix} a_{12}, & a_{22}, & \dots, & a_{m2}, \end{bmatrix}^T \\ &\vdots \\ T(e_n) &= \begin{bmatrix} a_{1n}, & a_{2n}, & \dots, & a_{mn}, \end{bmatrix}^T, \end{aligned}$$

then

$$\begin{aligned}
 T(x) &= T(x_1e_1 + x_2e_2 + \cdots + x_ne_n) \\
 &= x_1T(e_1) + x_2T(e_2) + \cdots + x_nT(e_n) \\
 &= \left[\sum_{k=1}^n a_{1k}x_k, \quad \dots, \quad \sum_{k=1}^n a_{mk}x_k \right]^T \\
 &= Ax.
 \end{aligned}$$

An $n \times n$ matrix A is *nonsingular* (*invertible*) exactly when there exists another matrix A^{-1} such that $A \cdot A^{-1} = A^{-1} \cdot A = I$, where

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

is the $n \times n$ identity matrix.

2.6 Reed-Muller and arithmetic transforms

The Reed-Muller canonical form can be extended to multiple-valued logic in several ways, depending on how its operations are generalized. In a q -valued logic system, the Reed-Muller expansion of an n -variable function is written as

$$f(x_{n-1}, \dots, x_0) = \sum_{i=0}^{q^n-1} b_i \prod_{j=n-1}^0 x_j^{i_j}. \quad (2.18)$$

In the above equation, the summation index i is a q -ary number containing n digits. The term $x_j^{i_j}$ denotes the i_j -th power of the variable x_j and $b_i \in GF(q)$. In this Reed-Muller expansion, all the operations are performed over $GF(q)$.

Ternary Reed-Muller transform. The ternary Reed-Muller (TRM) transform and its inverse transform are given by

$$TRM_1 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix}, \quad (2.19)$$

and

$$TRM_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 2 & 2 & 2 \end{bmatrix}. \quad (2.20)$$

For arbitrary n ,

$$TRM_n = \overset{n-1}{\otimes} TRM_1 = \overset{n-1}{\otimes} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix}, \quad (2.21)$$

and

$$TRM_n^{-1} = \overset{n-1}{\otimes} TRM_1^{-1} = \overset{n-1}{\otimes} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 2 & 2 & 2 \end{bmatrix}, \quad (2.22)$$

where " $\overset{n-1}{\otimes}$ " represents Kronecker product applied $n - 1$ times to either the matrix TRM_1 or TRM_1^{-1} with additions and multiplications performed over $GF(3)$.

Quaternary Reed-Muller transform. When $q = 4$, the quaternary Reed-Muller (QRM) transform and its inverse transform matrix are given by

$$QRM_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 3 & 2 & 1 \end{bmatrix} \quad (2.23)$$

and

$$QRM_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 3 & 2 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix}. \quad (2.24)$$

For arbitrary n ,

$$QRM_n = \overset{n-1}{\otimes} QRM_1, \quad (2.25)$$

and

$$QRM_n^{-1} = \overset{n-1}{\otimes} QRM_1^{-1}. \quad (2.26)$$

where the additions and multiplications are performed over GF(4).

Generalized Reed-Muller expansions. In the binary case, the Reed-Muller expansion can be generalized into 2^n expansions each with a different fixed polarity by replacing any combination of the n variables by their inverses. As $\bar{x}_i = x_i + 1$ over GF(2), this can be regarded as an additive modification to the variable. Over GF(3), two additive modifications are possible, $\dot{x}_i = x_i + 1$, and $\ddot{x}_i = x_i + 2$. Together with the unmodified form these enable 3^n fixed polarity

forms to be set up. Over $\text{GF}(4)$, we have the possibility of three modifications which are denoted as $\dot{x}_i = x_i + 1$, $\widehat{x}_i = x_i + 2$ and $\widetilde{x}_i = x_i + 3$. The four literal forms that each variable may take enable 4^n fixed polarity forms to be derived.

Generally, for the Reed-Muller transform defined by $\text{GF}(q)$, when $n = 1$,

$$T_1^{(k)} = \begin{bmatrix} 1, & (x+k), & (x+k)^2, & \dots, & (x+k)^{q-1} \end{bmatrix} \quad (2.27)$$

with $k \in \left(0, 1, 2, \dots, q-1 \right)$, i.e., when $q = 4$ while x takes the value from its four literal forms,

$$T_1^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 3 & 2 & 1 \end{bmatrix}; \text{ and } T_1^{(1)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 3 & 2 & 1 \\ 1 & 2 & 3 & 1 \end{bmatrix};$$

$$T_1^{(2)} = \begin{bmatrix} 1 & 2 & 3 & 1 \\ 1 & 3 & 2 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}; \text{ and } T_1^{(3)} = \begin{bmatrix} 1 & 3 & 2 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Arithmetic transform. The arithmetic transform is known as integer-valued Reed-Muller polynomial. The generalized arithmetic transform takes the form given in (2.18) while it is performed over standard algebra.

The ternary arithmetic (TAR) transform has the same forward matrix as (2.19) when $n = 1$ and (2.21) for arbitrary n . The inverse matrix is defined by

$$TAR_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ -1 & 2 & -1 \end{bmatrix}, \quad (2.28)$$

and for arbitrary n ,

$$TAR_n^{-1} = \overset{n-1}{\otimes} TAR_1^{-1} = \overset{n-1}{\otimes} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ -1 & 2 & -1 \end{bmatrix}. \quad (2.29)$$

The quaternary arithmetic (QA) transform also takes the same forward matrix as shown in (2.23) when $n = 1$ and (2.25) for arbitrary n . The inverse transform matrix of QA is given by

$$3QA_1^{-1} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & -1 & -1 & 2 \\ 0 & -1 & 2 & -1 \\ -3 & 5 & -1 & -1 \end{bmatrix}, \quad (2.30)$$

and for arbitrary n ,

$$QA_n^{-1} = \overset{n-1}{\otimes} QA_1^{-1}. \quad (2.31)$$

where the additions and multiplications are performed over standard quaternary algebra.

The generalized multi-polarity arithmetic transform are also defined based

on (2.27), i.e., when $q = 4$, if $(QA_n^{-1})^{(k)} = S_n^{(k)}$, then when $n = 1$

$$3S_1^{(0)} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & -1 & -1 & 2 \\ 0 & -1 & 2 & -1 \\ -3 & 5 & -1 & -1 \end{bmatrix}; \text{ and } 3S_1^{(1)} = \begin{bmatrix} 0 & 3 & 0 & 0 \\ -1 & 0 & 2 & -1 \\ -1 & 0 & -1 & 2 \\ 5 & -3 & -1 & -1 \end{bmatrix};$$

$$3S_1^{(2)} = \begin{bmatrix} 0 & 0 & 3 & 0 \\ -1 & 2 & 0 & -1 \\ 2 & -1 & 0 & -1 \\ -1 & -1 & -3 & 5 \end{bmatrix}; \text{ and } 3S_1^{(3)} = \begin{bmatrix} 0 & 0 & 0 & 3 \\ 2 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 5 & -3 \end{bmatrix}$$

Chapter 3

Families of Linearly Independent Transforms over $GF(3)$

In Chapter 3, the concepts of linearly independent binary logic and transforms have been initially extended to Galois Field (GF) (3). Three families of novel introduced linearly independent transforms are discussed in details for their properties, mutual relations and computational methods. These transforms are based on three different structures, $F - R$ type (first introduced by Falkowski and Rahardja in [FR97] for binary case), F^2 type (Falkowski and Fu in [FF]) and helix transforms (Fu and Falkowski in [FF04]). The family of fastest linearly independent transforms over $GF(3)$ is also presented in this chapter. To show the advantages of presented family of ternary linearly independent transforms a number of known binary benchmark functions from Microelectronics Center of North Carolina (MCNC), International Workshop for Logic Synthesis 93 (IWLS'93) and International Symposium on Circuits and Systems 95 (IS-

CAS'95) were converted to ternary benchmark functions and used to calculate ternary linearly independent spectra as well as ternary Reed-Muller transform.

3.1 Basic definitions for LI transforms over GF(3)

Definition 3.1.1. Let M_n be a $N \times N$ ($N = 3^n$) matrix with columns corresponding to some switching ternary functions of n variables. If the sets of columns are linearly independent with respect to *ternary Galois Field*, then M_n has only one inverse over GF(3) and is said to be *ternary linearly independent*. If M_n is linearly independent over GF(3), then M_n is a nonsingular square matrix and has a unique inverse M_n^{-1} with respect to GF(3).

Definition 3.1.2. The linearly independent polynomial expansion for any n -variable ternary function $f(\vec{x}_n)$ is

$$f(\vec{x}_n) = \sum_{j=0}^{3^n-1} A_j \vec{g}_j, \quad (3.1)$$

where g_j denotes a ternary discrete function, which truth vector constitutes the number $j + 1$ column of the LI matrix M_n , $0 \leq j \leq 3^n - 1$ and all operations are over GF(3).

Definition 3.1.3. The ternary linearly independent transform based on Definitions 3.1.1 and 3.1.2 can be described by the following general formulae performed over GF(3):

$$\vec{F} = M_n \vec{A} \quad (3.2)$$

and

$$\vec{A} = M_n^{-1} \vec{F}, \quad (3.3)$$

where $\vec{F} = [F_0, F_1, \dots, F_{3^n-1}]^T$ is a column vector defining the truth vector of a ternary function $f(\vec{x}_n)$ in a natural ternary ordering; M_n is a linearly independent ternary matrix of order $N = 3^n$ defined by any linearly independent set of n -variable ternary functions and $\vec{A} = [A_0, A_1, \dots, A_{3^n-1}]^T$ is the spectral coefficient column vector for the particular transform matrix M_n .

Definition 3.1.4. Let M_n be a matrix as specified in Definition 3.1.1. Furthermore, when $n \geq 1$ let M_n be defined by

$$M_n = \begin{bmatrix} M_{n-1}^{(1)} & M_{n-1}^{(2)} & M_{n-1}^{(3)} \\ M_{n-1}^{(4)} & M_{n-1}^{(5)} & M_{n-1}^{(6)} \\ M_{n-1}^{(7)} & M_{n-1}^{(8)} & M_{n-1}^{(9)} \end{bmatrix} \quad (3.4)$$

where each submatrix $M_{n-1}^{(j)}$, $j = (1, 2, 3, \dots, 8, 9)$ has a dimension of $3^{n-1} \times 3^{n-1}$, and contains one recursive equation which is either O_{n-1} , X_{n-1} , Y_{n-1} or M_{n-1} , where O_{n-1} is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements being zero. $X_{n-1} = I_{n-1}$ or $X_{n-1} = J_{n-1}$, I_{n-1} is a $3^{n-1} \times 3^{n-1}$ identity submatrix and J_{n-1} is a $3^{n-1} \times 3^{n-1}$ reverse-identity submatrix. Y_{n-1} is also introduced here, which is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements being zero except one element located at one corner of each matrix depending on the location of Y_{n-1} .

The matrix M_n is a $3^n \times 3^n$ square matrix that can be partitioned into three vertical submatrices, each with dimension $3^n \times 3^{n-1}$,

$$M_n = \begin{bmatrix} M_V^{(1)} & M_V^{(2)} & M_V^{(3)} \end{bmatrix}, \quad (3.5)$$

or partitioned into three horizontal submatrices, each with dimension $3^{n-1} \times 3^n$,

$$M_n = \begin{bmatrix} M_H^{(1)} \\ M_H^{(2)} \\ M_H^{(3)} \end{bmatrix}, \quad (3.6)$$

where the subscripts V or H denote the respective vertical or horizontal partitioning of the original matrix M_n .

Definition 3.1.5. Let M_n be a nonsingular square $3^n \times 3^n$ matrix as described by (3.5). The μ_H operator on M_n is defined as interchanging the left and right $3^n \times 3^{n-1}$ submatrices as

$$\mu_H[M_n] = \begin{bmatrix} M_V^{(3)} & M_V^{(2)} & M_V^{(1)} \end{bmatrix}. \quad (3.7)$$

Definition 3.1.6. Let M_n be a nonsingular square $3^n \times 3^n$ matrix as described by (3.6). The μ_V operator on M_n is defined as interchanging the top and bottom $3^{n-1} \times 3^n$ submatrices as

$$\mu_V[M_n] = \begin{bmatrix} M_H^{(3)} \\ M_H^{(2)} \\ M_H^{(1)} \end{bmatrix}. \quad (3.8)$$

Let M_n be a $3^n \times 3^n$ nonsingular square matrix that is partitioned into appropriate $3^{n-1} \times 3^{n-1}$ dimensional submatrices, as recursively defined in Definition 3.1.4.

Definition 3.1.7. The μ_{EH} operator on M_n is defined as grouping the recursive equations in the submatrices vertically and interchanging them in the subma-

trices horizontally

$$\mu_{EH}[M_n] = \begin{bmatrix} M_{n-1}^{(3)} & M_{n-1}^{(2)} & M_{n-1}^{(1)} \\ M_{n-1}^{(6)} & M_{n-1}^{(5)} & M_{n-1}^{(4)} \\ M_{n-1}^{(9)} & M_{n-1}^{(8)} & M_{n-1}^{(7)} \end{bmatrix}. \quad (3.9)$$

Definition 3.1.8. The μ_{EV} operator on M_n is defined as grouping the recursive equations in the submatrices horizontally and interchanging them in the submatrices vertically

$$\mu_{EV}[M_n] = \begin{bmatrix} M_{n-1}^{(7)} & M_{n-1}^{(8)} & M_{n-1}^{(9)} \\ M_{n-1}^{(4)} & M_{n-1}^{(5)} & M_{n-1}^{(6)} \\ M_{n-1}^{(1)} & M_{n-1}^{(2)} & M_{n-1}^{(3)} \end{bmatrix}. \quad (3.10)$$

Definition 3.1.9. The α_0 operator on the matrix M_n is defined as interchanging the diagonal submatrices

$$\alpha_0[M_n] = \begin{bmatrix} M_{n-1}^{(9)} & M_{n-1}^{(6)} & M_{n-1}^{(3)} \\ M_{n-1}^{(8)} & M_{n-1}^{(5)} & M_{n-1}^{(2)} \\ M_{n-1}^{(7)} & M_{n-1}^{(4)} & M_{n-1}^{(1)} \end{bmatrix}. \quad (3.11)$$

Definition 3.1.10. The α_1 operator on the matrix M_n is defined as interchanging the reverse-diagonal submatrices

$$\alpha_1[M_n] = \begin{bmatrix} M_{n-1}^{(1)} & M_{n-1}^{(4)} & M_{n-1}^{(7)} \\ M_{n-1}^{(2)} & M_{n-1}^{(5)} & M_{n-1}^{(8)} \\ M_{n-1}^{(3)} & M_{n-1}^{(6)} & M_{n-1}^{(9)} \end{bmatrix}. \quad (3.12)$$

Definition 3.1.11. If $M_{n-1}^{(j)} = O_{n-1}$, $j \in (1, 3, 7, 9)$, then the operator

β_A on M_n is defined as selecting the submatrix $M_{n-1}^{(10-j)}$, which is opposite to $M_{n-1}^{(j)}$, and interchanging it by the following operations:

- $X_{n-1} \rightarrow 2M_{n-1}$ if M_{n-1} and X_{n-1} are submatrices in $M_{n-1}^{(j)}$, or
- $M_{n-1} \rightarrow 2X_{n-1}$ if X_{n-1} is a submatrix in $M_{n-1}^{(j)}$,

Example 3.1.1. Let M_n be defined by Definition 3.1.4, and the example matrix is shown below:

$$M_n = \begin{bmatrix} J_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ J_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix},$$

then by using the first operation according to Definition 3.1.11,

$$\beta_A[M_n] = \begin{bmatrix} J_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ 2M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}.$$

□

Definition 3.1.12. Let M_n be a $3^n \times 3^n$ nonsingular square matrix that is partitioned into nine submatrices with five non-zero slots as described by (3.4), then the operator β_B on M_n is defined as multiplying 2 over all the non-zero submatrices of M_n , except the submatrix X_{n-1} .

Definition 3.1.13. Let M_n be a $3^n \times 3^n$ nonsingular square matrix that is partitioned into nine submatrices with five non-zero slots as described by (3.4), then the rotation operator R on M_n is defined as $R/4$ clockwise rotation on the nine $3^{n-1} \times 3^{n-1}$ submatrices.

3.2 F – R type fast LI transforms over GF(3)

In this section, properties of various recursive classes for LI transforms over GF(3), which possesses regular fast forward and inverse butterfly diagrams, are analyzed. They are different than the known multiple-valued transforms for ternary case with some properties of generating of their matrices that may be used for matching and checking similarities between different ternary functions. Since the recursive equations for expanding the transforms matrices are provided, then the presented results can be easily extended to higher dimensions. Finally, LI transform matrices that require permutations for both forward and inverse butterflies diagrams are also shown.

3.2.1 Classification of fast LI transforms

Altogether 132 pairs of forward and inverse ternary LI transforms that do not require any horizontal or vertical permutations are introduced in this section. They are recursively defined and constructed by three basic submatrices O_{n-1} , M_{n-1} , and X_{n-1} . There are four zero submatrices O_{n-1} located in the slots $M_{n-1}^{(i)}$, $i = (2, 4, 6, 8)$ as well as at most one O_{n-1} and at least one M_{n-1} could appear in other five slots $M_{n-1}^{(j)}$, $j = (1, 3, 5, 7, 9)$. Class A includes eighteen pairs of forward and inverse transform matrices, which have exactly the same shapes of forward and inverse butterfly diagrams except some differences on elements' values. Class A is divided into three subclasses depending on the different component submatrices in the four corners of the LI transform matrices. In this class, the submatrices do not have reverse-identity matrix J_{n-1} in any of the four corners. Only one submatrix O_{n-1} lies in either submatrix

$M_{n-1}^{(3)}$ or submatrix $M_{n-1}^{(7)}$ of the general matrix and $M_{n-1}^{(1)} = M_{n-1}^{(9)}$.

Let M_n be a nonsingular $3^n \times 3^n$ square matrix with one zero submatrix O_{n-1} in the location $M_{n-1}^{(3)}$, one submatrix M_{n-1} in $M_{n-1}^{(7)}$, and two submatrices I_{n-1} in locations $M_{n-1}^{(1)}$ and $M_{n-1}^{(9)}$, and the central submatrix $M_{n-1}^{(5)}$ can be filled by M_{n-1} , I_{n-1} or J_{n-1} . Hence there are three transforms that can be presented by an unified equation as

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{(5)} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}. \quad (3.13)$$

One half of the first sub-class in Class A can be derived from (3.13). Their inverse matrices are given by

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & [M_{n-1}^{(5)}]^{-1} & O_{n-1} \\ 2M_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}. \quad (3.14)$$

From (3.13), replacing the two identity submatrices in slots $M_{n-1}^{(1)}$ and $M_{n-1}^{(9)}$ by submatrices M_{n-1} , three forward transform matrices in the second sub-class of Class A are defined. Their inverse transform matrices can be derived from the forward transform matrices by multiplying 2 over the submatrix $M_{n-1}^{(7)}$ recursively from $n = 1$ to $n - 1$. The following two equations (3.15) and (3.16) present the recursive definitions of the forward and inverse transform matrices,

respectively.

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{(5)} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \quad (3.15)$$

and

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & [M_{n-1}^{(5)}]^{-1} & O_{n-1} \\ 2M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (3.16)$$

From (3.15), replacing the submatrix M_{n-1} in the slot $M_{n-1}^{(7)}$ by I_{n-1} , three transform matrices in the last sub-class of Class A are defined in (3.17), the recursive definition of the inverse transforms is also given in (3.18), as shown below.

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{(5)} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \quad (3.17)$$

and

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & [M_{n-1}^{(5)}]^{-1} & O_{n-1} \\ 2[M_{n-1}^{-1}]^2 & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (3.18)$$

There are also other nine pairs of the forward and inverse transform matrices in Class A. They can be derived from (3.13) to (3.18) by interchanging the submatrices $M_{n-1}^{(3)}$ and $M_{n-1}^{(7)}$. Table 3.1 shows all forward and inverse transforms in Class A and their corresponding butterfly diagrams for $n = 2$. The narrow and broad lines represent the value 1 and 2, respectively. It should be noticed, that some parts of the butterfly diagrams can be removed as they do not require any calculation and the reason why we keep them in the butterfly diagrams is

to find easier relations between transforms from different classes.

A symbolic calculation can be used to derive the general inverse matrix for Class A. For a forward matrix,

$$M_n = \begin{bmatrix} M_{n-1}^{(1)} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{(5)} & O_{n-1} \\ M_{n-1}^{(7)} & O_{n-1} & M_{n-1}^{(9)} \end{bmatrix}, \quad (3.19)$$

its inverse matrix is given by

$$M_{n-1}^{-1} = \begin{bmatrix} \left(M_{n-1}^{(1)}\right)^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & \left(M_{n-1}^{(5)}\right)^{-1} & O_{n-1} \\ Q & O_{n-1} & \left(M_{n-1}^{(9)}\right)^{-1} \end{bmatrix}, \quad (3.20)$$

where $Q = 2M_{n-1}^{(7)} \cdot \left(M_{n-1}^{(1)}\right)^{-1} \cdot \left(M_{n-1}^{(9)}\right)^{-1}$. This general inverse can be used as the proof to derive the inverse transforms for all classes in this chapter.

For each forward transform matrix in Class A, grouping the recursive equations in the submatrices vertically and interchanging them in the submatrices horizontally will bring the ternary LI transform matrices from Class A into Class B. In Class B, the submatrices do not have any identity matrix I_{n-1} located in any of the four corners, and the submatrices $M_{n-1}^{(3)} = M_{n-1}^{(7)}$. Their inverse transform matrices have similar relations to Class A as their forward transform matrices as shown in the following Properties 3.2.1 and 3.2.2.

Property 3.2.1. For any forward matrix M_A in Class A, a forward matrix M_B can be found in Class B and the following relations exist:

$$\mu_{EH}[M_A] = M_B \quad (3.21)$$

or

$$\mu_{EV} [M_A] = M_B. \quad (3.22)$$

Property 3.2.2. Let $\widetilde{M}_A$ be one of the square forward transform matrices in Class A, and $\widetilde{M}_B$ be one of the square forward transform matrices in Class B. Moreover, let $\widetilde{M}_A^{-1}$ and $\widetilde{M}_B^{-1}$ represent their corresponding inverse transform matrices.

From Property 3.2.1, if $\widetilde{M}_B = \mu_{EH} [\widetilde{M}_A]$, then

$$\widetilde{M}_B^{-1} = \mu_{EV} [\widetilde{M}_A^{-1}], \quad (3.23)$$

or if $\widetilde{M}_B = \mu_{EV} [\widetilde{M}_A]$, then

$$\widetilde{M}_B^{-1} = \mu_{EH} [\widetilde{M}_A^{-1}], \quad (3.24)$$

Table 3.2 shows the list of all forward and inverse transform matrices belonging to Class B.

Different combinations and permutations of identity matrix I_{n-1} or reverse-identity matrix J_{n-1} with M_{n-1} define some other classes in the family of ternary LI transforms. Class C includes 48 pairs of the forward and inverse ternary LI transform matrices. All the transform matrices in Class C are shown in Tables 3.3 and 3.4 with their corresponding butterfly diagrams for $n = 2$. For the transform matrices in Class C, the zero submatrix O_{n-1} lies in either $M_{n-1}^{(3)}$ or $M_{n-1}^{(7)}$, and their forward and inverse transform matrices are related by the following property.

Property 3.2.3. Let M_C be one of the $3^n \times 3^n$ square transform matrices in

3.2. $F - R$ type fast LI transforms over $GF(3)$

45

Table 3.1: Class A of LI transforms over $GF(3)$.

	$\begin{bmatrix} I_{n-1} & & \\ & M_{n-1} & \\ M_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & \\ & M_{n-1}^1 & \\ 2M_{n-1} & & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & & \\ & M_{n-1} & \\ I_{n-1} & & 2M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & 2M_{n-1} \\ & M_{n-1}^1 & \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & \\ & I_{n-1} & \\ 2M_{n-1} & & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & & \\ & M_{n-1} & \\ I_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & 2M_{n-1} \\ & I_{n-1} & \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & & \\ & J_{n-1} & \\ M_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & \\ & J_{n-1} & \\ 2M_{n-1} & & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & & \\ & M_{n-1} & \\ I_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & & 2M_{n-1} \\ & J_{n-1} & \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & M_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & M_{n-1}^1 & \\ 2M_{n-1}^1 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & M_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2M_{n-1}^1 \\ & M_{n-1}^1 & \\ & & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & I_{n-1} & \\ 2M_{n-1}^1 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & M_{n-1} & \\ I_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2M_{n-1}^1 \\ & I_{n-1} & \\ & & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & J_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & J_{n-1} & \\ 2M_{n-1}^1 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & M_{n-1} & \\ J_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2M_{n-1}^1 \\ & J_{n-1} & \\ & & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & M_{n-1} & \\ I_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & M_{n-1}^1 & \\ 2(M_{n-1}^1)^2 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2(M_{n-1}^1)^2 \\ & M_{n-1}^1 & \\ & & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & I_{n-1} & \\ I_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & I_{n-1} & \\ 2(M_{n-1}^1)^2 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2(M_{n-1}^1)^2 \\ & I_{n-1} & \\ & & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & & \\ & J_{n-1} & \\ I_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & \\ & J_{n-1} & \\ 2(M_{n-1}^1)^2 & & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & & \\ & I_{n-1} & \\ J_{n-1} & & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & & 2(M_{n-1}^1)^2 \\ & J_{n-1} & \\ & & M_{n-1}^1 \end{bmatrix}$	

Class C. Then, the inverse transform matrix of M_C is given by

$$M_C^{-1} = \beta_A [M_C], \quad (3.25)$$

and after application of β_A on M_C , all the submatrices M_{n-1} inside the inverse of M_C should be changed to M_{n-1}^{-1} .

Similar to the relations between Classes A and B, by shifting the zero submatrix O_{n-1} into either $M_{n-1}^{(1)}$ or $M_{n-1}^{(9)}$, and keeping the respective combinations and permutations of identity matrix I_{n-1} or reverse-identity matrix J_{n-1} with M_{n-1} from Class C, another class denoted as Class D will be obtained. The following properties of the forward and inverse transform matrices as well as the relations between Classes C and D may be derived.

Property 3.2.4. Let M_D be one of the $3^n \times 3^n$ square matrices in Class D. Then, the inverse of M_D is given by

$$M_D^{-1} = \alpha_0 \alpha_1 (\beta_A [M_D]), \quad (3.26)$$

and after application of β_A on M_D , all the submatrices M_{n-1} inside the inverse of M_D should be changed to M_{n-1}^{-1} .

Property 3.2.5. For any forward matrix M_C in Class C, a forward matrix M_D can be found in Class D and the following relations exist:

$$\mu_{EH} [M_C] = M_D \quad (3.27)$$

or

$$\mu_{EV} [M_C] = M_D. \quad (3.28)$$

Table 3.2: Class B of LI transforms over $GF(3)$.

	$\begin{bmatrix} & J_{n-1} \\ M_{n-1} & \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1} & J_{n-1} \\ J_{n-1} & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & J_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & J_{n-1} \\ J_{n-1} & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & J_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1} & J_{n-1} \\ J_{n-1} & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & J_{n-1} \\ J_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & J_{n-1} \\ J_{n-1} & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & J_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1} & J_{n-1} \\ J_{n-1} & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & J_{n-1} \\ J_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & J_{n-1} \\ J_{n-1} & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^1 & M_{n-1}^1 \\ M_{n-1}^1 & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^1 & M_{n-1}^1 \\ M_{n-1}^1 & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^1 & M_{n-1}^1 \\ M_{n-1}^1 & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2(M_{n-1}^1)^2 & M_{n-1}^1 \\ M_{n-1}^1 & M_{n-1}^1 \end{bmatrix}$			$\begin{bmatrix} J_{n-1} & M_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2(M_{n-1}^1)^2 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ J_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2(M_{n-1}^1)^2 & M_{n-1}^1 \\ M_{n-1}^1 & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} J_{n-1} & M_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2(M_{n-1}^1)^2 \end{bmatrix}$	
	$\begin{bmatrix} & M_{n-1} \\ I_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2(M_{n-1}^1)^2 & M_{n-1}^1 \\ M_{n-1}^1 & I_{n-1} \end{bmatrix}$			$\begin{bmatrix} J_{n-1} & M_{n-1} \\ M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} & M_{n-1}^1 \\ M_{n-1}^1 & 2(M_{n-1}^1)^2 \end{bmatrix}$	

Table 3.3: First part of Class C of LI transforms over $GF(3)$.

	$\begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & \\ \beta & J_{n-1} \\ 2M_{n-1}^t & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & & \\ & I_{n-1} & \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & \\ \beta & I_{n-1} \\ 2M_{n-1}^t & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & \\ \beta & J_{n-1} \\ 2J_{n-1} & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & \\ \beta & I_{n-1} \\ 2J_{n-1} & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & J_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2M_{n-1}^t \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & I_{n-1} \\ & I_{n-1} \\ & & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2M_{n-1}^t \\ & I_{n-1} \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & M_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2J_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ & I_{n-1} \\ & & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2J_{n-1} \\ & I_{n-1} \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & J_{n-1} \\ & J_{n-1} \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & 2M_{n-1}^t \\ & J_{n-1} \\ & & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & I_{n-1} \\ & I_{n-1} \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & 2M_{n-1}^t \\ & I_{n-1} \\ & & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & M_{n-1} \\ & J_{n-1} \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & 2J_{n-1} \\ & J_{n-1} \\ & & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & M_{n-1} \\ & I_{n-1} \\ & & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & 2J_{n-1} \\ & I_{n-1} \\ & & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & \\ & J_{n-1} \\ J_{n-1} & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t \\ & J_{n-1} \\ 2M_{n-1}^t & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & \\ & I_{n-1} \\ I_{n-1} & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t \\ & I_{n-1} \\ 2M_{n-1}^t & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & \\ & J_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t \\ & J_{n-1} \\ 2J_{n-1} & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & \\ & I_{n-1} \\ M_{n-1} & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t \\ & I_{n-1} \\ 2J_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & \\ & I_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & \\ \beta & I_{n-1} \\ 2M_{n-1}^t & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & \\ & J_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & \\ \beta & J_{n-1} \\ 2M_{n-1}^t & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & \\ & I_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & \\ \beta & I_{n-1} \\ 2J_{n-1} & M_{n-1}^t \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & \\ & J_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & \\ \beta & J_{n-1} \\ 2J_{n-1} & M_{n-1}^t \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & J_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2M_{n-1}^t \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & I_{n-1} \\ & J_{n-1} \\ & & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2M_{n-1}^t \\ & J_{n-1} \\ & & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & M_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2J_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ & J_{n-1} \\ & & I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1}^t & 2J_{n-1} \\ & J_{n-1} \\ & & I_{n-1} \end{bmatrix}$	

The μ_{EH} and μ_{EV} operations on the matrix are one to one and onto mapping. Hence, there are also 48 pairs of forward and inverse ternary LI transform matrices in Class D. Tables 3.5 and 3.6 presents all forward and inverse ternary LI transform matrices in Class D and their corresponding butterfly diagrams for $n = 2$. The following mathematical relationship may be derived.

Property 3.2.6. Let M_C and M_D represent any $3^n \times 3^n$ square matrices in Classes C and D, respectively. If $M_D = \mu_{EH} [M_C]$ or $M_D = \mu_{EV} [M_C]$, then from Properties 3.2.3 – 3.2.5,

$$M_D^{-1} = \mu_{EV} [M_C^{-1}] \quad (3.29)$$

or

$$M_D^{-1} = \mu_{EH} [M_C^{-1}], \quad (3.30)$$

respectively. It should be noticed that (3.29) and (3.30) apply to all 48 transform matrices in Class C but their result overlap i.e. Class D has also 48 transform matrices.

All the four classes, Classes A, B, C and D of ternary LI logic form the basic ternary LI transforms which possess fast algorithms in forward and inverse transform matrices. All the basic ternary LI transforms are classified according to their mathematical relationship of forward transform matrices with their respective inverse transform matrices. They are constructed through four basic submatrices, zero submatrix O_{n-1} , identity submatrix I_{n-1} , reverse-identity submatrix J_{n-1} , and recursive submatrix M_{n-1} , in which general rules applies to them, according to their classifications.

The following example shows applications of some transforms from various

Table 3.4: Second part of Class C of LI transforms over $GF(3)$.

	$\begin{bmatrix} J_{n-1} & J_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & 2M_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & I_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & 2M_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & M_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} & 2J_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} & M_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} & 2J_{n-1} \\ J_{n-1} & M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ I_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ I_{n-1} \\ 2M_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ J_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ J_{n-1} \\ 2M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ I_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ I_{n-1} \\ 2J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ J_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ J_{n-1} \\ 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} \\ M_{n-1} \\ 2M_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} \\ M_{n-1} \\ 2M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} \\ M_{n-1} \\ 2J_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} \\ M_{n-1} \\ 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ J_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ 2M_{n-1} \\ J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ 2J_{n-1} \\ J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ 2I_{n-1} \\ I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} J_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix}$			$\begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix} \begin{bmatrix} I_{n-1} \\ M_{n-1} \\ M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ M_{n-1} \\ J_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix} \begin{bmatrix} M_{n-1} \\ M_{n-1} \\ I_{n-1} \end{bmatrix}$	

classes to calculate spectra for a ternary LI switching function.

Example 3.2.1. Let the ternary switching function be $\vec{F} = \begin{bmatrix} 0 & 1 & 0 & 2 & 2 & 1 & 0 & 1 & 2 \end{bmatrix}^T$. The forward transform matrix of one ternary LI transform matrix M_n in Class A is defined by (3.15) as shown below:

$$M_A = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}.$$

For the order $n = 2$, and from (3.16),

$$\vec{A}_A = M_2^{-1} \vec{F} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 0 \\ 2 \end{bmatrix}.$$

By Definition 3.1.2,

$$\begin{aligned} f &= A_1 \cdot g_1 + A_3 \cdot g_3 + A_4 \cdot g_4 + A_5 \cdot g_5 + A_8 \cdot g_8 \\ &= g_1 + 2 \cdot g_3 + 2 \cdot g_4 + g_5 + 2 \cdot g_8, \end{aligned}$$

where the truth vectors of g_i , $i = (1, 3, 4, 5, 8)$, are obtained directly from columns 1, 3, 4, 5, and 8 in the forward transformation matrix M_A .

Table 3.5: First part of Class D of LI transforms over $\text{GF}(3)$.

[illegible]

Figure 3.1: Forward butterfly diagram for the LI transform from Class A when $n = 3$.

Fig. 3.1 shows the butterfly diagram of the forward transform, and the inverse butterfly diagram has exactly the same shape as Fig. 3.1 except that the value of the crossing lines is 2.

Let us apply μ_{EV} operation to a selected transform M_C from Class C to bring it into Class D as

$$M_C = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}$$

and

$$M_D = \mu_{EV} [M_C] = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}.$$

From Property 3.2.3 and by (3.3)

$$\begin{aligned}
 \vec{A}_C &= M_C^{-1} \cdot \vec{F} = \beta_A [M_C] \cdot \vec{F} = \beta_A \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \cdot \vec{F} \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 0 \\ 2 \end{bmatrix} .
 \end{aligned}$$

From Property 3.2.6,

$$\begin{aligned} \vec{A}_D = M_D^{-1} \cdot \vec{F} &= \mu_{EH} [M_C^{-1}] \cdot \vec{F} = \mu_{EH} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix} \cdot \vec{F} \\ &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 1 \\ 2 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}. \end{aligned}$$

For the transform matrix from Class B given by

$$M_B = \mu_{EV} [M_A] = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix},$$

the resulting ternary LI spectrum is

$$\vec{A}_B = \begin{bmatrix} 2 & 0 & 0 & 1 & 2 & 2 & 0 & 1 & 0 \end{bmatrix}^T.$$

Comparing $\vec{A}_A$, $\vec{A}_B$, $\vec{A}_C$ and $\vec{A}_D$, it can be seen that the transform matrices from Classes A, B and C all provide better spectral results that contain more spectral zero-elements than the original ternary switching function and hence can be implemented by the final ternary LI expansion based on (3.1). Even the worst one, the spectrum obtained from the transform matrix in Class D, also keeps the same number of zeros as the original ternary function. $\square$

Besides the four ternary LI transform Classes A, B, C and D, there exist other classes in which horizontal or vertical permutations are required before making the matrices recursive. The transform matrices of the new category can be derived from the four introduced classes.

Property 3.2.7. Let M_n be a $3^n \times 3^n$ square matrix with inverse M_n^{-1} over $GF(3)$, then

$$[\mu_H(M_n)]^{-1} = \mu_V(M_n^{-1}) \quad (3.31)$$

and

$$[\mu_V(M_n)]^{-1} = \mu_H(M_n^{-1}). \quad (3.32)$$

Definitions 3.1.5 and 3.1.6 transfer all the basic classes of ternary LI transforms into two difference categories: one of which the horizontal permutations are needed to be done in the forward transform; the other one where the vertical permutations are needed to be performed instead.

Property 3.2.8. Let M_Y be any matrix belonging to Class Y of ternary LI

Table 3.6: Second part of Class D of LI transforms over $GF(3)$.

	$\begin{bmatrix} I_{n+1} & I_{n+1} \\ J_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & \\ J_{n+1} & 2M_{k+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} J_{n+1} & J_{n+1} \\ I_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & \\ I_{n+1} & 2M_{k+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} & I_{n+1} \\ J_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & \\ J_{n+1} & 2I_{n+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & J_{n+1} \\ I_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & \\ I_{n+1} & 2J_{n+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} \\ J_{n+1} & I_{n+1} \\ I_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & I_{n+1} \\ M_{k+1} & J_{n+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} \\ I_{n+1} & J_{n+1} \\ J_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & J_{n+1} \\ M_{k+1} & I_{n+1} \\ M_{k+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} \\ J_{n+1} & M_{k+1} \\ I_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2I_{n+1} & I_{n+1} \\ M_{k+1} & J_{n+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} \\ I_{n+1} & M_{k+1} \\ J_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2J_{n+1} & J_{n+1} \\ M_{k+1} & I_{n+1} \\ M_{k+1} & \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} \\ M_{k+1} & I_{n+1} \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & M_{k+1} \\ M_{k+1} & I_{n+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} J_{n+1} \\ M_{k+1} & J_{n+1} \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & M_{k+1} \\ M_{k+1} & J_{n+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2I_{n+1} & M_{k+1} \\ M_{k+1} & I_{n+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} J_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2I_{n+1} & M_{k+1} \\ M_{k+1} & J_{n+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & M_{k+1} \\ I_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} I_{n+1} & M_{k+1} \\ M_{k+1} & 2M_{k+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} J_{n+1} & M_{k+1} \\ J_{n+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} J_{n+1} & M_{k+1} \\ M_{k+1} & 2M_{k+1} \\ M_{k+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} & M_{k+1} \\ M_{k+1} & \\ I_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} I_{n+1} & M_{k+1} \\ M_{k+1} & 2I_{n+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & M_{k+1} \\ M_{k+1} & \\ J_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} J_{n+1} & M_{k+1} \\ M_{k+1} & 2J_{n+1} \\ M_{k+1} & \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & I_{n+1} \\ M_{k+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & M_{k+1} \\ I_{n+1} & 2M_{k+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} J_{n+1} & J_{n+1} \\ M_{k+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & M_{k+1} \\ J_{n+1} & 2M_{k+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} & I_{n+1} \\ M_{k+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & M_{k+1} \\ I_{n+1} & 2I_{n+1} \\ I_{n+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & J_{n+1} \\ M_{k+1} & \\ M_{k+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} M_{k+1} & M_{k+1} \\ J_{n+1} & 2J_{n+1} \\ J_{n+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} \\ M_{k+1} & I_{n+1} \\ I_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & I_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} \\ M_{k+1} & J_{n+1} \\ J_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2M_{k+1} & J_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix}$	
	$\begin{bmatrix} M_{k+1} \\ M_{k+1} & M_{k+1} \\ I_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2I_{n+1} & I_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix}$			$\begin{bmatrix} M_{k+1} \\ M_{k+1} & M_{k+1} \\ J_{n+1} & \end{bmatrix} \xrightarrow{\alpha\alpha\beta} \begin{bmatrix} 2J_{n+1} & J_{n+1} \\ M_{k+1} & M_{k+1} \\ M_{k+1} & \end{bmatrix}$	

logic where $Y = A \cup B \cup C \cup D$, i.e., $M_Y = M_A \cup M_B \cup M_C \cup M_D$. Then if $M_n = \mu_H [M_Y]$, M_n belongs to a class named Y_H where $Y_H = A_H \cup B_H \cup C_H \cup D_H$, and A_H , B_H , C_H , and D_H , are classes in which their respective members are derived from the μ_H operation on the transform matrices in Class Y.

If M_Y^{-1} defines the inverse of the matrix M_Y over GF(3), then from Property 3.2.7,

$$M_n^{-1} = \mu_V [M_Y^{-1}]. \quad (3.33)$$

Similarly, if $M_n = \mu_V [M_Y]$, then

$$M_n^{-1} = \mu_H [M_Y^{-1}], \quad (3.34)$$

and M_n belongs to a class named Y_V where $Y_V = A_V \cup B_V \cup C_V \cup D_V$, and A_V , B_V , C_V , and D_V , are classes in which their respective members are derived from the μ_V operation on the transform matrices in Class Y.

Example 3.2.2. Let the ternary switching function be the same as in Example 3.2.1. The corresponding transform matrices that require vertical permutations

on the forward transforms are M_{AV} , M_{BV} , M_{CV} and M_{DV} , respectively. Then,

$$\begin{aligned}
 M_{CV} = \mu_V [M_C] = \mu_V & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \\
 = & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.
 \end{aligned}$$

Then from Properties 3.2.7 and 3.2.8, the corresponding inverse transform ma-

trix of M_{CV} is

$$M_{CV}^{-1} = \mu_H [M_C^{-1}] = \mu_H \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}.$$

Similarly, the spectra of the four transform matrices corresponding to those in Example 3.2.1 are presented below.

$$\vec{A}_{AV} = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 & 1 & 0 & 0 & 1 \end{bmatrix}^T,$$

Table 3.7: Class E of LI transforms over $GF(3)$.

	E_H			E_V	
	$\mu_H \begin{bmatrix} M_{n-1} & I_{n-1} \\ I_{n-1} & \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} 2M_{n-1} & I_{n-1} \\ I_{n-1} & \end{bmatrix}$			$\mu_V \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & 2M_{n-1} \end{bmatrix}$	
	$\mu_H \begin{bmatrix} M_{n-1} & I_{n-1} \\ I_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} 2M_{n-1} & I_{n-1} \\ I_{n-1} & J_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & 2M_{n-1} \end{bmatrix}$	
	$\mu_H \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & 2M_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} M_{n-1} & I_{n-1} \\ I_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} 2M_{n-1} & I_{n-1} \\ I_{n-1} & I_{n-1} \end{bmatrix}$	
	$\mu_H \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} I_{n-1} & I_{n-1} \\ I_{n-1} & 2M_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} M_{n-1} & I_{n-1} \\ I_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} 2M_{n-1} & I_{n-1} \\ I_{n-1} & J_{n-1} \end{bmatrix}$	

$$\vec{A}_{BV} = \begin{bmatrix} 1 & 0 & 0 & 1 & 2 & 2 & 2 & 1 & 0 \end{bmatrix}^T,$$

$$\vec{A}_{CV} = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 & 1 & 0 & 0 & 1 \end{bmatrix}^T,$$

$$\vec{A}_{DV} = \begin{bmatrix} 0 & 1 & 0 & 1 & 2 & 2 & 2 & 0 & 1 \end{bmatrix}^T.$$

□

Till now, all the introduced ternary LI transform matrices have been classified into three large categories. Class Y presents those transforms that do not require permutations. Classes Y_H and Y_V include such transforms that require horizontal and vertical permutations on the forward transform matrices, and in addition, vertical and horizontal permutations at the inverse transform matrices, respectively. Some small classes of ternary LI transforms will be presented in the following part, which have distinct properties from the above three large

classes of ternary LI transforms. The first one is denoted as Class E, in which the transform matrices require either both horizontal or both vertical permutations on forward and inverse transform matrices. The transform matrices in Class E have two identity submatrices I_{n-1} and none reverse identity submatrix J_{n-1} in four corners. Class E are divided into two subclasses that are denoted as E_H and E_V . Table 3.7 shows all the transforms in Class E.

Property 3.2.9. Let M_n be any nonsingular square matrices in Class E. Let M_{EH} and M_{EV} be two ternary LI transform matrices in Classes E_H and E_V accordingly, where $M_{EH} = \mu_H [M_n]$, $M_{EV} = \mu_V [M_n]$. Then

$$M_{EH}^{-1} = \mu_H [\beta_B (M_n)] = \beta_B [M_{EH}] \quad (3.35)$$

and

$$M_{EV}^{-1} = \mu_V [\beta_B (M_n)] = \beta_B [M_{EV}]. \quad (3.36)$$

As shown in Table 3.7, the inverse butterfly diagrams can be easily derived by vertical-flipping on the forward butterfly diagrams and multiplying 2 over the crossing lines. This property also applies to the transform matrices in Class A.

There is another Class F that requires permutations, either horizontal or vertical on the forward or inverse transform matrices. The transform matrices in Class F have similar properties as transform matrices in Class B, and they have two reverse identity submatrices J_{n-1} and none identity submatrix I_{n-1} in four corners.

Property 3.2.10. Let M_n be one of the nonsingular square matrices in Class F. Let M_{FH} and M_{FV} be two ternary LI transform matrices belonging to Classes

Table 3.8: Class F of LI transforms over $GF(3)$.

	F_H			F_V
$\mu_H \begin{bmatrix} J_{n-1} & M_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ & & 2M_{n-1} & J_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ M_{n-1} & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} J_{n-1} & 2M_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix}$

$\mu_H \begin{bmatrix} J_{n-1} & M_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} J_{n-1} & & \\ & I_{n-1} & \\ & & 2M_{n-1} & J_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} J_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} J_{n-1} & 2M_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix}$

$\mu_H \begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ M_{n-1} & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} J_{n-1} & 2M_{n-1} \\ & J_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} J_{n-1} & M_{n-1} \\ & J_{n-1} \\ & & 2M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} J_{n-1} & & \\ & J_{n-1} & \\ & & J_{n-1} \end{bmatrix}$

$\mu_H \begin{bmatrix} J_{n-1} & & \\ & I_{n-1} & \\ M_{n-1} & & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_H \begin{bmatrix} J_{n-1} & 2M_{n-1} \\ & I_{n-1} \\ & & J_{n-1} \end{bmatrix}$			$\mu_V \begin{bmatrix} J_{n-1} & M_{n-1} \\ & I_{n-1} \\ & & 2M_{n-1} & J_{n-1} \end{bmatrix} \Leftrightarrow \mu_V \begin{bmatrix} J_{n-1} & & \\ & I_{n-1} & \\ & & J_{n-1} \end{bmatrix}$

F_H and F_V , respectively. By definition, $M_{FH} = \mu_H [M_n]$, $M_{FV} = \mu_V [M_n]$, then

$$M_{FH}^{-1} = \mu_H [\alpha_1 \beta_B (M_n)] \quad (3.37)$$

and

$$M_{FV}^{-1} = \mu_V [\alpha_1 \beta_B (M_n)]. \quad (3.38)$$

All the forward and inverse transform matrices in Class F and their butterfly diagrams are shown in Table 3.8. Since the recursive matrices possess only one submatrix M_{n-1} , or its multiplication by 2 over $GF(3)$, the ternary LI transform matrices in Class F have the same computational costs as that of the transform matrices in Class E. This property also applies to the transform matrices in Class B.

3.2.2 Experimental results

Four ternary LI transform matrices are selected from Classes A, B, C, D and have been implemented using Microsoft Visual C++ and run on PIII 500MHz computer with 128 MB RAM. Some benchmark functions have been modified to represent ternary functions instead of original MCNC, IWLS'93 and ISCAS'95 binary benchmark functions. The translation from binary to ternary cases has been done by changing every 2 input (output) bits in binary files to an input (output) symbol in ternary files. If the number of input and/or output variables is odd, then a zero bit is added behind the binary cubes to make it even. For input (output), - - is taken as -, 00 is taken as 0, 01 is taken as 1 and 10 is taken as 2, whereas 11 is not used (taken as 0) for ternary case. The four implemented transform matrices are run on and tested using the modified Ternary Benchmark Functions (TBFs).

The four selected transform matrices belong to Classes A – D, respectively, as shown by

$$M_a = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, M_a^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ 2M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix};$$

$$M_b = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, M_b^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2M_{n-1}^{-1} \end{bmatrix};$$

Table 3.9: Number of non-zero spectral coefficients for $M_a - M_d$.

TBF	M_a^{-1}	M_b^{-1}	M_c^{-1}	M_d^{-1}	TRM^{-1}	$TRM_{<0>}^{-1}$
					Best polarity	Zero polarity
9sym	89	68	103	112	16	23
alu4	938	1092	1987	1959	1362	2041
apex4	194	237	162	162	142	242
clip	126	134	153	158	109	238
con1	19	18	39	41	17	45
ex5	76	63	77	76	55	78
ex1010	225	228	189	199	236	242
inc	52	51	54	53	44	81
misex1	66	53	63	61	31	65
rd84	54	67	58	55	5	5
squar5	15	18	16	18	10	25
xor5	17	14	12	11	5	9
z5xp1	35	40	53	53	27	69

$$M_c = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, M_c^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2I_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix};$$

$$M_d = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, M_d^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2J_{n-1} \end{bmatrix}.$$

The well-known ternary multi-polarity Reed-Muller (TRM) transform matrix is also implemented and run on the same TBFs. The numbers of non-zero spectral coefficients for TRM are used to compare with the results for the four selected transform matrices.

Table 3.9 shows the numbers of non-zero spectral coefficients for the four ternary LI transforms as well as ternary best polarity and zero polarity Reed-Muller transform of some TBFs.

Table 3.10: Calculation times in seconds for $M_a - M_d$.

TBF	M_a^{-1}	M_b^{-1}	M_c^{-1}	M_d^{-1}	TRM^{-1}	$TRM_{<0>}^{-1}$
					Best polarity	Zero polarity
9sym	0.02	0.01	0.01	0.01	4.98	0.02
alu4	1.87	1.87	0.57	0.58	3750.94	2.08
apex4	0.09	0.10	0.07	0.07	27.53	0.15
clip	0.03	0.04	0.03	0.03	10.11	0.06
con1	0.01	0.01	<0.01	<0.01	2.38	0.01
ex5	0.34	0.41	0.48	0.50	27.72	0.50
ex1010	0.04	0.05	0.03	0.03	15.04	0.08
inc	0.02	0.01	0.01	0.02	5.79	0.02
misex1	0.02	0.01	0.02	0.02	5.08	0.02
rd84	0.01	0.01	0.06	0.01	2.58	0.01
squar5	0.01	<0.01	<0.01	0.01	2.40	0.01
xor5	<0.01	0.01	0.01	0.01	0.68	0.03
z5xp1	0.01	0.01	0.02	0.01	5.72	0.06

Comparing the experimental results of the four selected ternary LI transform matrices in more details: for some TBFs (9sym, alu4, clip, con1, and z5xp1), M_a^{-1} and M_b^{-1} provide much smaller numbers of non-zero spectral coefficients comparing with the results of the other two LI transforms. It can be explained by the fact that M_a^{-1} and M_b^{-1} do not contain any submatrix X_{n-1} and due to that fact they can get more final spectral coefficients resulting in zero for these benchmark functions.

Referring to Tables 3.1 – 3.6, it is clear that the inverse transform matrices of M_b and M_d are horizontal flipping of the inverse transform matrices of M_a and M_c , respectively. The inverse butterfly diagrams of M_b and M_d are the same as M_a and M_c except that each butterfly diagram of M_b and M_d has an additional permutation part on the left side. It can be seen from Table 3.9 that for some benchmark functions, the number of non-zero spectral coefficients of M_b^{-1} and M_d^{-1} are smaller than or equal to the related numbers of M_a^{-1}

and M_c^{-1} , respectively. Hence, the additional permutation part of the butterfly diagrams plays an important role in reducing the number of non-zero spectral coefficients for some TBFs.

When comparing the results of the four ternary LI transform matrices to zero-polarity TRM transform, it is obvious that the four ternary LI transform matrices can always get better results on reducing the non-zero spectral coefficients. After searching through all the 3^n polarities, the results of best polarity TRM transform are better (smaller number of non-zero spectral coefficients) than the results of the four ternary LI transform matrices and are the most efficient. However, for the ternary benchmark function ex1010, all selected ternary LI transform matrices produce spectra that have smaller number of non-zero coefficients comparing to best polarity TRM transform while for alu4, M_a^{-1} and M_b^{-1} are better than the best polarity TRM transform. It should also be noticed that in order to obtain the best polarity results of TRM transform, all 3^n spectra have to be calculated, and it always takes long time to search through all 3^n polarities of TRM transform. On the other hand, the calculation of the ternary LI transforms needs to be performed only once for each of them and due to that fact the speed of calculations for the ternary LI transforms is always much faster than that of TRM transform. Table 3.10 shows the calculation times of the four ternary LI transforms and calculation times for searching through all 3^n polarities as well as the times to calculate only the zero polarity of TRM transform. It is clear that the calculation times to search for the best results of TRM transform are much longer than the corresponding calculation times for ternary LI transforms.

3.3 F^2 type linearly independent transforms over $GF(3)$

In this section, novel polynomial expansions for ternary functions over $GF(3)$ have been introduced. They are based on new classes of ternary linearly independent forms and their corresponding butterfly diagrams are also shown. The new transform is regrouped into eight related classes and their butterfly diagrams as well as computational costs are also discussed. Logic circuit realization of the new class of polynomial expansions based on introduced transforms are also presented.

3.3.1 Classification of F^2 type fast LI transforms

Based on Definition 3.1.4, a family of new fast classes of transforms which can be recursively defined has been developed. They are constructed by the basic recursive submatrices O_{n-1} , X_{n-1} and M_{n-1} .

Let M_n be a $3^n \times 3^n$ square matrix as specified in Definition 3.1.4. It has one submatrix M_{n-1} in one corner $M_{n-1}^{(1)}$ and three submatrices I_{n-1} fill in the slots $M_{n-1}^{(5)}$, $M_{n-1}^{(6)}$ and $M_{n-1}^{(8)}$. The remaining five submatrices of M_n are filled by zero submatrix O_{n-1} . Then one of the fast transforms is defined by

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}. \quad (3.39)$$

Its forward butterfly diagram is shown in the left upper corner of Table 3.11

for $n = 2$. In all the butterfly diagrams presented in this section, the narrow and broad lines represent the value 1 and 2, respectively. It also should be noticed that due to the fact that many of the connections in that butterfly diagram are direct connections not requiring any calculation the whole butterfly diagram can be executed in one time period what speeds up its computation.

Due to the recursive defined matrix structures, similar factorization and appearance of butterfly diagrams applies for higher dimensions. For example for any n , the butterfly diagram of $n - 1$ size will stay on the top and other two parts that have the same size of $n - 1$ will be placed below and there is the crossing between them.

The inverse matrix of (3.39) is given by

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix}. \quad (3.40)$$

There are altogether four pairs of fast forward and inverse transform matrices which are categorized as Class S.

The three fast forward transform matrices can be derived from (3.39) by using the following operations together or separately,

1. Multiplying by 2 the central submatrix;
2. Executing the operators μ_{EH} and μ_{EV} together.

Similarly, the three inverse transform matrices can be derived from (3.40) by applying the same operations described above.

Table 3.11: Transform matrices and butterfly diagrams of Class S.

	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	

From the definitions of Class S, the following properties can be obtained.

Property 3.3.1. Let $M_S = M_n$ be one of the square matrices in Class S. Then, the inverse matrix of M_S can be obtained by the following steps,

1. Selecting the central submatrix and the submatrix that is opposite to M_{n-1} ;
2. Multiplying by 2 the two selected submatrices and interchanging them with each other.

The entire recursive fast forward and inverse transform matrices of Class S and their fast butterfly diagrams for $n = 2$ are shown in Table 3.11.

Horizontal-flipping or vertical-flipping of the transform matrices of Class S will develop a new class called Class P.

Example 3.3.1. Based on (3.39), the horizontal-flipping of the matrix M'_n will create a new matrix M''_n as follows,

$$\begin{aligned} M''_n = \mu_{EH}[M'_n] &= \mu_{EH} \begin{bmatrix} M'_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \\ &= \begin{bmatrix} O_{n-1} & O_{n-1} & M'_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}, \end{aligned}$$

and its inverse transform matrix M''^{-1}_n is obtained by vertical-flipping of the transform matrix in (3.40) as follows,

$$\begin{aligned} M''^{-1}_n = \mu_{EV}[M'^{-1}_n] &= \mu_{EV} \begin{bmatrix} M'^{-1}_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix} \\ &= \begin{bmatrix} O_{n-1} & J_{n-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M'^{-1}_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \end{aligned}$$

□

Table 3.12 gives all the recursive forward and inverse transform matrices and their corresponding butterfly diagrams for $n = 2$ of Class P. The following properties of Class P may be derived.

Property 3.3.2. Let $M_P = M_n$ be one of the square matrices in Class P. Then, the inverse of M_P is given by

Table 3.12: Transform matrices and butterfly diagrams of Class P.

$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

1. Selecting the central submatrix and the submatrix which is opposite to M_n ;
2. Multiplying by 2 over the two selected submatrices and interchanging them;
3. Executing the two operators μ_{EH} and μ_{EV} over the matrix.

The relations between Classes S and P are given by the following properties.

Property 3.3.3. For any forward matrix M_P in Class P, a forward matrix M_S can be found in Class S and the following relations exist:

$$\mu_{EH}[M_S] = M_P, \mu_{EH}[M_S^{-1}] = M_P^{-1} \quad (3.41)$$

or

$$\mu_{EV}[M_S] = M_P, \mu_{EV}[M_S^{-1}] = M_P^{-1}. \quad (3.42)$$

Table 3.13: Transform matrices and butterfly diagrams of Class S^* .

	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	

Property 3.3.4. Let $\widetilde{M}_S$ and $\widetilde{M}_P$ represent any square matrix in Classes S and P, respectively. If $\widetilde{M}_S = \mu_{EH}[\widetilde{M}_P]$ or $\widetilde{M}_S = \mu_{EV}[\widetilde{M}_P]$, then from Properties 3.3.1 – 3.3.3,

$$\widetilde{M}_P^{-1} = \mu_{EV}[\widetilde{M}_S^{-1}] \quad (3.43)$$

and

$$\widetilde{M}_P^{-1} = \mu_{EH}[\widetilde{M}_S^{-1}]. \quad (3.44)$$

Proof. Let $\widetilde{M}_S = M_n$ and $\widetilde{M}_P = \mu_{EH}[M_n]$. Comparing Properties 3.3.1 and 3.3.2, it is clear that their first two operations are the same and when they are denoted by operator κ , then

$$\widetilde{M}_P^{-1} = \mu_{EH}\mu_{EV}[\kappa[\widetilde{M}_P]].$$

Table 3.14: Transform matrices and butterfly diagrams of Class P^* .

$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$

$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

Replacing $\widetilde{M}_P$ by $\mu_{EH}[M_n]$,

$$\widetilde{M}_P^{-1} = \mu_{EH}\mu_{EV}[\kappa[\mu_{EH}[M_n]]] = \mu_{EV}[\kappa[M_n]].$$

From Property 3.3.1,

$$\widetilde{M}_S^{-1} = \kappa[\widetilde{M}_S] = \kappa[M_n].$$

Then

$$\widetilde{M}_P^{-1} = \mu_{EV}[\widetilde{M}_S^{-1}].$$

(3.44) of Property 3.3.4 can have similar proof. $\square$

There are also two other classes that possess structures similar to Classes S and P. They are denoted as Classes S^* and P^* . The transform matrices of Class S^* are derived from Class S by interchanging all submatrices M_{n-1} with I_{n-1} .

The following equation presents one transform matrix of Class S^* derived from (3.39)

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix}. \quad (3.45)$$

Its inverse matrix below can be obtained from (3.40) by interchanging submatrices M_{n-1} and I_{n-1} as shown below

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \end{bmatrix}. \quad (3.46)$$

Class P^* is also derived from Class P in the same manner as Class S^* . Tables 3.13 and 3.14 give the transform matrices and butterfly diagrams for $n = 2$ in these two classes. Some properties of Class S^* and Class P^* are given below.

Property 3.3.5. Let $M_{S^*} = M_n$ be one of the square matrices in Class S^* . Then, the inverse matrix of M_{S^*} can be obtained by the following steps,

1. Selecting the central submatrix and the submatrix that is opposite to I_{n-1} ;
2. Multiplying by 2 over the two selected submatrices and interchanging them with each other.

Property 3.3.6. Let $M_{P^*} = M_n$ be one of the square matrices in Class P^* . Then, the inverse of M_{P^*} is given by,

1. Selecting the central submatrix and the submatrix which is opposite to J_n ;

2. Multiplying by 2 the two selected submatrices and interchanging them;
3. Executing the two operators μ_{EH} and μ_{EV} over the matrix.

Classes S^* and P^* also possess the same relations that apply to Classes S and P that were given by Properties 3.3.3 and 3.3.4.

Interchanging the submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(6)}$ in (3.39), a new transform is given by

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}. \quad (3.47)$$

Another similar new transform is given by interchanging the two submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(8)}$ of (3.39), as given in the following equation,

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix}. \quad (3.48)$$

The following example shows the relations to calculate two new transforms for higher sizes.

Example 3.3.2. The symbols $M_{n(a)}$ and $M_{n(b)}$ are used here to denote two new transforms which have been defined by (3.47) and (3.48), respectively.

$$A(n) = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix},$$

$$B(2) = \begin{bmatrix} I_1 & O_1 & O_1 \\ O_1 & I_1 & I_1 \\ O_1 & I_1 & O_1 \end{bmatrix}, B(n) = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & B(n-1) \end{bmatrix}.$$

For $n = 2$,

$$M_{2(a)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} = B(2) \times A(2),$$

$$M_{2(b)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} = A(2) \times B(2).$$

Figs. 3.2 and 3.3 give the forward butterfly diagrams of $M_{n(a)}$ and $M_{n(b)}$, respectively.

Figure 3.2: Forward butterfly diagram of $M_{n(a)}$. Figure 3.3: Forward butterfly diagram of $M_{n(b)}$.

It is obvious that the corresponding parts of $M_{n(a)}$ and $M_{n(b)}$ have the same butterfly structures, but the ordering of the parts is opposite. Even for higher n , they all have regular structures and keep the same relations. $\square$

By using the following operations over (3.47) and (3.48), together or separately, other six fast transform matrices can be derived,

1. Multiplying by 2 over the central submatrix;
2. Executing the operators μ_{EH} and μ_{EH} together.

Moreover, by replacing the central submatrices I_{n-1} by M_{n-1} , eight more transform matrices are created. There are altogether 16 pairs of forward and inverse transform matrices created this way that are categorized as Class T. All

of them together with their butterfly diagrams for $n = 2$ are presented in Tables 3.15 and 3.16.

Property 3.3.7. Let $M_T = M_n$ be one of the square matrices in Class T. Then, the inverse of M_T is obtained by the following steps:

1. Replacing the submatrix $M_{n-1}^{(5)}$ by $M_{n-1} \Leftrightarrow I_{n-1}$ and multiplying it by 2;
2. Interchanging the submatrix $M_{n-1}^{(5)}$ with the zero submatrix O_{n-1} , which also locates in the diagonal;
3. Interchanging the two non-zero submatrices which locate in either the central column or the central row.

For the two fast transform matrices of Class T defined by (3.47) and (3.48), from Property 3.3.7, their inverse transform matrices are

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \end{bmatrix}, \quad (3.49)$$

and

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & 2M_{n-1}^{-1} \end{bmatrix}, \quad (3.50)$$

respectively.

Another new class of fast transforms is derived from Class T by making horizontal-flipping of their transform matrices. Altogether 32 transform matrices including 16 forward and 16 inverse transform matrices derived from Class T are categorized as Class Q. The transforms of Class Q have similar structures

Table 3.15: Transform matrices and butterfly diagrams of the first half of Class T.

	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^1 & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^1 & M_{n-1}^1 & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^1 \\ O_{n-1} & I_{n-1} & 2M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^1 & I_{n-1} & O_{n-1} \\ M_{n-1}^1 & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^1 & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & M_{n-1}^1 & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^1 \\ O_{n-1} & I_{n-1} & M_{n-1}^1 \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^1 & I_{n-1} & O_{n-1} \\ M_{n-1}^1 & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	

Table 3.16: Transform matrices and butterfly diagrams of the second half of Class T.

	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2I_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2I_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$	

of their butterfly diagrams with the corresponding transforms in Class T. The difference is that each matrix has an additional permutation part in the left part of its butterfly diagram. Tables 3.17 and 3.18 give the entire forward and inverse transform matrices and their butterfly diagrams for $n = 2$ of Class Q.

Properties 3.3.3 and 3.3.4 are valid also for Classes T and Q with the only change of symbols S and P to symbols T and Q accordingly.

There are still two new classes of fast transform matrices which are denoted as Classes T^* and Q^* . They are derived from Classes T and Q, respectively, by the following two steps,

1. Replacing all submatrices M_{n-1} by I_{n-1} (for Class T), or by J_{n-1} (for Class Q);
2. Replacing all submatrices I_{n-1} (for Class T) or J_{n-1} (for Class Q) by M_{n-1} .

It is obvious that Classes T^* and Q^* have similar properties as Classes T and Q. Each of them contains 16 pairs of forward and inverse transform matrices. Tables 3.19 – 3.22 present all forward and inverse transform matrices as well as their butterfly diagrams for $n = 2$ of Classes T^* and Q^* .

All the eight classes, Classes S, P, T and Q, and Classes S^* , P^* , T^* and Q^* form a family of linearly independent transforms which possess fast algorithms in forward and inverse transform matrices. All the introduced linearly independent transforms are classified according to their mathematical relationship of forward transform matrices with their respective inverse transform matrices. They are constructed through four basic submatrices, the zero submatrix O_{n-1} , the identity submatrix I_{n-1} , the reverse-identity submatrix J_{n-1} and the recur-

Table 3.17: Transform matrices and butterfly diagrams of the first half of Class Q.

	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & 2M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2M_{n-1}^{-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	

Table 3.18: Transform matrices and butterfly diagrams of the second half of Class Q.

	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	

sive submatrix M_{n-1} in which general rules applies to them, according to their classifications.

Moreover, from the basic linearly independent transform matrices, there are some other classes that can be derived based on the following property.

Property 3.3.8. Let M_n be any nonsingular square $3^n \times 3^n$ matrix with modulo-3 inverse M_n^{-1} , Then

$$[\mu_H[M_n]]^{-1} = \mu_V[M_n^{-1}] \quad (3.51)$$

and

$$[\mu_V[M_n]]^{-1} = \mu_H[M_n^{-1}] \quad (3.52)$$

In this new category, horizontal or vertical permutations are required before making the matrices recursive.

Example 3.3.3. Let a 2-variable ternary switching function be

$$\vec{F} = \begin{bmatrix} 2 & 0 & 1 & 0 & 2 & 1 & 2 & 2 & 0 \end{bmatrix}^T.$$

Let ternary linearly independent matrix M_Q of order 3^n belong to Class Q, with recursive equation defined as

$$M_n = \begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}.$$

Table 3.19: Transform matrices and butterfly diagrams of the first half of Class T^* .

	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2I_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & 2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2I_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	

Table 3.20: Transform matrices and butterfly diagrams for the second half of Class T*.

	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & 2M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ I_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	

When $n = 2$,

$$M_2 = M_Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

From (3.3),

$$\vec{A}_Q = M_Q^{-1} \vec{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \\ 2 \\ 1 \\ 2 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 2 \\ 1 \\ 0 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

It is clear that for the particular truth vector, the transform matrix selected from Class Q results in a simpler linearly independent spectrum with more number of zeros.

From Definition 3.1.2, the ternary linearly independent polynomial expan-

sion of the same function can be written as

$$\begin{aligned} f(x_1, x_0) &= A_1 \cdot g_1 + A_2 \cdot g_2 + A_3 \cdot g_3 + A_5 \cdot g_5 \\ &= 2 \cdot g_1 + 2 \cdot g_2 + g_3 + 2 \cdot g_5, \end{aligned}$$

where ternary basis functions are,

$$\begin{aligned} g_1 &= x_1^2 \cdot x_0^1 \\ g_2 &= x_1^2 \cdot x_0^0 \\ g_3 &= x_1^0 \cdot x_0^2 + x_1^1 \cdot x_0^2 \\ g_5 &= x_1^0 \cdot x_0^0 + x_1^1 \cdot x_0^1. \end{aligned}$$

The symbol x_j^i represents ternary variables and the superscript i refers to the corresponding ternary values and all the operations are performed over $GF(3)$.

The ternary linearly independent expansion can be implemented using a generator of basis transform functions, its multiplications over spectral vector and the final additions. The logic circuit realization is shown in Fig. 3.4. $\square$

3.3.2 Computational costs

Due to their similar basic constructions, matrices and corresponding butterfly diagrams as shown in Tables 3.11 – 3.22, the eight classes of new ternary LI transform matrices can be categorized into two groups. One contains four Classes S, P, T and Q, and the other one contains the remaining four Classes S^* , P^* , T^* and Q^* . The four classes of each group have the same computation complexity. The following formula that are based on butterfly diagrams as de-

Table 3.21: Transform matrices and butterfly diagrams of the first half of Class Q^* .

	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ M_{n-1} & 2M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2M_{n-1} & M_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	

Table 3.22: Transform matrices and butterfly diagrams of the second half of Class Q*.

	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & 2M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2M_{n-1}^{-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ M_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & 2M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & M_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ M_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & M_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \end{bmatrix}$

Figure 3.4: Logic circuit realization of M_Q .

scribed in previous tables can be used to calculate the total number of additions required to compute the transform of an n -variable ternary function of all the eight classes.

$$S_n = S_{n-1} + 3^{n-1}. \quad (3.53)$$

For Classes S, P, T and Q, since $S_1 = 1$, and for arbitrary n ,

$$S_n = (3^n - 1)/2. \quad (3.54)$$

For Classes S^* , P^* , T^* and Q^* , since $S_1^* = 2$, and for arbitrary n ,

$$S_n^* = (3^n + 1)/2. \quad (3.55)$$

3.4 Classification of linearly independent helix transforms

In this section, properties of the LI helix transforms over $\text{GF}(3)$ are investigated and compared with the best known transform over $\text{GF}(3)$ i.e. ternary Reed-Muller transform. They are named linearly independent helix transforms over $\text{GF}(3)$ due to their symmetrical structure along the diagonal in the transform matrices. Relations and properties between different helix matrices are also shown. The introduced ternary helix transforms have very regular butterfly diagrams due to their recursive equations thus they can be easily expanded to any dimension. Experimental results for them show that the new LI helix transforms are able to give polynomial expansions with less number of non-zero spectral coefficients than those based on ternary Reed-Muller transform for most ternary benchmark functions.

3.4.1 Properties of helix transforms

There are four classes of LI helix transforms presented in this section and their different properties and relations are discussed. They are all $3^n \times 3^n$ square matrices, which can be separated into nine submatrices with the matrix size $3^{n-1} \times 3^{n-1}$. All of them are constructed by four types submatrices, H_{n-1} and O_{n-1} , I_{n-1} and J_{n-1} .

Table 3.23 gives five pairs of forward and inverse transform matrices, which are known as *Left-Positive-Helix* transforms, from LPa to LPe . The first helix

Table 3.23: Class LP of helix transforms.

a		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ H_{n-1} & H_{n-1} & H_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2H_{n-1}^{-1} & H_{n-1}^{-1} & 2H_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
b		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2H_{n-1}^{-1} & I_{n-1} & 2H_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
c		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ H_{n-1} & I_{n-1} & H_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2I_{n-1} & I_{n-1} & 2I_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
d		$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & H_{n-1} & I_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ 2H_{n-1}^{-1} & H_{n-1}^{-1} & 2H_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ H_{n-1} & H_{n-1} & H_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ 2I_{n-1} & H_{n-1}^{-1} & 2I_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$

transform LPa in this class is given below,

$$H_n = \begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ H_{n-1} & H_{n-1} & H_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix}. \quad (3.56)$$

It contains only two types submatrices, H_{n-1} and O_{n-1} , and is recursively defined. The inverse transform matrix of LPa is obtained by multiplying 2 over all $3^{n-1} \times 3^{n-1}$ submatrices, which are not in the diagonal of the matrix, as

shown below,

$$H_n^{-1} = \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ 2H_{n-1}^{-1} & H_{n-1}^{-1} & 2H_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}. \quad (3.57)$$

Generally, the inverse helix transform matrices of Class LP can be obtained from their corresponding forward helix transform matrices following Rule 3.4.1.

Rule 3.4.1.

1. If the matrix does not contain the submatrix I_{n-1} , then go to Step 3, else
2. Select the two non-zero submatrices that are not in the diagonal of the matrix, interchange them from H_{n-1} to I_{n-1} , or from I_{n-1} to H_{n-1} ($H_{n-1} \Leftrightarrow I_{n-1}$), and
3. Multiply the two non-zero submatrices in Step 2 by 2.

Due to the fact that helix transforms contain very few non-zero elements in their matrices, analyzing the structure of the transform matrices and detecting the location of non-zero elements are important for reducing computational costs of helix transforms.

In the following properties, the symbols i and j are used to denote the number of row and column, respectively. Both of them belong to the set $\{ 0, \dots, 3^{n-1} - 1 \}$ in natural ternary ordering.

Property 3.4.1. In the forward transform matrix LPa , the sum of all elements located in every even number of row i is equal to 1 except for the central row in which the sum is equal to 0; the sum of all elements located in every odd number of row i is equal to 0.

Property 3.4.2. In the forward transform matrix LPa , the sum of all elements located in every even number of column j is equal to 1; the sum of all elements located in every odd number of column j is equal to 2.

Property 3.4.3. In the inverse transform matrix LPa^{-1} , the sum of all elements located in every even number of row i is equal to 1 while the sum of every odd number of row i is equal to 2.

Property 3.4.4. In the inverse transform matrix LPa^{-1} , the sum of all elements located in every column j is equal to 0 except that the central column contains only one non-zero element '1' in the central slot.

Property 3.4.5. In the forward transform matrix LPb , the sum of all elements located in row i , $3^{n-1} \leq i \leq 2 \cdot 3^{n-1} - 1$, is equal to 0, while the sum is equal to 2 for the same row in LPc^{-1} .

Property 3.4.6. In the inverse transform matrix LPb^{-1} , the sum of all elements located in column j , $3^{n-1} \leq j \leq 2 \cdot 3^{n-1} - 1$, is equal to 1 and the same property applies to LPc .

Property 3.4.7. In the forward transform matrix LPd , the sum of all elements located in column j , $0 \leq j \leq 3^{n-1}$ and $2 \cdot 3^{n-1} - 1 \leq j \leq 3^n - 1$, is equal to 2, while the sum is equal to 0 for the same column in LPe^{-1} .

Property 3.4.8. In the inverse transform matrix LPd^{-1} , the sum of all elements located in row i , $0 \leq i \leq 3^{n-1}$ and $2 \cdot 3^{n-1} - 1 \leq i \leq 3^n - 1$, is equal to 1 and the same property applies to LPe .

Class RP contains five pairs of *Right-Positive-Helix* transforms, which are denoted from RPa to RPe . They are shown in Table 3.24. The transform

Table 3.24: Class RP of helix transforms.

a		$\begin{bmatrix} H_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & 2H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & 2H_{n-1}^{-1} & H_{n-1}^{-1} \end{bmatrix}$
b		$\begin{bmatrix} H_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & 2H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & 2H_{n-1}^{-1} & H_{n-1}^{-1} \end{bmatrix}$
c		$\begin{bmatrix} H_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
d		$\begin{bmatrix} I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & 2H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & 2H_{n-1}^{-1} & I_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} I_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \end{bmatrix}$

matrix RPa in this class is given below,

$$H_n = \begin{bmatrix} H_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & H_{n-1} \end{bmatrix}. \quad (3.58)$$

Similar to LPa , RPa contains only two types submatrices, H_{n-1} and O_{n-1} , and is recursively defined. The inverse transform matrix of RPa is obtained by multiplying 2 over all $4^{n-1} \times 3^{n-1}$ submatrices, which are not in the diagonal of

the matrix, as shown below,

$$H_n^{-1} = \begin{bmatrix} H_{n-1}^{-1} & 2H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & 2H_{n-1}^{-1} & H_{n-1}^{-1} \end{bmatrix}. \quad (3.59)$$

In general, the inverse matrices of all the helix transforms in Class RP can be derived from their forward matrices following Rule 3.4.1.

The following properties are derived from the matrix structures of all helix transforms in Class RP.

Property 3.4.9. In the forward transform matrix RPa , the sum of all elements located in every even number of row i is equal to 1; the sum of all elements located in every odd number of row i is equal to 2.

Property 3.4.10. In the forward transform matrix RPa , the sum of all elements located in every even number of column j is equal to 1 except for the central row in which the sum is equal to 0; the sum of all elements located in every odd number of row i is equal to 0.

Property 3.4.11. In the inverse transform matrix RPa^{-1} , the sum of all elements located in every row i is equal to 0 except that the central row contains only one non-zero element '1' in the central column.

Property 3.4.12. In the inverse transform matrix RPa^{-1} , the sum of all elements located in every even number of column j is equal to 1 while the sum of every odd number of column j is equal to 2.

Property 3.4.13. In the forward transform matrix RPb , the sum of all elements located in column j , $3^{n-1} \leq j \leq 2 \cdot 3^{n-1} - 1$, is equal to 0, while the sum is equal

to 2 for the same row in RPc^{-1} .

Property 3.4.14. In the inverse transform matrix RPb^{-1} , the sum of all elements located in row i , $3^{n-1} \leq i \leq 2 \cdot 3^{n-1} - 1$, is equal to 1, while the same property applies to RPc .

Property 3.4.15. In the forward transform matrix RPd , the sum of all elements located in row i , $0 \leq i \leq 3^{n-1}$ and $2 \cdot 3^{n-1} - 1 \leq i \leq 3^n - 1$, is equal to 2, while the sum is equal to 0 for the same column in RPe^{-1} .

Property 3.4.16. In the inverse transform matrix RPd^{-1} , the sum of all elements located in column j , $0 \leq j \leq 3^{n-1}$ and $2 \cdot 3^{n-1} - 1 \leq j \leq 3^n - 1$, is equal to 1, while the same property applies to RPe .

The relations between the helix transforms of Classes LP and RP are presented in Property 3.4.17.

Property 3.4.17. For a helix transform RPx in Class RP with its inverse transform RPx^{-1} , there must be a helix transform LPx in Class LP with its inverse transform LPx^{-1} , which have the following relations,

$$RPx = LPx^T \quad (3.60)$$

and

$$RPx^{-1} = [LPx^{-1}]^T, \quad (3.61)$$

where the superscript T indicates matrix transpose operation.

Horizontal flipping on both forward and inverse transform matrices in Class LP will derive a new class called Class LN, which contains five pairs of *Left-Negative-Helix* transforms. They are given in Table 3.25. In these transform

Table 3.25: Class LN of helix transforms.

a		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ H_{n-1} & H_{n-1} & H_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^{-1} \\ 2H_{n-1}^{-1} & H_{n-1}^{-1} & 2H_{n-1}^{-1} \\ H_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
b		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ J_{n-1} & J_{n-1} & J_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^{-1} \\ 2H_{n-1}^{-1} & J_{n-1} & 2H_{n-1}^{-1} \\ H_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
c		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ H_{n-1} & J_{n-1} & H_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^{-1} \\ 2J_{n-1} & J_{n-1} & 2J_{n-1} \\ H_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d		$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ J_{n-1} & H_{n-1} & J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ 2H_{n-1}^{-1} & H_{n-1}^{-1} & 2H_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ H_{n-1} & H_{n-1} & H_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ 2J_{n-1} & H_{n-1}^{-1} & 2J_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

matrices, only two types of non-zero submatrices H_{n-1} and the reverse-identity submatrix J_{n-1} exist. Rule 3.4.2 shows the relations between the forward and inverse transform matrices of Class LN.

Rule 3.4.2.

1. If the matrix does not contain the submatrix J_{n-1} , then go to Step 3, else
2. Select the two non-zero submatrices that are not in the diagonal of the matrix, interchange them from H_{n-1} to J_{n-1} , or from J_{n-1} to H_{n-1} ($H_{n-1} \leftrightarrow J_{n-1}$), and
3. Multiply the two non-zero submatrices in Step 2 by 2.

Properties 3.4.1 – 3.4.8 also apply to the corresponding forward and inverse transform matrices in Class LN.

There are also five pairs of helix transforms which are called *Right-Negative-Helix* transforms. They are categorized into Class RN. They can also be considered as obtained by horizontal flipping of both forward and inverse transform matrices in Class RP. All the forward and inverse transforms in Class RP are presented in Table 3.26. Rule 3.4.2 also applies to derive the inverse transform matrices of Class RP.

Properties 3.4.9 – 3.4.16 also apply to the corresponding forward and inverse transform matrices in Class RN.

Similar relations as shown in Property 3.4.17 also exist between the helix transforms of Classes LN and RN.

Property 3.4.18. For a helix transform RNy in Class RN with its inverse transform RNy^{-1} , there must be a helix transform LNy in Class LN with its

Table 3.26: Class RN of helix transforms.

a		$\begin{bmatrix} O_{n-1} & H_{n-1} & H_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ H_{n-1} & H_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & 2H_{n-1}^{-1} & H_{n-1}^{-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ H_{n-1}^{-1} & 2H_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
b		$\begin{bmatrix} O_{n-1} & J_{n-1} & H_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & 2H_{n-1}^{-1} & H_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1}^{-1} & 2H_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
c		$\begin{bmatrix} O_{n-1} & H_{n-1} & H_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1} & H_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & 2J_{n-1} & H_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1}^{-1} & 2J_{n-1} & O_{n-1} \end{bmatrix}$	
d		$\begin{bmatrix} O_{n-1} & J_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & 2H_{n-1}^{-1} & J_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ J_{n-1} & 2H_{n-1}^{-1} & O_{n-1} \end{bmatrix}$	
e		$\begin{bmatrix} O_{n-1} & H_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ J_{n-1} & H_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & 2J_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ J_{n-1} & 2J_{n-1} & O_{n-1} \end{bmatrix}$	

inverse transform LNy^{-1} , which have the following relations,

$$RNx = LNx^T \quad (3.62)$$

and

$$RNx^{-1} = [LNx^{-1}]^T, \quad (3.63)$$

where the superscript T indicates matrix transpose operation.

From the properties presented above, some interesting results for the four helix transforms can be derived to speed up calculation of their spectral coefficients.

The truth vector $\vec{F}$ of any ternary switching functions must be a $3^n \times 1$ vector and it can be separated into n consecutive 3×1 sub-vector.

For the spectral coefficients vector $\vec{A}$ of the inverse transform matrix LPa^{-1} , the following property exists,

Property 3.4.19. If the elements of $\vec{F}$ are all '1's, then in the spectral coefficients vector $\vec{A}$, the elements in even number of rows are all '1's and others are all '2's; if the elements of $\vec{F}$ are all '2's, then in the spectral coefficients vector $\vec{A}$, the elements in even number of rows are all '2's and others are all '1's.

Example 3.4.1. When $n = 2$, the spectral coefficients vector $\vec{A}$ of the inverse transform matrix LPa^{-1} for the truth vector $\vec{F} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}^T$

is

$$\begin{aligned}\vec{A} &= LPa_2^{-1} \cdot \vec{F} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 2 \\ 1 \\ 2 \\ 1 \\ 2 \\ 1 \\ 2 \end{bmatrix}.\end{aligned}$$

The spectral coefficients in slots 0, 2, 4, 6 and 8 are all '2's and in slots 1, 3, 5 and 7 are all '1's. $\square$

Property 3.4.19 also applies to the inverse transform matrix $LN a^{-1}$.

For the spectral coefficients vector $\vec{A}$ of the inverse transform matrix $RP a^{-1}$, the following property exists,

Property 3.4.20. If the elements of $\vec{F}$ are all '1's or '2's, then in the spectral coefficients vector $\vec{A}$, the values of all elements are equal to zero except the central element is equal to '1' or '2' accordingly.

Property 3.4.20 also applies to the inverse transform matrix $RN a^{-1}$.

The following two properties apply to all helix transforms.

Property 3.4.21. Let the truth vector $\vec{F}$ be separated into three $3^{n-1} \times 1$ sub-vectors as $\vec{F} = \begin{bmatrix} \vec{F}_0 & \vec{F}_1 & \vec{F}_2 \end{bmatrix}^T$. In every sub-vector, all the elements are

non-zero and have the same value. Let the spectral coefficients vector $\vec{A}$ be separated into three $3^{n-1} \times 1$ sub-vectors as $\vec{A} = [\vec{A}_0 \ \vec{A}_1 \ \vec{A}_2]^T$, then the following two relations exist,

1. If $\vec{F}_0 = \vec{F}_1 \neq \vec{F}_2$, then $|\vec{A}_0| \neq |\vec{A}_1| = |\vec{A}_2|$;
2. If $\vec{F}_0 \neq \vec{F}_1 = \vec{F}_2$, then $|\vec{A}_0| = |\vec{A}_1| \neq |\vec{A}_2|$.

Example 3.4.2. When $n = 2$, the spectral coefficients vector $\vec{A}$ of the inverse transform matrix RPa^{-1} for the truth vector $\vec{F} = [1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 2 \ 2 \ 2]^T$ is

$$\begin{aligned} \vec{A} &= RPa_2^{-1} \cdot \vec{F} \\ &= \begin{bmatrix} 1 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}. \end{aligned}$$

□

Analyzing the helix transform matrices, it is clear that all the non-zero elements are distributed symmetrically to the diagonal or reverse-diagonal according to their different helix transform matrices. Due to these special structures, the following symmetry property can be derived.

Property 3.4.22. If all the elements of the truth vector $\vec{F}$ are distributed symmetrically along the central slot, then all the elements of the spectral coefficients vector $\vec{A}$ have also symmetry along the central element.

3.4.2 Computational complexity

The following matrix decomposition method is useful for further reducing the computational costs of calculating the spectra of the LI helix transform.

For the inverse matrix of LPa transform, when $n = 3$, it can be written as a product of three transforms, $LPa_3^{-1} = T_3(3) \cdot T_2(3) \cdot T_1(3)$, where the composite transforms are given in the equations below.

$$T_1(1) = LPa_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix},$$

$$T_1(n) = \begin{bmatrix} T_1(n-1) & O_{n-1} & O_{n-1} \\ O_{n-1} & T_1(n-1) & O_{n-1} \\ O_{n-1} & O_{n-1} & T_1(n-1) \end{bmatrix};$$

$$T_2(2) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_2(n) = \begin{bmatrix} T_2(n-1) & O_{n-1} & O_{n-1} \\ O_{n-1} & T_2(n-1) & O_{n-1} \\ O_{n-1} & O_{n-1} & T_2(n-1) \end{bmatrix};$$

$$T_3(3) = \begin{bmatrix} I_2 & O_2 & O_2 \\ 2I_2 & I_2 & 2I_2 \\ O_2 & O_2 & I_2 \end{bmatrix}.$$

Based on the matrix decomposition method presented, the forward and inverse butterfly diagrams of all helix transforms are also given in Tables 3.23 – 3.26 for the case $n = 2$. The narrow and broad lines denote the values 1 and 2, respectively.

It can be observed that every butterfly diagrams in Classes LN and RN has a permutation part in the left side of the diagram. In fact this part can be located once in any position of the butterfly diagrams, such as in the middle, or in the right side, without affecting the final outputs of the butterfly diagrams. It also should be noticed that the inputs and outputs of the butterfly diagram

are exchangeable without changing the matrices for the first helix transform in each class (LPa , RPa , LNa and RNa).

Considering computational cost of the first helix transform in each class, they have very regular patterns for higher n as described in previous properties. They also have the same computation cost, both for the forward and inverse transform matrices. The total number of required 2-place additions by direct matrix computation method is given by

$$S_{(a)n} = 5^n - 3^n. \quad (3.64)$$

Based on the matrix decomposition method, the number of additions is reduced to

$$S'_{(a)n} = 2n \cdot 3^{n-1}. \quad (3.65)$$

The second and third helix transforms in each class have the same reduced computational costs based on the matrix decomposition method both for the forward and inverse transform matrices, as shown in the following equations.

$$S'_{(b)n} = S'_{(c)n} = 2^n + 6 \cdot (3^{n-1} - 2^{n-1}). \quad (3.66)$$

The last two helix transforms in each class also have the same reduced computational costs as shown below.

$$S'_{(d)n} = S'_{(e)n} = 3^n - 1. \quad (3.67)$$

Table 3.27 gives the comparison results of the computational costs between

the first helix transform in each class and ternary Reed-Muller (TRM) transform both in direct matrix computation method and through the fast transform algorithm as discussed in this section.

Example 3.4.3. Let a ternary switching function be represented by its truth vector $\vec{F} = [2 \ 1 \ 2 \ 2 \ 1 \ 2 \ 2 \ 0 \ 1]^T$. When $n = 2$, the inverse LPa transform is

$$\begin{aligned}
 \overrightarrow{A_{LPa}} &= LPa_2^{-1} \cdot \vec{F} \\
 &= \left(\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \right) \cdot \vec{F} \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 1 \\ 2 \\ 2 \\ 1 \\ 2 \\ 2 \\ 0 \\ 1 \end{bmatrix} \\
 &= [2 \ 0 \ 2 \ 1 \ 0 \ 2 \ 2 \ 0 \ 1]^T.
 \end{aligned}$$

By Definition 3.1.2,

$$f_{LPa} = 2 \cdot (g_0 + g_2 + g_5 + g_6) + g_3 + g_8.$$

Table 3.27: Computational costs of helix transforms.

<i>n</i>	<i>TRM</i>		<i>TRM</i> ⁻¹		<i>Helix</i> , <i>Helix</i> ⁻¹	
	direct	fast	direct	fast	direct	fast
	7 ^{<i>n</i>} − 3 ^{<i>n</i>}	4 <i>n</i> · 3 ^{<i>n</i>−1}	6 ^{<i>n</i>} − 3 ^{<i>n</i>}	3 <i>n</i> · 3 ^{<i>n</i>−1}	5 ^{<i>n</i>} − 3 ^{<i>n</i>}	2 <i>n</i> · 3 ^{<i>n</i>−1}
1	4	4	3	3	2	2
2	40	24	27	18	16	12
3	316	108	189	81	98	54
4	2320	432	1215	324	544	216
5	16564	1620	7533	1215	2882	810
6	116920	5832	45927	4374	14896	2916
7	821356	20412	277749	15309	75938	10206

The spectra of other three ternary helix transforms are

$$\begin{aligned}\overrightarrow{A_{RPa}} &= \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 0 \end{bmatrix}; \\ \overrightarrow{A_{LN a}} &= \begin{bmatrix} 1 & 0 & 2 & 2 & 0 & 1 & 2 & 0 & 2 \end{bmatrix}; \\ \overrightarrow{A_{RN a}} &= \begin{bmatrix} 0 & 2 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}.\end{aligned}$$

From the above results, it can be easily seen that the spectra of $\overrightarrow{A_{RPa}}$ and $\overrightarrow{A_{RN a}}$ as well as $\overrightarrow{A_{LPa}}$ and $\overrightarrow{A_{LN a}}$ have the same number of coefficients as the transform matrices in Classes RN and LN can be derived from Classes RP and LP, respectively, by horizontal flipping on the matrices. □

3.4.3 Experimental results

The first helix transform in each class has been implemented using Microsoft Visual C++ and run on PIII 500 MHz computer with 128 MB RAM.

Table 3.28 shows the numbers of non-zero spectral coefficients for the four helix transforms being compared with the results for the best polarity as well as zero polarity TRM transform. It should be noticed that the helix transforms

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 111

Table 3.28: Number of non-zero spectral coefficients for helix transforms.

TBF	$LPa^{-1} (LNa^{-1})$	$RPa^{-1} (RNa^{-1})$	TRM^{-1}	TRM^{-1}
			Best polarity	Zero polarity
9sym	75	16	16	24
alu4	2183	1141	1362	1978
apex4	242	160	142	241
clip	234	107	109	235
con1	60	17	17	45
ex5	81	46	55	67
ex1010	237	241	236	242
inc	81	54	44	81
misex1	70	37	31	53
rd84	57	16	5	5
squar5	24	18	10	24
xor5	18	7	5	8
z5xp1	80	39	27	68

LPa and LNa have the same experimental results; RPa and RNa also have the same results. It is clear that the results for RPa and RNa are almost always better than LPa and LNa and zero polarity TRM. They also have better results than the best polarity TRM for ternary benchmark functions such as $alu4$, $clip$ and $ex5$.

3.5 Classes of fastest linearly independent transforms over GF(3) and their properties

This section discusses six ternary fastest LI transforms that have the least computational complexity among all ternary LI transforms. It is also shown that the fastest transform are advantageous over well know ternary Reed-Muller transform in terms of the necessary computation time and for some ternary functions the number of non-zero spectral coefficient of the polynomial expansion based

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 112

on the fastest transform can be smaller.

3.5.1 Ternary fastest linearly independent transforms

With the introduction of a new basic recursive submatrix Y_{n-1} in the construction of the matrices, two classes of basic ternary fastest LI transforms are discussed in this section, which have the fastest and most efficient computational complexity in terms of the number of their additions and multiplications over $GF(3)$. They are denoted as Classes Y1 and Y2, and all the corresponding transforms matrices are presented in Tables 3.29 and 3.30, respectively.

All the six pairs of forward and inverse ternary fastest LI transforms of Class Y1 presented in Table 3.29 are denoted as Y1.1 – Y1.6. They are square matrices with the dimension $3^n \times 3^n$, and they are constructed by submatrices M_{n-1} , O_{n-1} , X_{n-1} and Y_{n-1} . Submatrix Y_{n-1} is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements equal to zero except one that is equal to 1 at the bottom left corner for transforms Y1.1 – Y1.3 or at the top right corner for transforms Y1.4 – Y1.6. Submatrix X_{n-1} can only be located in slot $M_{n-1}^{(5)}$ and at most one submatrix O_{n-1} and one submatrix Y_{n-1} are in two opposite slots of $M_{n-1}^{(j)}$ while two submatrices M_{n-1} fill up the remaining two slots, $j \in (1, 3, 7, 9)$. Then the basic $3^n \times 3^n$ forward transform matrix Y1.1 is defined by

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}. \quad (3.68)$$

Fig. 3.5 shows the fast butterfly diagram of the basic forward transform Y1.1 for $n = 3$. In all the butterfly diagrams, the narrow and broad lines represent

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 113

Table 3.29: Fastest LI transforms in Class Y1.

Y1.1		$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
Y1.2		$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
Y1.3		$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
Y1.4		$\begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
Y1.5		$\begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
Y1.6		$\begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 114

Table 3.30: Fastest LI transforms in Class Y2.

Y2.1		$\begin{bmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \end{bmatrix}$
Y2.2		$\begin{bmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \end{bmatrix}$
Y2.3		$\begin{bmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \end{bmatrix}$
Y2.4		$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & Y_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
Y2.5		$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & Y_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
Y2.6		$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & Y_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 115

Figure 3.5: Forward butterfly diagram of Y1.1.

the values 1 and 2, respectively. The following example describes the matrix decomposition method that is used to derive the forward and inverse butterfly diagrams.

The inverse transform matrix of Y1.1 has very similar structure to (3.68). The only difference is that in the inverse transform matrix there is a multiplication by 2 over the submatrix Y_{n-1} , as shown in the following equation,

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (3.69)$$

Fig. 3.6 shows the inverse butterfly diagram of (3.69).

It is obvious, that the only difference between the forward and inverse butterfly diagrams is the value of the cross-lines, which is 1 in the forward butterfly diagram and 2 in the inverse one.

Moving the submatrix Y_{n-1} into $M_{n-1}^{(3)}$ will create another pair of the fastest

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 116

Figure 3.6: Inverse butterfly diagram of Y1.1.

forward and inverse transform matrices named Y1.4. They are described in (3.70) and (3.71),

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \quad (3.70)$$

and

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}, \quad (3.71)$$

respectively.

Their fast forward and inverse butterfly diagrams can be derived from the diagrams shown in Figs. 3.5 and 3.6 by their vertical flipping.

Replacing the central submatrix I_{n-1} in (3.68) and (3.69) by J_{n-1} will create the forward and inverse transform matrices of Y1.2 that are defined by (3.72)

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 117

and (3.73),

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, \quad (3.72)$$

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (3.73)$$

Similarly, the transform matrices of Y1.5 are obtained by moving the submatrix Y_{n-1} of the above equation (3.72) into the slot $M_{n-1}^{(3)}$, what influences its inverse transform also, as shown below,

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \quad (3.74)$$

and

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (3.75)$$

It is obvious that the forward and inverse butterfly diagrams of Y1.5 are obtained by vertical-flipping the butterfly diagrams of the transform matrices described in (3.72) and (3.73).

In (3.76) and (3.77), the submatrix M_{n-1} is located in the central slot instead of the submatrix X_{n-1} . These two ternary fastest LI transforms are Y1.3 and Y1.6. Their inverse transform matrices are obtained in a similar way as other

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 118

transforms by multiplying 2 over submatrix Y_{n-1} .

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, \quad (3.76)$$

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}. \quad (3.77)$$

The forward and inverse butterfly diagrams of all the transform matrices in Class Y1 are also given in Table 3.29. Four pairs of ternary fastest transforms, Y1.1, Y1.2, Y1.4 and Y1.5, have the same computational complexities because they have the same butterfly configurations as permutations do not increase computational additions and multiplications. With the introduction of submatrix Y_{n-1} in the construction of the matrices, it is obvious that the new transform matrices of Class Y1 are the fastest and most efficient LI transform matrices in $GF(3)$ based on their recursive equations from (3.68) to (3.77).

The second category of ternary fastest transform matrices named Class Y2 includes six pairs of forward and inverse basic transform matrices as well. Let M_n be a $3^n \times 3^n$ square matrix as specified in (3.4). The submatrix Y_{n-1} lies in either $M_{n-1}^{(1)}$ or $M_{n-1}^{(9)}$, one submatrix O_{n-1} lies in another slot opposite to the submatrix Y_{n-1} , and the submatrix X_{n-1} is in slot $M_{n-1}^{(5)}$. Two submatrices M_{n-1} fill up the remaining two slots, $M_{n-1}^{(3)}$ and $M_{n-1}^{(7)}$. Then the forward transform

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 119

matrices are defined by

$$M_n = \begin{bmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & X_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \quad (3.78)$$

and

$$M_n = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & X_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & Y_{n-1} \end{bmatrix}. \quad (3.79)$$

Their inverse transform matrices are given in (3.80) and (3.81),

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & X_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \end{bmatrix} \quad (3.80)$$

and

$$M_n^{-1} = \begin{bmatrix} 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & X_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix} \quad (3.81)$$

respectively.

Figs. 3.7 and 3.8 show the fast transform butterfly diagrams of (3.78) and (3.80), respectively, when $X_{n-1} = J_{n-1}$ and $n = 3$.

The last two pairs of ternary fastest transform matrices are obtained by replacing the central submatrices X_{n-1} by M_{n-1} from (3.78) to (3.81).

The butterfly diagrams of the transforms in Class Y2 have very similar patterns to those in Class Y1, except that each of them has an additional vertical-

3.5. *Classes of fastest linearly independent transforms over $GF(3)$ and their properties* 120

Figure 3.7: Butterfly diagram of (3.78).

Figure 3.8: Butterfly diagram of (3.80).

3.5. *Classes of fastest linearly independent transforms over GF(3) and their properties* 121

flipping permutation part in the left. It is evident that the transform matrices in Class Y2 have the same computational complexities as the corresponding transform matrices in Class Y1, because the extra flipping part has no computational cost. Due to the above facts, the transforms in Class Y2 are also the fastest LI transform matrices over GF(3). All the six pairs forward and inverse transform matrices belonging to Class Y2 are presented in Table 3.30 with their corresponding butterfly diagrams.

Other than the presented two Classes Y1 and Y2 of ternary fastest LI transforms, there are also other two classes of ternary LI transforms. They are derived from the previous two Classes Y1 and Y2 by applying the rotation operator R defined in Definition 3.1.13 in the submatrices. Hence, their transform matrices can also be defined in recursive equations. They are denoted as Classes Z1 and Z2, respectively. The following definitions are used to derive these two new transforms.

Example 3.5.1. The first transform matrix Z1.1 of Class Z1 is defined as

$$M_n = R \left(\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \right).$$

Consider the matrix decomposition using the method introduced in Example

3.5. *Classes of fastest linearly independent transforms over GF(3) and their properties* 122

3.3.2, when $n = 2$,

$$\begin{aligned}
 M_2 = R & \left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \right) \\
 & = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} .
 \end{aligned}$$

Fig. 3.9 shows the forward butterfly diagram of transform Z1.1.

The inverse transform matrix of Z1.1 is given in below equation, and its butterfly diagram is shown in Fig. 3.10.

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 123

Figure 3.9: Forward butterfly diagram of Z1.1.

$$M_n^{-1} = R^{-1} \left(\begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix} \right).$$

□

It is obvious that every pair of forward and inverse transform matrix in Class Z1 has a corresponding transform matrix in Class Y1. The butterfly diagrams of the transform matrix in Class Z1 have a permutation part in the left, which do not have computational costs. Thus, all the transforms introduced in Class Z1 are also called ternary fastest LI transforms. Table 3.31 gives all the transform matrices and their corresponding butterfly diagrams of Class Z1.

From Class Y2 and using the rotation operator defined in Definitions 3.1.7 and 3.1.8, Class Z2 can also be derived and all the transform matrices are presented in Table 3.32 together with their butterfly diagrams. The relations and properties of the two Classes Z1 and Z2 together with other classes will also be discussed in Section 3.5.2.

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 124

Table 3.31: Fastest LI transforms in Class Z1.

Z1.1		$R \begin{pmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$
Z1.2		$R \begin{pmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$
Z1.3		$R \begin{pmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$
Z1.4		$R \begin{pmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$
Z1.5		$R \begin{pmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$
Z1.6		$R \begin{pmatrix} M_{n-1} & O_{n-1} & Y_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{pmatrix} \Leftrightarrow R^{-1} \begin{pmatrix} M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{pmatrix}$

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 125

Table 3.32: Fastest LI transforms in Class Z2.

Z2.1		$R^1 \begin{pmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} O_{n-1} & O_{n-1} & M_{n-1}^1 \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^1 & O_{n-1} & 2Y_{n-1} \end{pmatrix}$
Z2.2		$R^1 \begin{pmatrix} Y_{n-1} & Q_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & Q_{n-1} \\ M_{n-1} & Q_{n-1} & Q_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} Q_{n-1} & Q_{n-1} & M_{n-1}^1 \\ O_{n-1} & J_{n-1} & Q_{n-1} \\ M_{n-1}^1 & Q_{n-1} & 2Y_{n-1} \end{pmatrix}$
Z2.3		$R^1 \begin{pmatrix} Y_{n-1} & Q_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & Q_{n-1} \\ M_{n-1} & Q_{n-1} & Q_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} Q_{n-1} & Q_{n-1} & M_{n-1}^1 \\ Q_{n-1} & M_{n-1}^1 & Q_{n-1} \\ M_{n-1}^1 & Q_{n-1} & 2Y_{n-1} \end{pmatrix}$
Z2.4		$R^1 \begin{pmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & Y_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} 2Y_{n-1} & O_{n-1} & M_{n-1}^1 \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^1 & O_{n-1} & Q_{n-1} \end{pmatrix}$
Z2.5		$R^1 \begin{pmatrix} O_{n-1} & Q_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & Q_{n-1} \\ M_{n-1} & Q_{n-1} & Y_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} 2Y_{n-1} & Q_{n-1} & M_{n-1}^1 \\ Q_{n-1} & J_{n-1} & Q_{n-1} \\ M_{n-1}^1 & Q_{n-1} & Q_{n-1} \end{pmatrix}$
Z2.6		$R^1 \begin{pmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & Q_{n-1} \\ M_{n-1} & Q_{n-1} & Y_{n-1} \end{pmatrix} \Leftrightarrow R \begin{pmatrix} 2Y_{n-1} & Q_{n-1} & M_{n-1}^1 \\ Q_{n-1} & M_{n-1}^1 & Q_{n-1} \\ M_{n-1}^1 & Q_{n-1} & Q_{n-1} \end{pmatrix}$

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 126

Figure 3.10: Inverse butterfly diagram of Z1.1.

3.5.2 Properties and relations

Property 3.5.1. The total numbers of non-zero elements in the forward and inverse transform matrices of Y1.1, Y1.2, Y1.4 and Y1.5 are equal to $3^n + 2^n - 1$.

There are 3^n '1's and $2^n - 1$ '2's in the inverse transform matrices while all non-zero elements are '1's in the forward transform matrices.

Property 3.5.2. The total numbers of non-zero elements in the forward and inverse transform matrices of Y1.3 and Y1.6 are equal to $(3^{n+1} - 1)/2$.

All non-zero elements in the forward transform matrices are '1's and there are $(3^n - 1)/2$ '2's and 3^n '1's in the inverse transform matrices.

Property 3.5.3. The row i ($0 \leq i \leq 3^n - 1$) of the forward transform matrices of Y1.1 and Y1.2 has $(1+d)$ '1' elements, where d follows the rules: d is the number of '2' elements behind the last 0 of the ternary representation of i if it does not contain '1', otherwise d is equal to zero.

The row i of the inverse transform matrices has one '1' element and d '2' elements.

3.5. *Classes of fastest linearly independent transforms over GF(3) and their properties* 127

If D is defined as $\{1, 2, \dots, d\}$, then the '2's inside the row i are located at the columns: $i - (3^1 - 1), i - (3^2 - 1), \dots, i - (3^d - 1)$.

Property 3.5.4. $M_{(Y1.4)n} = [M_{(Y1.1)n}]^T$ and $M_{(Y1.5)n} = [M_{(Y1.2)n}]^T$, where the superscript T indicates matrix transpose operation.

Due to this property, a similar approach to the position of non-zero elements in both forward and inverse transform matrices of Y1.4 and Y1.5 can be derived as Property 3.5.3.

Property 3.5.5. The row i ($0 \leq i \leq 3^n - 1$) of the forward transform matrices of Y1.3 and Y1.6 has $(1+d')$ '1' elements, where d' follows the rules: d' is the number of '0' elements behind the last non-zero element of the ternary representation of $(i+1)$.

The row i of their inverse transform matrices has one '1' element and d' '2' elements.

If D' is defined as $\{1, 2, \dots, d'\}$, then the '2's inside the row i are located at the columns $i - (3^1 - 1), i - (3^2 - 1), \dots, i - (3^{d'} - 1)$.

Property 3.5.6. $M_{(Y1.6)n} = [M_{(Y1.3)n}]^T$, where the superscript T indicates matrix transpose operation.

Again, the position of non-zero elements in the forward and inverse transform matrices of Y1.6 can be described in a similar manner as Property 3.5.5.

Till now, the properties for the six pairs of ternary fastest LI transform Y1.1 – Y1.6 have been presented. Other six pairs of ternary fastest transforms that have similar properties to the transforms Y1.1 – Y1.6 but require vertical or horizontal permutations also exist. They are denoted as Y2.1 – Y2.6.

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 128

Property 3.5.7. If M_a is a forward transform matrix in Y2.x, $x \in (0, \dots, 6)$, and M_b is a forward transform matrix in Y1.y, $y \in (0, \dots, 6)$, then when $x = y$,

$$M_a = \mu_{EV}[M_b]. \quad (3.82)$$

Property 3.5.8. If M_a^{-1} is an inverse transform matrix in Y2.x, $x \in (0, \dots, 6)$, and M_b^{-1} is an inverse transform matrix in Y1.y, $y \in (0, \dots, 6)$, then when $x = y$,

$$M_a^{-1} = \mu_{EH}[M_b^{-1}]. \quad (3.83)$$

Looking at the matrix structure of Y2.x, $x \in (0, \dots, 6)$, by Properties 3.5.5 and 3.5.6, the spectral coefficient vectors that are computed from the inverse transforms of Y2.x, $x \in (0, \dots, 6)$, for any ternary function $f(\vec{x}_n)$ are vertical permutation of the spectral coefficient vectors that are computed from the inverse transforms of Y2.y, $y \in (0, \dots, 6)$, when $x = y$. Due to this condition, similar properties as Properties 3.5.1 – 3.5.4 can be obtained from the ternary fastest transforms Y2.x, $x \in (0, \dots, 6)$.

Classes Z1 and Z2 also have very simple and regular matrix structures so similar properties as Properties 3.5.5 – 3.5.8 can be derived from the ternary fastest LI transforms in Classes Z1 and Z2. In addition, the following properties can also be obtained.

Property 3.5.9. Let M_n be a $3^n \times 3^n$ nonsingular square matrix in Class Z1 or Z2, with inverse M_n^{-1} . Defining the integer power of the rotation operator as $R^2(M_n) = R[R(M_n)]$, the following relations are presented.

$$R^r(M_n) = R^{r-4}(M_n), \quad (3.84)$$

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 129

for any $r \in (0, 1, 2, 3)$;

$$R^{-1} [R (M_n)] = R [R^{-1} (M_n)] = M_n; \quad (3.85)$$

$$[R (M_n)]^{-1} = R^{-1} (M_n^{-1}). \quad (3.86)$$

Property 3.5.10. If M_c and M_c^{-1} is a pair of forward and inverse transform matrix in Z1.x, $x \in (0, \dots, 6)$, M_d and M_d^{-1} is a pair of forward and inverse transform matrix in Y1.y, $y \in (0, \dots, 6)$, then when $x = y$,

$$M_c = R [M_d], \quad (3.87)$$

and

$$M_c^{-1} = R^{-1} [M_d^{-1}]. \quad (3.88)$$

Property 3.5.11. If M_e and M_e^{-1} is a pair of forward and inverse transform matrices in Z2.x, $x \in (0, \dots, 6)$, M_f and M_f^{-1} is a pair of forward and inverse transform matrices in Y2.y, $y \in (0, \dots, 6)$, then when $x = y$,

$$M_e = R^{-1} [M_f], \quad (3.89)$$

and

$$M_e^{-1} = R [M_f^{-1}]. \quad (3.90)$$

3.5.3 Computational complexity considerations

In this section, the computation costs of the ternary fastest LI transforms over GF(3) will be derived. The results will be compared with the complexity of

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 130

computing fixed polarity ternary Reed-Muller expansion.

Example 3.5.2. Let a 2-variable ternary switching function be

$\vec{F} = [1 \ 2 \ 0 \ 0 \ 2 \ 0 \ 1 \ 1 \ 1]^T$, then from (2.18) and (2.20),

$$\vec{B} = T_2^{-1} \cdot \vec{F} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \\ 2 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 2 \\ 2 \\ 1 \\ 1 \\ 2 \end{bmatrix}.$$

The polynomial Reed-Muller expansion of this ternary function can be written as

$$\begin{aligned} f_a(x_1, x_0) = & 1 + x_0^0 + x_1^0 + 2 \cdot x_1^0 \cdot x_0^0 + 2 \cdot x_1^0 \cdot x_0^2 + x_1^2 \\ & + x_1^2 \cdot x_0^0 + 2 \cdot x_1^2 \cdot x_0^2. \end{aligned}$$

The ternary fastest LI transform Y2.2 presented in Section 3.5.1 is given by

$$M_n = \begin{bmatrix} Y_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}.$$

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 131

Its inverse transform matrix can be derived from (3.80) as

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2Y_{n-1} \end{bmatrix}.$$

Figs. 3.7 and 3.8 show their forward and inverse butterfly diagrams. For the same ternary switching function $\vec{F}$, when $n=2$, the coefficient vector $\vec{A}$ is given by

$$\vec{A} = M_2^{-1} \cdot \vec{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \\ 2 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 2 \\ 0 \\ 0 \\ 2 \\ 0 \end{bmatrix}.$$

From Definition 3.1.2, the ternary fastest LI polynomial expansion of the same function can be written as

$$\begin{aligned} f_b(x_1, x_0) &= A_0 \cdot g_0 + A_1 \cdot g_1 + A_4 \cdot g_4 + A_7 \cdot g_7 \\ &= g_0 + g_1 + 2 \cdot g_4 + 2 \cdot g_7, \end{aligned}$$

where the truth vectors for g_i , $i = (0, 1, 4, 7)$, are obtained directly from columns 0, 1, 4, and 7 in the forward transform matrix M_2 .

It is clear that f_b has less addition and multiplication operators over $GF(3)$

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 132

than f_a . Both ternary fastest LI and ternary Reed-Muller expansions can be implemented in a similar manner using a generator of basis transform functions, their multiplications over spectral vector and final additions. Similar approach was shown earlier in Fig. 3.4. $\square$

In order to evaluate complexity calculation, Figs. 3.5 – 3.8 can be used. Then the following theorem can be derived.

Theorem 3.5.1. Based on the matrix decomposition method presented here, the number of additions required to compute the ternary fastest LI transforms Y1.1, Y1.2, Y1.4 and Y1.5 are $2^n - 1$ both for the forward and inverse transforms.

Theorem 3.5.1 also applies to Y2.1, Y2.2, Y2.4 and Y2.5.

Theorem 3.5.2. Based on the matrix decomposition method presented here, the number of additions required to compute the ternary fastest LI transforms Y1.3 and Y1.6 are $(3^n - 1)/2$ both for the forward and inverse transforms.

Theorem 2 also applies to Y2.3 and Y2.6.

As shown in Example 3.5.1, transform Z1.x has the same computational cost as transform Y1.y, when $x = y$. The same relation exists between transforms Z2.x and Y2.y, only if $x = y$.

3.5.4 Experimental results

All the fastest LI transforms in Classes Y1 and Z1 have been implemented using Microsoft Visual C++ on Pentium IV 2.8GHz computer with 256 MB RAM.

The numbers of non-zero spectral coefficients for transforms Y1.1 – Y1.6 are given in Table 3.33 as well as the results for transforms Z1.1 – Z1.6 are

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 133

Table 3.33: Number of non-zero spectral coefficients for Class Y1.

TBF	Output	Y1.1	Y1.2	Y1.3	Y1.4	Y1.5	Y1.6	TRM^{-1}	$TRM_{<0>}^{-1}$
								Best polarity	Zero polarity
alu4	O ₁	1390	1390	1214	1393	1393	1214	442	1112
	O ₂	2151	2151	1794	2150	2150	1816	382	1045
	O ₃	1550	1550	1510	1557	1557	1459	592	1322
	O ₄	1131	1131	1145	1130	1160	1111	702	1287
apex4	O ₁	14	14	19	13	13	20	51	112
	O ₂	74	74	81	70	70	69	78	128
	O ₃	54	54	60	57	57	66	77	137
	O ₄	52	52	57	57	57	66	79	116
	O ₅	62	62	65	68	68	73	79	138
	O ₆	106	106	95	103	103	89	76	146
	O ₇	47	47	47	49	49	45	85	159
	O ₈	60	60	62	59	59	55	82	160
	O ₉	11	11	11	14	14	16	52	118
	O ₁₀	49	49	41	46	46	35	51	118
ex1010	O ₁	64	64	78	71	71	77	133	163
	O ₂	70	70	80	66	66	77	134	160
	O ₃	55	55	63	54	54	65	136	156
	O ₄	60	60	73	61	61	72	127	152
	O ₅	58	58	71	59	59	71	137	157
inc	O ₁	32	32	24	33	33	30	17	47
	O ₂	38	38	32	38	38	36	30	51
	O ₃	31	31	27	32	32	25	22	54
	O ₄	17	17	20	16	16	16	23	54
	O ₅	18	18	12	18	18	12	2	6

3.5. *Classes of fastest linearly independent transforms over GF(3) and their properties* 134

Table 3.34: Number of non-zero spectral coefficients for Class Z1.

TBF	Output	Z1.1	Z1.2	Z1.3	Z1.4	Z1.5	Z1.6	TRM^{-1}	$TRM_{<0>}^{-1}$
								Best polarity	Zero polarity
alu4	O ₁	1400	1400	1399	1372	1372	1300	442	1112
	O ₂	2151	2151	1777	2167	2167	1370	382	1045
	O ₃	1546	1546	1515	1544	1544	1323	592	1322
	O ₄	1131	1131	1214	1131	1131	1130	702	1287
apex4	O ₁	13	13	13	16	16	20	51	112
	O ₂	71	71	70	79	79	106	78	128
	O ₃	52	52	52	56	56	77	77	137
	O ₄	53	53	53	56	56	81	79	116
	O ₅	62	62	62	64	64	93	79	138
	O ₆	106	106	105	114	114	147	76	146
	O ₇	46	46	46	51	51	73	85	159
	O ₈	59	59	59	68	68	89	82	160
	O ₉	13	13	13	13	13	17	52	118
	O ₁₀	48	48	48	52	52	66	51	118
ex1010	O ₁	65	65	83	64	64	80	133	163
	O ₂	65	65	79	64	64	81	134	160
	O ₃	58	58	68	53	53	70	136	156
	O ₄	60	60	75	62	62	75	127	152
	O ₅	55	55	61	61	61	78	137	157
inc	O ₁	33	33	32	37	37	48	17	47
	O ₂	40	40	40	44	44	55	30	51
	O ₃	34	34	33	36	36	48	22	54
	O ₄	18	18	18	20	20	29	23	54
	O ₅	18	18	17	18	18	23	2	6

3.5. Classes of fastest linearly independent transforms over GF(3) and their properties 135

Table 3.35: Calculation times for Class Y1.

TBF	Y1.1	Y1.2	Y1.3	Y1.4	Y1.5	Y1.6	TRM^{-1}	$TRM_{<0>}^{-1}$
							Best polarity	Zero polarity
alu4	0.41	0.39	0.39	0.41	0.38	0.38	3750.94	2.08
apex4	0.03	0.02	0.03	0.02	0.02	0.02	27.53	0.15
ex1010	0.02	0.01	<0.01	0.02	<0.01	0.02	15.04	0.08
inc	<0.01	<0.01	<0.01	<0.01	0.02	0.00	5.79	0.02

Table 3.36: Calculation times for Class Z1.

TBF	Z1.1	Z1.2	Z1.3	Z1.4	Z1.5	Z1.6	RM^{-1}	$RM_{<0>}^{-1}$
							Best polarity	Zero polarity
alu4	0.44	0.42	0.42	0.44	0.42	0.42	3750.94	2.08
apex4	0.02	0.02	0.03	0.03	0.02	0.03	27.53	0.15
ex1010	0.02	<0.01	0.03	0.01	0.02	0.03	15.04	0.08
inc	0.01	0.02	<0.01	<0.01	0.01	<0.01	5.79	0.02

given in Table 3.34. The experimental results of best polarity and zero polarity TRM transform are also given in the last two columns of the table. The results for the ternary fastest transforms are better than the results of zero polarity TRM transform for majority of cases. Even though ternary fixed polarity Reed-Muller transform gets smaller number of non-zero spectral coefficients than other transform matrices after searching all 3^n polarities, the fastest transforms still can obtain less spectral coefficients in some cases, such as for TBF apex4 and ex1010.

It also should be noticed that in order to obtain the best polarity result for TRM transform all 3^n spectra have to be calculated what takes a lot of time. On the other hand, the calculation of the fastest transforms needs to be performed only once for each of them. Tables 3.35 and 3.36 gives the calculation time of the transforms in Classes Y1 and Z1, respectively, compared with TRM transform when all the spectral coefficients for all outputs of these TBF have

3.5. Classes of fastest linearly independent transforms over $GF(3)$ and their properties 136

been calculated.

Chapter 4

Linearly Independent Ternary Arithmetic Transforms

It has been shown that in many applications where logic functions need to be analyzed, it is useful to transform a Boolean function into the equivalent arithmetic expansion. In particular, such arithmetic expansions are used in probabilistic verification of a pair of functions and to describe stochastic behavior of a logical circuit under random values of input variables. In this chapter, new approaches to the development of the arithmetic transforms for ternary functions are proposed. Different matrices that satisfy the conditions of linear independence are shown, with particular emphasis on those matrices which can be defined recursively and have fast transforms. Formulae deriving the inverses of recursive matrices are presented.

4.1 Basic definitions of linearly independent ternary arithmetic transforms

Definition 4.1.1. Let M_n be a $N \times N$ ($N = 3^n$) matrix with rows corresponding to minterms and columns corresponding to some switching ternary functions of n variables. If the sets of rows are linearly independent with respect to *Ternary Galois Field*, then M_n has only one inverse in $\text{GF}(3)$ and is said to be *Linearly Independent Ternary Arithmetic* (LITA). If M_n is LITA in $\text{GF}(3)$, then it is a non-singular square matrix with respect to standard algebra and has a unique inverse M_n^{-1} .

Definition 4.1.2. The LITA polynomial expansion for any n -variable ternary function $f(\vec{x}_n)$ is

$$f(\vec{x}_n) = \sum_{j=0}^{3^n-1} A_j g_j, \quad (4.1)$$

where g_j denotes a ternary discrete function which truth vector constitutes the number $j + 1$ column of the LI matrix M_n , $0 \leq j \leq 3^n - 1$, and all operations are over standard algebra.

Definition 4.1.3. The LITA transform based on Definitions 4.1.1 and 4.1.2 can be described by the following general formulae performed in standard algebra:

$$\vec{F} = M_n \vec{A}, \quad (4.2)$$

and

$$\vec{A} = M_n^{-1} \vec{F}, \quad (4.3)$$

where $\vec{F} = [F_0, F_1, \dots, F_{3^n-1}]^T$ is a column vector defining the truth vector of a ternary function $f(\vec{x}_n)$ in a natural ternary ordering, M_n is a LITA ma-

trix of order $N = 3^n$ defined by any n -variable ternary functions and $\vec{A} = [A_0, A_1, \dots, A_{3^n-1}]^T$ is the coefficient column vector for the particular transform matrix M_n with arithmetic inverse M_n^{-1} .

Definition 4.1.4. Let M_n be a matrix as specified in Definition 1. Additionally, M_n is recursively defined by

$$M_n = \begin{bmatrix} M_{n-1}^{(1)} & M_{n-1}^{(2)} & M_{n-1}^{(3)} \\ M_{n-1}^{(4)} & M_{n-1}^{(5)} & M_{n-1}^{(6)} \\ M_{n-1}^{(7)} & M_{n-1}^{(8)} & M_{n-1}^{(9)} \end{bmatrix} \quad (4.4)$$

where each submatrix $M_{n-1}^{(i)}$, $i = (1, 2, 3, \dots, 8, 9)$ has a dimension of $3^{n-1} \times 3^{n-1}$.

Definition 4.1.5. The entire Rotation operator R_E on the square recursive matrix M_n is recursively defined as $R_E/4$ clockwise rotation involving all submatrices each with size 3^{r-1} for $r = n, n-1, \dots, 2$.

Definition 4.1.6. The inverse entire Rotation operator R_E^{-1} on the square recursive matrix M_n is recursively defined as $-R_E/4$ counterclockwise rotation involving all submatrices each with size 3^{r-1} for $r = n, n-1, \dots, 2$.

Property 4.1.1. Let M_n be a $3^n \times 3^n$ nonsingular square matrix with inverse M_n^{-1} . Defining the integer power of the rotation operator as $R_E^2(M_n) = R_E[R_E(M_n)]$, the following properties may be derived. For any $r \in (0, 1, 2, 3)$

$$R_E^r(M_n) = R_E^{r-4}(M_n); \quad (4.5)$$

$$R_E^{-1}[R_E(M_n)] = R_E[R_E^{-1}(M_n)] = M_n; \quad (4.6)$$

$$[R_E(M_n)]^{-1} = R_E^{-1}(M_n^{-1}). \quad (4.7)$$

Example 4.1.1. A $3^n \times 3^n$ nonsingular square matrix M'_n and its inverse matrix M'^{-1}_n are given by the recursive equations,

$$M'_n = \begin{bmatrix} M'_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M'_{n-1} & O_{n-1} \\ M'_{n-1} & O_{n-1} & M'_{n-1} \end{bmatrix}, \quad M'^{-1}_n = \begin{bmatrix} M'^{-1}_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M'^{-1}_{n-1} & O_{n-1} \\ -M'^{-1}_{n-1} & O_{n-1} & M'^{-1}_{n-1} \end{bmatrix}.$$

From Property 4.1.1,

$$R_E^{1-4}(M'_n) = \begin{bmatrix} M'_{n-1} & O_{n-1} & M'_{n-1} \\ O_{n-1} & M'_{n-1} & O_{n-1} \\ M'_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} = R_E^1(M'_n);$$

$$\begin{aligned} [R_E(M'_n)]^{-1} &= \begin{bmatrix} M'_{n-1} & O_{n-1} & M'_{n-1} \\ O_{n-1} & M'_{n-1} & O_{n-1} \\ M'_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M'^{-1}_{n-1} \\ O_{n-1} & M'^{-1}_{n-1} & O_{n-1} \\ M'^{-1}_{n-1} & O_{n-1} & -M'^{-1}_{n-1} \end{bmatrix} \\ &= R_E^{-1} \begin{bmatrix} M'^{-1}_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M'^{-1}_{n-1} & O_{n-1} \\ -M'^{-1}_{n-1} & O_{n-1} & M'^{-1}_{n-1} \end{bmatrix} = R_E^{-1}(M'^{-1}_n). \end{aligned}$$

Definition 4.1.7. Let M_n be a $3^n \times 3^n$ nonsingular square matrix, which is partitioned into appropriate $3^{n-1} \times 3^{n-1}$ dimensional submatrices, as shown in (4.4). If $M^{(j*)}_{n-1} = O_{n-1}$, $j* = \begin{pmatrix} 1, & 3, & 7, & 9 \end{pmatrix}$, then the operator α_A on M_n is defined as selecting the submatrix $M^{(10-j*)}_{n-1}$, which is opposite to $M^{(j*)}_{n-1}$, and changing it to $-M^{(10-j*)}_{n-1}$.

The following notation for submatrices will also be used: I_{n-1} is a $3^{n-1} \times 3^{n-1}$

identity matrix, $J_{n-1}(= R(I_{n-1}))$ is a $3^{n-1} \times 3^{n-1}$ reverse-identity matrix and X_{n-1} is either I_{n-1} or J_{n-1} .

Property 4.1.2. Based on Definition 4.1.7,

$$\alpha_A [\alpha_A (M_n)] = \alpha_A^2 (M_n) = M_n. \quad (4.8)$$

Definition 4.1.8. Let M_n be a $3^n \times 3^n$ nonsingular square matrix, which is partitioned into appropriate $3^{n-1} \times 3^{n-1}$ dimensional submatrices, where $M_{n-1}^{(j*)}$, $j^* = \left(1, 3, 7, 9\right)$, contain only one zero-submatrix O_{n-1} and at least one submatrix M_{n-1} is in the other corner opposite to this zero-submatrix O_{n-1} , and the left two corners contain any combinations of the submatrices X_{n-1} or M_{n-1} but not consecutive I_{n-1} and J_{n-1} together.

The β_A operator on M_n is defined as selecting the submatrix $M_{n-1}^{(10-j*)}$, which is opposite to $M_{n-1}^{(j*)}$, and changing it by the following operations:

- $M_{n-1} \leftrightarrow -X_{n-1}$ if M_{n-1} and X_{n-1} are submatrices of $M_{n-1}^{(j)}$, or
- $X_{n-1} \leftrightarrow -M_{n-1}$ if X_{n-1} is a submatrix of $M_{n-1}^{(j)}$, $j = \left(1, 3, 7, 9\right)$.

Example 4.1.2. A $3^2 \times 3^2$ nonsingular square matrix M_2'' is given by

$$M_2'' = \begin{bmatrix} M_1'' & O_1 & O_1 \\ O_1 & M_1'' & O_1 \\ M_1'' & O_1 & I_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

and when β_A operator applies to M_2'' , the submatrix $M_1^{(7)}$ is changed as $M_1'' \leftrightarrow -I_{n-1}$ by Definition 4.1.8,

$$\begin{aligned} \beta_A[M_2''] &= \beta_A \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} M_1'' & O_1 & O_1 \\ O_1 & M_1'' & O_1 \\ -I_1 & O_1 & I_1 \end{bmatrix}. \end{aligned}$$

The matrix permutation operators μ_H , μ_V , μ_{EH} and μ_{EV} defined in previous chapter from Definition 3.1.5 – 3.1.8 are also used in this chapter.

4.2 F – R type linearly independent ternary arithmetic transforms

In this section, the family of F – R type Linearly Independent Ternary Arithmetic (LITA) transform matrices is discussed in detail. From Definitions 4.1.1 and 4.1.4, altogether 240 F – R type LITA forward transform matrices are selected here and classified into six Classes A – F, which do not require any horizontal and vertical permutations in the matrices. All of them can be formulated recursively and possess fast algorithms. Their respective fast transforms are shown in Tables 4.1 – 4.6. The solid and dotted lines correspond to addition and subtraction; the narrow and broad lines represent the value 1 and 2, respectively. There are also some other classes that require vertical or horizontal permutations. They are discussed in Section 4.2.2. All the transforms matrices are constructed by the basic recursive submatrices O_{n-1} , M_{n-1} , and X_{n-1} , where, at most, one O_{n-1} and two X_{n-1} could appear in the recursive Definition 4.1.4 in locations $M_{n-1}^{(j)}$, $j = \left(1, 3, 7, 9\right)$. As specified in Definition 4.1.4, four submatrices to the left, to the right, to the top and to the bottom of the central submatrix $M_{n-1}^{(5)}$ are zero submatrices and the submatrix $M_{n-1}^{(5)}$ can be either M_{n-1} or X_{n-1} .

4.2.1 Basic recursive F – R type LITA transforms

There are 24 F – R type LITA transform matrices to be categorized as Class A, of which 12 do not contain any submatrix X_{n-1} in the four corners of the matrix and are defined as Class A1 and shown in the upper half of Table 4.1. Four of them do not even contain submatrix X_{n-1} in the central submatrix and have one zero submatrix O_{n-1} in one corner either $M_{n-1}^{(3)}$ or $M_{n-1}^{(7)}$. After exchanging the columns horizontally in all the matrices in Class A1, Class B1 is derived and shown in the upper half of Table 4.2.

Property 4.2.1. Let M_{A1} and M_{B1} be any $3^n \times 3^n$ square matrices in Class A1 and Class B1, respectively. Then, their inverses are given by

$$M_{A1}^{-1} = \alpha_A[M_{A1}] \quad (4.9)$$

and

$$M_{B1}^{-1} = \alpha_A[R^2(M_{B1})]. \quad (4.10)$$

The following example shows a general matrix decomposition method for introduced LITA transforms. Based on the above matrix decomposition, the forward and inverse butterfly diagrams for $n = 3$ are shown in Figs. 4.1 and 4.2, respectively. The same decomposition method can be used for any n .

Example 4.2.1. Let us consider factorization of one transform from Class A for which $X_{n-1} = I_{n-1}$, and its inverse matrix can be obtained from Property

Table 4.1: LITA transforms of Class A

	$\begin{bmatrix} M_k & & \\ & M_{k+1} & \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & M_{k+1}' \\ & & -M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & M_{k+1} \\ & M_{k+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -M_{k+1}' \\ & M_{k+1}' \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} M_k & & \\ & M_{k+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & M_{k+1}' \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & 2M_{k+1} \\ & M_{k+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -2M_{k+1}' \\ & M_{k+1}' \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} M_k & & \\ & I_{n+1} & \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & I_{n+1} \\ & & -M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & M_{k+1} \\ & I_{n+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -M_{k+1}' \\ & I_{n+1} \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} M_k & & \\ & I_{n+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & I_{n+1} \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & 2M_{k+1} \\ & I_{n+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -2M_{k+1}' \\ & I_{n+1} \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} M_k & & \\ & J_{n+1} & \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & J_{n+1} \\ & & -M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & M_{k+1} \\ & J_{n+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -M_{k+1}' \\ & J_{n+1} \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} M_k & & \\ & J_{n+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & \\ & J_{n+1} \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} M_k & 2M_{k+1} \\ & J_{n+1} \\ & & M_{k+1} \end{bmatrix} \begin{bmatrix} M_k' & -2M_{k+1}' \\ & J_{n+1} \\ & & M_{k+1}' \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & M_{k+1} & \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & M_{k+1}' \\ & & -M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & M_{k+1} \\ & M_{k+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -M_{k+1}' \\ & M_{k+1}' \\ & & I_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & M_{k+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & M_{k+1}' \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & 2M_{k+1} \\ & M_{k+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -2M_{k+1}' \\ & M_{k+1}' \\ & & I_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & I_{n+1} & \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & I_{n+1} \\ & & -I_{n+1} \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & M_{k+1} \\ & I_{n+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -M_{k+1}' \\ & I_{n+1} \\ & & I_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & I_{n+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & I_{n+1} \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & 2M_{k+1} \\ & I_{n+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -2M_{k+1}' \\ & I_{n+1} \\ & & I_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & J_{n+1} & \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & J_{n+1} \\ & & -I_{n+1}' \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & M_{k+1} \\ & J_{n+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -M_{k+1}' \\ & J_{n+1} \\ & & I_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n+1} & & \\ & J_{n+1} & \\ & & 2M_{k+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & \\ & J_{n+1} \\ & & -2M_{k+1}' \end{bmatrix}$			$\begin{bmatrix} I_{n+1} & 2M_{k+1} \\ & J_{n+1} \\ & & I_{n+1} \end{bmatrix} \begin{bmatrix} I_{n+1} & -2M_{k+1}' \\ & J_{n+1} \\ & & I_{n+1} \end{bmatrix}$	

4.2.1, where $X_{n-1} = I_{n-1}$ as

$$M_a = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \xleftrightarrow{\alpha_A} M_a^{-1} \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ -2M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}.$$

For $n = 3$, $M_n = C(n) \cdot B(n) \cdot A(n)$,

$$A(1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix}, A(n) = \begin{bmatrix} A(n-1) & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & A(n-1) \end{bmatrix};$$

$$B(2) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

$$B(n) = \begin{bmatrix} B(n-1) & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & B(n-1) \end{bmatrix}; C(n) = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2I_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}.$$

□

For all the transform matrices in Class A1, by keeping the zero submatrix

Figure 4.1: Forward butterfly diagram of M_a .

Figure 4.2: Inverse butterfly diagram of M_a .

O_{n-1} which is in one corner and another submatrix in its opposite corner and changing the other two submatrix which are in the other two corners from M_{n-1} to I_{n-1} , the other 12 transform matrices are defined, and they are categorized into sub-class Class A2. Their inverses can be obtained by multiplying -1 through submatrices $M_{n-1}^{(3)}$ and $M_{n-1}^{(7)}$ of the forward transform matrices. Similarly, Class B2 can be derived from Class A2 after exchanging columns horizontally in all the matrices of Class A2. Their inverses can be obtained by multiplying -1 through submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(9)}$ of the forward transform matrices and executing the rotation operator R_E twice on them. Both forward and inverse transform matrices for Class A2 and Class B2 are shown in the lower half of Tables 4.1 and 4.2, respectively.

Let one submatrix X_{n-1} occupy $M_{n-1}^{(9)}$ or two submatrices X_{n-1} occupy $M_{n-1}^{(1)}$ and $M_{n-1}^{(7)}$, and one zero submatrix O_{n-1} be in $M_{n-1}^{(3)}$, then 24 transform matrices of Class C are defined. There are altogether 48 forward transform matrices in Class C. The other 24 transforms are derived by executing the operator R_E^2 on the first 24 transform matrices of Class C. Table 4.3 shows only butterfly diagrams of Class C for the case $n = 2$ and $X_{n-1} = I_{n-1}$.

Property 4.2.2. Let M_C be any $3^n \times 3^n$ square matrices in Class C. Then, the inverse matrix M_C^{-1} is given by

$$M_C^{-1} = \beta_A [M_C]. \quad (4.11)$$

After exchanging the columns horizontally in all the matrices of Class C, Class D is created. It is clear that Class D also contains 48 forward transform matrices. All the transform matrices of Class D and their corresponding butterfly diagrams are shown in Table 4.4, in case of $n = 2$ and $X_{n-1} = J_{n-1}$.

Table 4.2: LITA transforms of Class B

	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ M_{k_1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & M_{k_1}^* \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & M_{k_1} \\ M_{k_1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ M_{k_1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & M_{k_1}^* \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & M_{k_1} \\ M_{k_1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -2M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & J_{n+1} \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & M_{k_1} \\ M_{k_1} & J_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ J_{n+1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & J_{n+1} \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & M_{k_1} \\ M_{k_1} & J_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -2M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ I_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & I_{n+1} \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & M_{k_1} \\ M_{k_1} & I_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} M_{k_1} & M_{k_1} \\ I_{n+1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & I_{n+1} \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & M_{k_1} \\ M_{k_1} & I_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} M_{k_1}^* & M_{k_1}^* \\ M_{k_1}^* & -2M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & M_{k_1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & J_{n+1} \\ J_{n+1} & M_{k_1}^* \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & J_{n+1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & M_{k_1}^* \\ J_{n+1} & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & M_{k_1} \\ J_{n+1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & J_{n+1} \\ J_{n+1} & M_{k_1}^* \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & J_{n+1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & M_{k_1}^* \\ J_{n+1} & -2M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & I_{n+1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & J_{n+1} \\ J_{n+1} & I_{n+1} \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & J_{n+1} \\ J_{n+1} & I_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & I_{n+1} \\ J_{n+1} & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & I_{n+1} \\ J_{n+1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & J_{n+1} \\ J_{n+1} & I_{n+1} \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & J_{n+1} \\ J_{n+1} & I_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & I_{n+1} \\ J_{n+1} & -2M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & J_{n+1} \\ J_{n+1} & M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -M_{k_1}^* & J_{n+1} \\ J_{n+1} & J_{n+1} \end{bmatrix}$			$\begin{bmatrix} M_{k_1} & J_{n+1} \\ J_{n+1} & J_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & J_{n+1} \\ J_{n+1} & -M_{k_1}^* \end{bmatrix}$	
	$\begin{bmatrix} J_{n+1} & J_{n+1} \\ J_{n+1} & 2M_{k_1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} -2M_{k_1}^* & J_{n+1} \\ J_{n+1} & J_{n+1} \end{bmatrix}$			$\begin{bmatrix} 2M_{k_1} & J_{n+1} \\ J_{n+1} & J_{n+1} \end{bmatrix} \xrightarrow{\alpha_k^R} \begin{bmatrix} J_{n+1} & J_{n+1} \\ J_{n+1} & -2M_{k_1}^* \end{bmatrix}$	

Table 4.4: LITA transforms of Class D

	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -X_{n+1} & X_{n+1} \\ M_{n+1}^t & M_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} M_{n+1} & X_{n+1} \\ M_{n+1} & M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & 2M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2X_{n+1} & X_{n+1} \\ M_{n+1}^t & M_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2M_{n+1} & X_{n+1} \\ M_{n+1} & M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -2X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ M_{n+1} & X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & M_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & M_{n+1} \\ X_{n+1} & M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & X_{n+1} \\ M_{n+1}^t & -M_{n+1}^t \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ M_{n+1} & 2X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & M_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2X_{n+1} & M_{n+1} \\ X_{n+1} & M_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} X_{n+1} & M_{n+1}^t \\ M_{n+1}^t & -2M_{n+1}^t \end{bmatrix}$	
	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -X_{n+1} & X_{n+1} \\ M_{n+1}^t & J_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} M_{n+1} & X_{n+1} \\ M_{n+1} & J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & 2J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2X_{n+1} & X_{n+1} \\ M_{n+1}^t & J_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2M_{n+1} & X_{n+1} \\ M_{n+1} & J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -2X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ J_{n+1} & X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & J_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & M_{n+1} \\ X_{n+1} & J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} X_{n+1} & M_{n+1}^t \\ M_{n+1}^t & -M_{n+1}^t \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ J_{n+1} & 2X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & J_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2X_{n+1} & M_{n+1} \\ X_{n+1} & J_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} X_{n+1} & M_{n+1}^t \\ M_{n+1}^t & -2M_{n+1}^t \end{bmatrix}$	
	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -X_{n+1} & X_{n+1} \\ M_{n+1}^t & I_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} M_{n+1} & X_{n+1} \\ M_{n+1} & I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n+1} & M_{n+1} \\ X_{n+1} & 2I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2X_{n+1} & X_{n+1} \\ M_{n+1}^t & I_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2M_{n+1} & X_{n+1} \\ M_{n+1} & I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & -2X_{n+1} \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ I_{n+1} & X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & I_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & M_{n+1} \\ X_{n+1} & I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} X_{n+1} & M_{n+1}^t \\ M_{n+1}^t & -M_{n+1}^t \end{bmatrix}$	
	$\begin{bmatrix} X_{n+1} & M_{n+1} \\ I_{n+1} & 2X_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2M_{n+1}^t & M_{n+1}^t \\ X_{n+1} & I_{n+1}^t \end{bmatrix}$			$\begin{bmatrix} 2X_{n+1} & M_{n+1} \\ X_{n+1} & I_{n+1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} X_{n+1} & M_{n+1}^t \\ M_{n+1}^t & -2M_{n+1}^t \end{bmatrix}$	

Property 4.2.3. Let M_D be any $3^n \times 3^n$ square matrices in Class D. Then, the inverse matrix M_D^{-1} is given by

$$M_D^{-1} = \beta_A [R_E^2(M_C)] . \quad (4.12)$$

There are also two classes that contain 48 forward transform matrices each. They are denoted as Classes E and F separately. Class E is derived from Class C by interchanging the two submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(9)}$. Similarly, Class F is derived from Class D by interchanging submatrices $M_{n-1}^{(3)}$ and $M_{n-1}^{(7)}$. Table 4.5 shows the transform matrices and their corresponding butterfly diagrams of Class E, for the case $n = 2$ and $X_{n-1} = I_{n-1}$. Table 4.6 shows the transform matrices and their corresponding butterfly diagrams of Class F, for the case $n = 2$ and $X_{n-1} = J_{n-1}$.

Property 4.2.4. Let M_E and M_F be any $3^n \times 3^n$ square matrices in Classes E and F, respectively. Then, their inverses are given by

$$M_E^{-1} = \beta_A [M_E] \quad (4.13)$$

and

$$M_F^{-1} = \beta_A [R^2(M_C)] . \quad (4.14)$$

The following properties show the relations of the introduced classes.

Property 4.2.5. For any forward matrix M_F in Class F, a forward matrix M_C can be found in Class C and the following relation exists:

$$R_E[M_F] = M_C. \quad (4.15)$$

Table 4.5: LITA transforms of Class E

$\begin{bmatrix} M_{k+1} & & \\ & M_{k+1} & \\ X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & M_{k+1} \\ -M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & X_{n+1} \\ & M_{k+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -M_{k+1} \\ \beta & M_{k+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} M_{k+1} & & \\ & M_{k+1} & \\ 2X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & M_{k+1} \\ -2M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & 2X_{n+1} \\ & M_{k+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -2M_{k+1} \\ \beta & M_{k+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & M_{k+1} & \\ M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & M_{k+1} \\ -X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & M_{k+1} \\ & M_{k+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -X_{n+1} \\ \beta & M_{k+1} \\ & X_{n+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & M_{k+1} & \\ 2M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & M_{k+1} \\ -2X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & 2M_{k+1} \\ & M_{k+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -2X_{n+1} \\ \beta & M_{k+1} \\ & X_{n+1} \end{bmatrix}$

$\begin{bmatrix} M_{k+1} & & \\ & I_{n+1} & \\ X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & I_{n+1} \\ -M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & X_{n+1} \\ & I_{n+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -M_{k+1} \\ \beta & I_{n+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} M_{k+1} & & \\ & I_{n+1} & \\ 2X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & I_{n+1} \\ -2M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & 2X_{n+1} \\ & I_{n+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -2M_{k+1} \\ \beta & I_{n+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & I_{n+1} & \\ M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & I_{n+1} \\ -X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & M_{k+1} \\ & I_{n+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -X_{n+1} \\ \beta & I_{n+1} \\ & X_{n+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & I_{n+1} & \\ 2M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & I_{n+1} \\ -2X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & 2M_{k+1} \\ & I_{n+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -2X_{n+1} \\ \beta & I_{n+1} \\ & X_{n+1} \end{bmatrix}$

$\begin{bmatrix} M_{k+1} & & \\ & J_{n+1} & \\ X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & J_{n+1} \\ -M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & X_{n+1} \\ & J_{n+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -M_{k+1} \\ \beta & J_{n+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} M_{k+1} & & \\ & J_{n+1} & \\ 2X_{n+1} & X_{n+1} & \end{bmatrix} \begin{bmatrix} M_{k+1} & \\ \beta & J_{n+1} \\ -2M_{k+1} & X_{n+1} \end{bmatrix}$			$\begin{bmatrix} X_{n+1} & 2X_{n+1} \\ & J_{n+1} \\ & M_{k+1} \end{bmatrix} \begin{bmatrix} X_{n+1} & -2M_{k+1} \\ \beta & J_{n+1} \\ & M_{k+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & J_{n+1} & \\ M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & J_{n+1} \\ -X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & M_{k+1} \\ & J_{n+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -X_{n+1} \\ \beta & J_{n+1} \\ & X_{n+1} \end{bmatrix}$

$\begin{bmatrix} X_{n+1} & & \\ & J_{n+1} & \\ 2M_{k+1} & M_{k+1} & \end{bmatrix} \begin{bmatrix} X_{n+1} & \\ \beta & J_{n+1} \\ -2X_{n+1} & M_{k+1} \end{bmatrix}$			$\begin{bmatrix} M_{k+1} & 2M_{k+1} \\ & J_{n+1} \\ & X_{n+1} \end{bmatrix} \begin{bmatrix} M_{k+1} & -2X_{n+1} \\ \beta & J_{n+1} \\ & X_{n+1} \end{bmatrix}$

4.2. $F - R$ type linearly independent ternary arithmetic transforms

154

Table 4.6: LITA transforms of Class F

	$\begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -M_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & 2X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2M_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2X_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -2M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -X_{n-1} & M_{n-1} \\ M_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & X_{n-1} \\ M_{n-1} & -X_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & 2M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2X_{n-1} & M_{n-1} \\ M_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2M_{n-1} & M_{n-1} \\ X_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & X_{n-1} \\ M_{n-1} & -2X_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & M_{n-1} \\ X_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -M_{n-1} & X_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & M_{n-1} \\ X_{n-1} & 2X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2M_{n-1} & X_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2X_{n-1} & X_{n-1} \\ M_{n-1} & J_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -2M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -X_{n-1} & M_{n-1} \\ J_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ J_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} J_{n-1} & X_{n-1} \\ M_{n-1} & -X_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} J_{n-1} & X_{n-1} \\ M_{n-1} & 2M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2X_{n-1} & M_{n-1} \\ J_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2M_{n-1} & M_{n-1} \\ J_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} J_{n-1} & X_{n-1} \\ M_{n-1} & -2X_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & M_{n-1} \\ X_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -M_{n-1} & X_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & I_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & M_{n-1} \\ X_{n-1} & 2X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2M_{n-1} & X_{n-1} \\ I_{n-1} & M_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2X_{n-1} & X_{n-1} \\ M_{n-1} & I_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} M_{n-1} & M_{n-1} \\ X_{n-1} & -2M_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & X_{n-1} \\ M_{n-1} & M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -X_{n-1} & M_{n-1} \\ I_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} M_{n-1} & M_{n-1} \\ I_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & -X_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & X_{n-1} \\ M_{n-1} & 2M_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} -2X_{n-1} & M_{n-1} \\ I_{n-1} & X_{n-1} \end{bmatrix}$			$\begin{bmatrix} 2M_{n-1} & M_{n-1} \\ I_{n-1} & X_{n-1} \end{bmatrix} \leftrightarrow \begin{bmatrix} X_{n-1} & X_{n-1} \\ M_{n-1} & -2X_{n-1} \end{bmatrix}$	

If $\widetilde{M}_C$ and $\widetilde{M}_F$ be any $3^n \times 3^n$ square matrices in Classes C and F, respectively, and $R_E [\widetilde{M}_F] = \widetilde{M}_C$, then,

$$M_C^{-1} = R_E^{-1} [M_F^{-1}]. \quad (4.16)$$

Property 4.2.6. For any forward matrix M_E in Class E, a forward matrix M_D can be found in Class D and the following relation exists:

$$R_E^{-1} [M_E] = M_D. \quad (4.17)$$

If $\widetilde{M}_D$ and $\widetilde{M}_E$ be any $3^n \times 3^n$ square matrices in Classes D and E, respectively, and $R^{-1} [\widetilde{M}_E] = \widetilde{M}_D$, then,

$$M_D^{-1} = R_E [M_E^{-1}]. \quad (4.18)$$

Example 4.2.2. Let us consider the ternary function $\vec{F} = \begin{bmatrix} 1 & 2 & 0 & 2 & 1 & 1 & 2 & 1 & 0 \end{bmatrix}^T$. Let a $F - R$ type LITA transform matrix belong to Class C, e.g.,

$$M_C = \begin{bmatrix} I_{n-1} & O_{n-1} & 2M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}.$$

For $n = 2$,

$$M_C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and using Property 4.2.2 from (4.11), the inverse transform matrix is

$$M_C^{-1} = \beta_A[M_C] = \begin{bmatrix} I_{n-1} & O_{n-1} & -2I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Interchanging submatrices $M_{n-1}^{(3)}$ and $M_{n-1}^{(7)}$ inside M_C will bring it into Class

E, i.e.,

$$M_E = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ 2M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}.$$

For $n = 2$,

$$M_E = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix},$$

and using Property 4.2.4 from (4.13), the inverse transform matrix is

$$M_E^{-1} = \beta_A [M_E] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 1 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -2 & 0 & 0 & 0 & -2 & 0 & 1 \end{bmatrix}.$$

By (4.3) from Definition 4.1.3,

$$\begin{aligned}\vec{A}_C &= M_C^{-1} \cdot \vec{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 0 \end{bmatrix}^T; \\ \vec{A}_E &= M_E^{-1} \cdot \vec{F} = \begin{bmatrix} 1 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & -1 \end{bmatrix}^T.\end{aligned}$$

It is clear that for the particular truth vector, the transform matrix in Classes C and E all yield a simpler LITA spectrum with smaller number of non-zero elements. The transform matrix in Class C is even better than Class E. From Definition 4.1.2, the LITA expansion is

$$f_C = g_4 + g_5 + 2 \cdot g_6 + g_7$$

where the ternary basis functions are

$$\begin{aligned}g_4 &= x_1^1 \cdot x_0^1 \\ g_5 &= 2x_1^1 \cdot x_0^0 + x_1^1 \cdot x_0^2 \\ g_6 &= 2x_1^0 \cdot x_0^0 + x_1^2 \cdot x_0^0 \\ g_7 &= 2x_1^0 \cdot x_0^1 + x_1^2 \cdot x_0^1.\end{aligned}$$

In the above equations, the symbol x_j^i represents ternary variables and the superscript i refers to the corresponding ternary values, and all the operations are performed in standard algebra.

Let us compare the obtained result with one transform matrix selected from

Class A

$$M_A = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ 2M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}$$

with its inverse

$$M_A^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ -2M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}.$$

Then, $\vec{A}_A = M_A^{-1} \cdot \vec{F} = \begin{bmatrix} 1 & 2 & -2 & 2 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T$. It is obvious that the better LITA expansion is obtained from the transform in Class C. $\square$

4.2.2 $F - R$ type LITA transforms with permutations

Applying horizontal and vertical permutations, there are some other $F - R$ LITA transform matrices, which are also defined by recursive equations. They are categorized into Classes $A_H - F_H$ and Classes $A_V - F_V$ for the horizontal and vertical types, respectively.

Property 4.2.7. Let M_n be any nonsingular square $3^n \times 3^n$ matrix with arithmetic inverse M_n^{-1} . Then,

$$[\mu_H [M_n]]^{-1} = \mu_V [M_n^{-1}] \quad (4.19)$$

and

$$[\mu_V [M_n]]^{-1} = \mu_H [M_n^{-1}]. \quad (4.20)$$

Property 4.2.8. Let M_n be a recursive transform matrix with arithmetic inverse M_n^{-1} , where $M_n \in Y$ with Class $Y \in \{A, B, C, D, E, F\}$. Then, from

Definitions 4.1.1 and 4.1.4, and Property 4.2.7,

- $\mu_V [M_n] \in Y_V$ with $Y_V \in \{A_V, B_V, C_V, D_V, E_V, F_V\}$;
- $\mu_H [M_n] \in Y_H$ with $Y_H \in \{A_H, B_H, C_H, D_H, E_H, F_H\}$.

The inverse matrices of the transforms are evaluated by (4.19) and (4.20) of Property 4.2.7.

4.2.3 Experimental results

Four selected transform matrices from Classes A, C, and E have been implemented using Microsoft Visual C++ and run on PIII 500 MHz computer with 128 MB RAM. They are run on several benchmark functions that have been modified to represent ternary functions instead of original MCNC, IWLS'93 and ISCAS'95 binary benchmark functions as described in Chapter 3.

Four selected transform matrices are

M_{A1} selected from Class A1

$$M_{A1} = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, M_{A1}^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ -M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix};$$

M_{A2} selected from Class A2

$$M_{A2} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}, M_{A2}^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ -M_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix};$$

M_C selected from Class C

$$M_C = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}, M_C^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ -I_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix};$$

M_E selected from Class E

$$M_E = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, M_E^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ -I_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}.$$

The number of non-zero spectral coefficients in some ternary benchmark functions for four selected transform matrices will be compared with the results of only known multi-polarity ternary arithmetic (TAR) transform matrix.

Table 4.7 gives the number of non-zero spectral coefficients of the four LITA transforms and ternary best polarity as well as zero polarity arithmetic transforms for some ternary benchmark functions. This table shows that almost for each selected LITA transform, we can find ternary benchmark function where such a LITA transform provides a spectrum with fewer number of non-zero spectral coefficients.

Comparing the experimental results among the four selected LITA transform matrices in more details, M_{A2}^{-1} gets very close numbers of non-zero spectral coefficients compared with the results of M_E^{-1} for almost all benchmark circuits; M_C^{-1} always has much smaller numbers of non-zero spectral coefficients than M_E^{-1} ; For some ternary benchmark circuits (con1, squar5, xor5, and z9sym), M_{A1}^{-1} gets very few non-zero spectral coefficients when compared with the other

Table 4.7: Number of non-zero spectral coefficients for LITA and TRM transforms.

TBF	M_{A1}^{-1}	M_{A2}^{-1}	M_C^{-1}	M_E^{-1}	TAR^{-1}	$TAR_{<0>}^{-1}$
					Best polarity	Zero polarity
alu4	1938	1985	1731	1987	1362	2055
apex4	216	162	157	162	142	239
clip	171	154	147	153	109	234
con1	23	38	26	39	17	46
ex5	81	77	71	77	55	81
ex1010	237	187	189	189	236	239
inc	59	54	53	54	44	81
misex1	71	62	60	63	31	62
squar5	13	16	16	16	10	24
xor5	7	11	12	12	5	7
z5xp1	54	53	53	53	27	77
z9sym	10	96	88	103	16	20

three LITA transforms. It can be explained by the fact that M_{A1}^{-1} contains no submatrix X_{n-1} and more final spectral coefficients result in zero.

Let us also compare the results of the four LITA transform matrices with ternary arithmetic transform. It is obvious that the four LITA transform matrices can always get better results of reducing the spectral coefficients of all the benchmark functions compared with the results of ternary arithmetic transform for zero polarity. After searching through all the 3^n polarities, the ternary arithmetic transform can get better results (smaller number of non-zero spectral coefficients) than the LITA transform matrices and is the most efficient. However, for some ternary benchmark functions (ex1010, squar5, xor5, and z9sym), M_{A1}^{-1} produces spectrum that has fewer or close number of non-zero coefficients similar to the results of ternary arithmetic transform in best polarity. It should be also noticed that in order to obtain the best polarity result for ternary arithmetic transform all 3^n spectra have to be calculated, and it always takes a lot

of time to search through all the polarities of ternary arithmetic transform. On the other hand, the calculation of the LITA transforms needs to be performed only once for each of them and due to that fact the calculation processes for the LITA transforms are always much faster than those for ternary arithmetic transform.

Table 4.8 shows the calculation time for the four LITA transforms and the calculation time for searching through all the polarities as well as the time to calculate only the zero polarity for ternary arithmetic transform. It is obvious that the calculation times to search for the best polarity results for the ternary arithmetic transform are so much longer than the corresponding times for calculating LITA transforms. For the four selected LITA transforms, the calculation times for M_{A2}^{-1} , M_C^{-1} , and M_E^{-1} are very short and similar to each other. The calculation times of M_{A1}^{-1} are longer than for the other three LITA transform matrices but still much shorter than the calculation times for obtaining zero polarity spectral coefficients for the ternary arithmetic transform.

Table 4.8: Calculation times in seconds for LITA transforms.

TBF	M_{A1}^{-1}	M_{A2}^{-1}	M_C^{-1}	M_E^{-1}	TAR^{-1}	$TAR_{<0>}^{-1}$
					Best polarity	Zero polarity
alu4	2.05	0.55	0.58	0.56	3500.32	2.08
apex4	0.12	0.08	0.08	0.07	27.01	0.12
clip	0.05	0.02	0.02	0.02	9.73	0.17
con1	0.01	0.01	0.01	0.01	2.23	0.04
ex5	0.51	0.48	0.48	0.50	27.38	0.52
ex1010	0.06	0.03	0.04	0.03	14.53	0.06
inc	0.02	0.02	0.02	0.02	5.67	0.05
misex1	0.02	0.02	0.01	0.01	4.87	0.04
squar5	0.01	< 0.01	0.01	0.01	2.39	0.01
xor5	0.01	< 0.01	< 0.01	0.01	0.65	0.05
z5xp1	0.01	0.01	0.01	0.01	5.72	0.05
z9sym	0.01	0.01	0.01	0.01	4.99	0.04

Till now we have compared selected LITA transforms from Classes A, C and E. In order to justify introduction of classes with permutation, we will compare the LITA transforms from classes with and without permutation. Hence, three other transform matrices M_{B1} , M_D , and M_F are selected from Classes B1, D, and F, respectively. Their forward and inverse transform matrices are

M_{B1} from Class B1

$$M_{B1} = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, M_{B1}^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix};$$

M_D from Class D

$$M_D = \begin{bmatrix} M_{n-1} & O_{n-1} & J_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, M_D^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ J_{n-1} & O_{n-1} & -J_{n-1} \end{bmatrix};$$

M_F from Class F

$$M_F = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, M_F^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & -J_{n-1} \end{bmatrix}.$$

Referring to Tables 4.1 - 4.6, it is clear that the inverse transform matrices of M_{B1} , M_D , and M_F are horizontal flipping of the inverse transform matrices of M_{A1} , M_C , and M_E , respectively. The inverse butterfly diagrams of M_{B1} , M_D , and M_F are the same as M_{A1} , M_C , and M_E , except that each butterfly diagram of M_{B1} , M_D , and M_F have an additional permutation part on the

left side. In Table 4.9, the numbers of non-zero spectral coefficients for the six LITA transform matrices are listed. It can be seen from this table that for some benchmark functions, the non-zero spectral coefficients' numbers of M_{B1}^{-1} , M_D^{-1} , and M_F^{-1} , written in bold italic, are much smaller than the related numbers of M_{A1}^{-1} , M_C^{-1} , and M_E^{-1} , respectively. Hence, the additional permutation part of the butterfly diagrams plays an important role in reducing the non-zero spectral coefficients for some ternary benchmark functions. It should be noticed that the calculation time of all the spectral coefficients is very similar for all the classes presented in Table 4.9 and hence not shown here. We consider this analysis as a justification for the introduction and the usage of different LITA transforms supported by presented results that these transforms may share some useful properties and are much faster in calculation than the only known multi-polarity ternary arithmetic transform.

Table 4.9: Number of non-zero spectral coefficients for LITA transforms.						
TBF	M_{A1}^{-1}	M_{B1}^{-1}	M_C^{-1}	M_D^{-1}	M_E^{-1}	M_F^{-1}
alu4	1938	1789	1731	1737	1987	1960
apex4	216	162	157	162	162	162
b12	873	213	3384	3158	3984	4028
clip	171	133	147	147	153	157
con1	23	20	26	27	39	41
ex5	81	81	71	70	77	77
inc	59	55	53	51	54	54
misex1	71	50	60	69	63	62
t481	4747	1283	2468	2471	3275	3131
table3	967	779	858	416	700	414
xor5	7	19	12	13	12	12
z5xp1	54	54	53	54	53	54

4.3 F^2 type linearly independent ternary arithmetic transforms

Different new classes of F^2 type Linearly Independent Ternary Arithmetic (LITA) transforms being the bases of ternary arithmetic polynomial expansions are introduced in this section. Recursive equations defining this type of LITA transforms and their corresponding butterfly diagrams are shown. Various properties and relations between introduced classes of F^2 type LITA transforms are discussed. Computational costs to calculate their spectra and applications of corresponding polynomial expansions in logic design are also discussed.

4.3.1 Family of F^2 type LITA transforms

In this section, we introduced a family of new transforms that are divided into some classes. The basic transform matrices are constructed by the basic recursive submatrices O_{n-1} , X_{n-1} and M_{n-1} , where O_{n-1} is a $3^{n-1} \times 3^{n-1}$ matrix with all its elements 0, I_{n-1} is a $3^{n-1} \times 3^{n-1}$ identity matrix, J_{n-1} is a $3^{n-1} \times 3^{n-1}$ reverse-identity matrix, i.e., it has elements in the reverse-diagonal position equal to 1 while all other elements are 0 and X_{n-1} is either I_{n-1} or J_{n-1} .

Let M_n be a $3^n \times 3^n$ square matrix as specified in Definition 4.1.4 having one submatrix M_{n-1} in one corner $M_{n-1}^{(1)}$, and three submatrices I_{n-1} are at $M_{n-1}^{(5)}$, $M_{n-1}^{(6)}$ and $M_{n-1}^{(8)}$. The remaining five submatrices of M_n are O_{n-1} . Then one of

the basic F^2 type LITA transform is defined by

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.21)$$

For $n = 2$,

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Its forward butterfly diagram is shown in the left part of Fig. 4.3. In all the butterfly diagrams presented in this section, the solid and dotted lines represent addition and subtraction while the narrow and broad lines represent the value 1 and 2, respectively. It is obvious that due to the fact that many of the connections in that butterfly diagram are direct connections not requiring any calculation the whole butterfly diagram can be executed in one time period what speeds up its computation and the final butterfly diagram can be represented as shown in the right part of Fig. 4.3.

Due to the recursive definition, similar factorization and appearance of but-

Figure 4.3: Forward butterfly diagram of $M_{(4.21)}$.

terfly diagrams applies for higher dimensions. For example for any n , the butterfly diagram of $n - 1$ variables will stay on the top and other two parts that have the same size of $n - 1$ will be placed below and there is the crossing between them.

The inverse matrix in standard algebra for (4.21) is defined by

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & -I_{n-1} \end{bmatrix}. \quad (4.22)$$

There are total eight basic forward transform matrices which are categorized as Class S . All of them can be derived from (4.22). One of them is obtained by multiplying 2 over the central submatrix I_{n-1} as

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.23)$$

By executing the two operators μ_{EH} and μ_{EV} over (4.21) and (4.23) sepa-

rately, two more forward transform matrices can be derived.

From (4.21), by interchanging the central submatrix $M_{n-1}^{(5)}$ and the submatrix $M_{n-1}^{(9)}$ that is opposite to M_{n-1} , the fifth basic forward transform matrix is given by

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}. \quad (4.24)$$

Similarly, other three basic forward transform matrices can be derived from (4.24) by using the following operations together or separately: multiplying the central submatrix by 2 or executing the operators μ_{EH} and μ_{EV} over the matrix. Their inverse matrices can be derived from (4.22) in a similar manner that the forward matrices are obtained. From the ways of constructions of Class S , the following property can be derived.

Property 4.3.1. Let $M_S = M_n$ be one of the square matrices in Class S . Then, the inverse matrix of M_S can be obtained by selecting the central submatrix and the submatrix that is opposite to M_{n-1} , after that they are multiplied by -1 and interchanged with each other.

The entire recursive basic forward and inverse transform matrices of Class S and their fast butterfly diagrams for $n = 2$ are shown in Table 4.10.

Horizontal-flipping of the transform matrix based on (4.21) will bring it into Class P . The new transform matrix is defined by

$$M_n = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}, \quad (4.25)$$

Table 4.10: Transform matrices and butterfly diagrams for Class S .

	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & -I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & -I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & -I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & 2I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & -2I_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & 2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} -2I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	
	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & 2I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & -2I_{n-1} & I_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} 2I_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & -2I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$	

and its inverse transform matrix can be derived from the transform matrix of (4.22) by vertical-flipping as follows

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & J_{n-1} & -J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.26)$$

Table 4.11 gives the list of eight forward and inverse F^2 type LITA transforms in Class P and their corresponding butterfly diagrams for $n = 2$. The following properties of Class P apply.

Property 4.3.2. Let $M_P = M_n$ be one of the square matrices in Class P . Then, the inverse of M_P is given by:

1. Selecting the central submatrix and the submatrix which is opposite to M_n , then multiplying them by -1 and interchanging them;
2. Executing the two operators μ_{EH} and μ_{EV} over the matrix.

Property 4.3.3. For any forward matrix M_P in Class P , a forward matrix M_S can be found in Class S and the following relations exist:

$$\mu_{EH}[M_S] = M_P \quad (4.27)$$

or

$$\mu_{EV}[M_S] = M_P. \quad (4.28)$$

Replacing all the submatrices M_{n-1} and I_{n-1} inside (4.21) creates a new

Table 4.11: Transform matrices and butterfly diagrams for Class P .

	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ -J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & -J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & -J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & -J_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & 2J_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ -2J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & 2J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & -2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & J_{n-1} & 2J_{n-1} \\ O_{n-1} & O_{n-1} & J_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ J_{n-1} & -2J_{n-1} & O_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix}$	
	$\begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \\ 2J_{n-1} & J_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & J_{n-1} & O_{n-1} \\ O_{n-1} & -2J_{n-1} & J_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	

transform matrix defined as

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix}, \quad (4.29)$$

and its inverse matrix is given by

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & -M_{n-1}^{-1} \end{bmatrix}, \quad (4.30)$$

that is again similar to (4.22) with the same replacement as for the forward transform matrix.

From the other seven forward transform matrices of Class S , by replacing the submatrices M_{n-1} and I_{n-1} , the Class S^* can be derived. Similarly, the Class P^* is defined by replacing the submatrices M_{n-1} and J_{n-1} for all the transform matrices of Class P .

Fig. 4.4 shows the forward and inverse butterfly diagrams of (4.29) and (4.30), respectively.

Comparing Figs. 4.3 and 4.4, it is clear that their second parts are the same. Even for higher n , the only difference between them exists in the first part of the butterfly diagrams. Class P and Class P^* also have the same relations.

Classes S^* and P^* have also similar properties to Properties 4.3.1 – 4.3.2.

Property 4.3.4. Let $M_{S^*} = M_n$ be one of the square matrices in Class S^* . Then, the inverse matrix of M_{S^*} can be obtained by selecting the central sub-

Figure 4.4: Forward and inverse butterfly diagrams of $M_{(4.29)}$.

matrix and the submatrix that is opposite to I_{n-1} , after that they are multiplied by -1 and interchanged with each other.

Property 4.3.5. Let $M_{P^*} = M_n$ be one of the square matrices in Class P^* . Then, the inverse of M_{P^*} is given by

1. Selecting the central submatrix and the submatrix which is opposite to I_n , then multiplying them by -1 and interchanging them;
2. Executing the two operators μ_{EH} and μ_{EV} over the matrix.

Property 4.3.4 is valid also for Class S^* and P^* with the only change of symbols S and P to symbols S^* and P^* accordingly.

In what follows, we will concentrate on other possible related F^2 type LITA families. From (4.21), by interchanging the locations and numbers of different submatrices, some other classes of new transforms can be developed.

For example, by interchanging the submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(6)}$ in (4.27), a

new transform is defined as

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.31)$$

Another similar new transform is obtained by interchanging the two submatrices $M_{n-1}^{(1)}$ and $M_{n-1}^{(8)}$ in (4.21), as defined below,

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.32)$$

The following example shows the relations for calculating the two new transforms of higher sizes.

Example 4.3.1. The symbols $M_{n-(a)}$ and $M_{n-(b)}$ are used here to denote the two new transforms defined by (4.31) and (4.32), respectively.

$$A(n) = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix},$$

$$B(2) = \begin{bmatrix} I_1 & O_1 & O_1 \\ O_1 & I_1 & I_1 \\ O_1 & I_1 & O_1 \end{bmatrix}, B(n) = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & B(n-1) \end{bmatrix}.$$

For $n = 2$,

$$M_{2-(a)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} = B(2) \times A(2),$$

$$M_{2-(b)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} = A(2) \times B(2).$$

Figs. 4.5 and 4.6 give the forward butterfly diagrams of $M_{2-(a)}$ and $M_{2-(b)}$, respectively.

It is obvious that the corresponding parts of $M_{n-(a)}$ and $M_{n-(b)}$ have the same butterfly structures, but the order of the parts is opposite. Even for higher n , they all have regular parts and keep the same relations. $\square$

Figure 4.5: Forward butterfly diagram of $M_{n-(a)}$.

Figure 4.6: Forward butterfly diagram of $M_{n-(b)}$.

The inverse transform matrices of $M_{n-(a)}$ and $M_{n-(b)}$ are given by replacing the central submatrices I_{n-1} with the submatrices $-M_{n-1}$ and interchanging the pairs of submatrices: $M_{n-1}^{(5)}$ and $M_{n-1}^{(9)}$, $M_{n-1}^{(6)}$ and $M_{n-1}^{(8)}$, as shown in the following two equations, respectively.

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & -M_{n-1}^{-1} \end{bmatrix}, \quad (4.33)$$

and

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & -M_{n-1}^{-1} \end{bmatrix}. \quad (4.34)$$

It is clear that the corresponding butterfly diagrams of two inverse transforms also have the same structures with opposite order of separate butterflies.

Based on (4.31) and (4.32), other two new transforms are introduced by interchanging the central submatrices I_{n-1} with the submatrices M_{n-1} , as defined by

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \end{bmatrix}, \quad (4.35)$$

and

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.36)$$

Their inverse transform matrices can be derived from (4.33) and (4.34) by replacing the submatrices $-M_{n-1}$ in the location $M_{n-1}^{(9)}$ by the submatrices $-I_{n-1}$, as shown in the following equations:

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & -I_{n-1} \end{bmatrix}, \quad (4.37)$$

and

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & I_{n-1} & -I_{n-1} \end{bmatrix}. \quad (4.38)$$

Multiplying 2 over the central submatrix $M_{n-1}^{(5)}$ will bring other four new transform matrices. Further more, by interchanging the central submatrix $M_{n-1}^{(5)}$ and the submatrix $M_{n-1}^{(9)}$, and executing the two operators μ_{EH} and μ_{EV} over the matrices, altogether 32 pairs of LITA forward and inverse transform matrices can be derived, and they are categorized as Class T .

All the inverse transform matrices of Class T can be derived from (4.33), (4.34) and (4.37), (4.38) following the same steps that have been applied to the forward transform matrices changes.

Similarly to the relations between Class S and P , another class of new fast transforms is derived from Class T by making horizontal-flipping of the transform matrices. Altogether 64 derived transform matrices are categorized as Class Q . The transforms of Class Q have similar structure of their butterfly diagrams to the corresponding transforms in Class P . The difference is that each of the matrices in Class Q has an additional permutation part.

Property 4.3.4 is valid also for Class T and Q with the only change of symbols S and P to T and Q , respectively.

There are still two classes of new F^2 type LITA transform matrices which are denoted as Class T^* and Class Q^* . They are derived from Class T and Class Q , respectively, by the following two steps:

- Replacing all submatrices M_{n-1} by I_{n-1} (for Class T), or by J_{n-1} (for Class Q);
- Replacing all submatrices I_{n-1} (for Class T) or J_{n-1} (for Class Q) by M_{n-1} .

It is obvious that Class T^* and Q^* have similar properties as Class T and Q . Each of them contains 32 pairs of forward and inverse transform matrices.

All the eight classes, Class S , P , T and Q , and Class S^* , P^* , T^* and Q^* form a family of F^2 type LITA transforms which possess fast algorithms in forward and inverse transform matrices. All the introduced F^2 type LITA transforms are classified according to their mathematical relationship of forward transform matrices with their respective inverse transform matrices. They are constructed through four basic submatrices, the zero submatrix O_{n-1} , the identity submatrix I_{n-1} , the reverse-identity submatrix J_{n-1} and the recursive submatrix M_{n-1} in which general rules applies to them, according to their classifications.

4.3.2 Computation complexity consideration

Due to their similar basic constructions, the eight classes of F^2 type LITA transform matrices can be categorized into two groups. One contains four Classes, Class

S , P , T and Q , and the other one contains the remaining four classes, Class S^* , P^* , T^* and Q^* . The four classes of each group have the same computation complexity. The formula presented in the following equation can be used to calculate the total number of arithmetic additions/subtractions required to compute the transform of an n -variable ternary function of all the eight classes.

$$S_n = S_{n-1} + 3^{n-1}. \quad (4.39)$$

For Class S , P , T and Q , since $S_1 = 1$, and for arbitrary n ,

$$S_n = (3^n - 1)/2. \quad (4.40)$$

For Class S^* , P^* , T^* and Q^* , since $S^*_1 = 2$, and for arbitrary n ,

$$S^*_n = (3^n + 1)/2. \quad (4.41)$$

4.4 Ternary arithmetic helix transforms

Classes of linearly independent ternary arithmetic transforms in standard algebra called ternary arithmetic helix transforms are introduced in this section. Four types of helix transform matrices with detailed recursive equations are shown. Various properties, mutual relationships among transform matrices and spectra as well as results of helix transforms for some special cases of ternary logic functions are discussed. Computational costs of the calculation for new transforms are also presented.

4.4.1 Properties and relations

There are altogether four classes of ternary arithmetic helix transforms that are given in Tables 4.12 – 4.15 with the name *Left-Positive-Helix* (Class LP), *Right-Positive-Helix* (Class RP), *Left-Negative-Helix* (Class LN) and *Right-Negative-Helix* (Class RN). Each class includes five pairs of forward and inverse helix transform matrices which are denoted from $a - e$ in the tables. All helix transforms are $3^n \times 3^n$ square transform matrices which are constructed from $3^{n-1} \times 3^{n-1}$ submatrices H_{n-1} and O_{n-1} , I_{n-1} and J_{n-1} . The submatrix O_{n-1} is a zero submatrix with all its elements equal to zero. The submatrix I_{n-1} is an identity submatrix with all its non-zero elements '1' in the diagonal of the matrix while the submatrix J_{n-1} is the reverse-identity submatrix with all its non-zero elements '1' in the reverse diagonal of the matrix. In this section, some useful properties and relations between forward and inverse transforms of the ternary arithmetic helix transforms will be analyzed.

The first helix transform LPa in Class LP is given below

$$H_n = \begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix}. \quad (4.42)$$

It contains only two types submatrices, H_{n-1} and O_{n-1} , and is recursively defined. the parameter x belongs to the set $(1, 2)$. The inverse transform matrix of LPa is obtained by multiplying -1 over all $3^{n-1} \times 3^{n-1}$ submatrices, which

Table 4.12: Class LP of ternary arithmetic helix transforms.

a		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -xH_{n-1}^{-1} & H_{n-1}^{-1} & -xH_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
b		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ xI_{n-1} & I_{n-1} & xI_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -xH_{n-1}^{-1} & I_{n-1} & -xH_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
c		$\begin{bmatrix} H_{n-1} & O_{n-1} & O_{n-1} \\ xH_{n-1} & I_{n-1} & xH_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -xI_{n-1} & I_{n-1} & -xI_{n-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}$
d		$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ xI_{n-1} & H_{n-1} & xI_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ -xH_{n-1}^{-1} & H_{n-1}^{-1} & -xH_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} \\ -xI_{n-1} & H_{n-1}^{-1} & -xI_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} \end{bmatrix}$

are not in the diagonal of the matrix, as shown below.

$$H_n^{-1} = \begin{bmatrix} H_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -xH_{n-1}^{-1} & H_{n-1}^{-1} & -xH_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & H_{n-1}^{-1} \end{bmatrix}. \quad (4.43)$$

Generally, the inverse helix transform matrices of Class LP can be obtained from their corresponding forward helix transform matrices following Rule 4.4.1.

Rule 4.4.1.

1. If the matrix does not contain the submatrix I_{n-1} , then go to Step 3, else
2. Select the two non-zero submatrices that are not in the diagonal of the matrix, interchange them from H_{n-1} to I_{n-1} , or from I_{n-1} to H_{n-1} ($H_{n-1} \Leftrightarrow I_{n-1}$), and
3. Multiply the two non-zero submatrices that are not in the diagonal of the matrix by -1 .

Rule 4.4.1 also applies to the five pairs of forward and inverse helix transform matrices in Class RP, as presented in Table 4.13.

All helix transform matrices in Class LN can be recognized as the results of horizontal flipping on both forward and inverse transform matrices in Class LP.

The first helix transform LN_a in Class LN is given below,

$$H_n = \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.44)$$

Table 4.13: Class RP of ternary arithmetic helix transforms.

a		$\begin{bmatrix} H_{n-1} & xH_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & xH_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & -xH_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & -xH_{n-1}^{-1} & H_{n-1}^{-1} \end{bmatrix}$	
b		$\begin{bmatrix} H_{n-1} & xI_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & xI_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & -xH_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & -xH_{n-1}^{-1} & H_{n-1}^{-1} \end{bmatrix}$	
c		$\begin{bmatrix} H_{n-1} & xH_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & xH_{n-1} & H_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} H_{n-1}^{-1} & -xI_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & -xI_{n-1} & H_{n-1}^{-1} \end{bmatrix}$	
d		$\begin{bmatrix} I_{n-1} & xI_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & xI_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & -xH_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & -xH_{n-1}^{-1} & I_{n-1} \end{bmatrix}$	
e		$\begin{bmatrix} I_{n-1} & xH_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ O_{n-1} & xH_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & -xI_{n-1} & O_{n-1} \\ O_{n-1} & H_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & -xI_{n-1} & I_{n-1} \end{bmatrix}$	

The parameter x is also used in this class. The inverse transform matrix of LNa is obtained by multiplying -1 over all $3^{n-1} \times 3^{n-1}$ submatrices, which are not in the diagonal of the matrix, as shown below,

$$H_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^{-1} \\ -xH_{n-1}^{-1} & H_{n-1}^{-1} & -xH_{n-1}^{-1} \\ H_{n-1}^{-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (4.45)$$

In these transform matrices, only two types of non-zero submatrices H_{n-1} and the reverse-identity submatrix J_{n-1} exist. Rule 4.4.2 shows the relations between the forward and inverse transform matrices of Class LN.

Rule 4.4.2.

1. If the matrix does not contain the submatrix J_{n-1} , then go to Step 3, else
2. Select the two non-zero submatrices that are not in the diagonal of the matrix, interchange them from H_{n-1} to J_{n-1} , or from J_{n-1} to H_{n-1} ($H_{n-1} \Leftrightarrow J_{n-1}$), and
3. Multiply the two non-zero submatrices that are not in the diagonal of the matrix by -1 .

The relations between the forward and inverse transform matrices of Class RN presented in Table 4.15 also follow Rule 4.4.2.

The following general matrix form is defined to present the properties of the

Table 4.14: Class LN of ternary arithmetic helix transforms.

a		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^1 \\ -xH_{n-1}^1 & H_{n-1}^1 & -xH_{n-1}^1 \\ H_{n-1}^1 & O_{n-1} & O_{n-1} \end{bmatrix}$
b		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ xJ_{n-1} & J_{n-1} & xJ_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^1 \\ -xH_{n-1}^1 & J_{n-1} & -xH_{n-1}^1 \\ H_{n-1}^1 & O_{n-1} & O_{n-1} \end{bmatrix}$
c		$\begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1} \\ xH_{n-1} & J_{n-1} & xH_{n-1} \\ H_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & H_{n-1}^1 \\ -xJ_{n-1} & J_{n-1} & -xJ_{n-1} \\ H_{n-1}^1 & O_{n-1} & O_{n-1} \end{bmatrix}$
d		$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ xJ_{n-1} & H_{n-1} & xJ_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ -xH_{n-1}^1 & H_{n-1}^1 & -xH_{n-1}^1 \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ xH_{n-1} & H_{n-1} & xH_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & J_{n-1} \\ -xJ_{n-1} & H_{n-1}^1 & -xJ_{n-1} \\ J_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

helix transforms to be discussed in the following parts.

$$H_n = \begin{bmatrix} H_{0,0} & H_{0,1} & \cdots & H_{0,3^n-1} \\ H_{1,0} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ H_{3^n-1,0} & \cdots & \cdots & H_{3^n-1,3^n-1} \end{bmatrix}. \quad (4.46)$$

Based on the analysis of the forward and inverse matrices for ternary arithmetic helix transforms, where for the size n , the number of row i and column j all belong to the set $\left(0, \dots, 3^n - 1\right)$, the following properties can be obtained

when the parameter $x = 1$.

Property 4.4.1. In the forward transform matrices LPa and LNa , the sum of all the elements located in each even number of row i is equal to 1 while the sum of all the elements in each odd number of row i is equal to 0 except that the sum in the central row is also equal to 0.

The row number i of the helix transforms can be represented as ternary number. The total number of non-zero elements in each row is associated with the number of '1's in the ternary representation of the row number.

Property 4.4.2. In the forward transform matrices LPa and LNa , the number of non-zero elements in every row is equal to 3^d , where d is the number of '1's in the ternary representation of the row number i .

Example 4.4.1. For LPa transform matrix, when $n=2$,

$$LPa_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \parallel \begin{matrix} 00 \\ 01 \\ 02 \\ 10 \\ 11 \\ 12 \\ 20 \\ 21 \\ 22 \end{matrix} \Rightarrow \left\{ \begin{matrix} 3^0 \\ 3^1 \\ 3^0 \\ 3^1 \\ 3^2 \\ 3^1 \\ 3^0 \\ 3^1 \\ 3^0 \end{matrix} \right\}.$$

There are 9 non-zero elements in row '11' and 3 non-zero elements in rows '01', '10', '12' and '21'. □

Table 4.15: Class RN of ternary arithmetic helix transforms.

a		$\begin{bmatrix} O_{n-1} & xH_{n-1} & H_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ H_{n-1} & xH_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & -xH_{n-1}^1 & H_{n-1}^1 \\ O_{n-1} & H_{n-1}^1 & O_{n-1} \\ H_{n-1}^1 & -xH_{n-1}^1 & O_{n-1} \end{bmatrix}$
b		$\begin{bmatrix} O_{n-1} & xH_{n-1} & H_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1} & xH_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & -xJ_{n-1} & H_{n-1}^1 \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1}^1 & -xJ_{n-1} & O_{n-1} \end{bmatrix}$
c		$\begin{bmatrix} O_{n-1} & xJ_{n-1} & H_{n-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1} & xJ_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & -xH_{n-1}^1 & H_{n-1}^1 \\ O_{n-1} & J_{n-1} & O_{n-1} \\ H_{n-1}^1 & -xH_{n-1}^1 & O_{n-1} \end{bmatrix}$
d		$\begin{bmatrix} O_{n-1} & xJ_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ J_{n-1} & xJ_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & -xH_{n-1}^1 & J_{n-1} \\ O_{n-1} & H_{n-1}^1 & O_{n-1} \\ J_{n-1} & -xH_{n-1}^1 & O_{n-1} \end{bmatrix}$
e		$\begin{bmatrix} O_{n-1} & xH_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1} & O_{n-1} \\ J_{n-1} & xH_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & -xJ_{n-1} & J_{n-1} \\ O_{n-1} & H_{n-1}^1 & O_{n-1} \\ J_{n-1} & -xJ_{n-1} & O_{n-1} \end{bmatrix}$

Property 4.4.3. In the inverse transform matrices LPa^{-1} and LNa^{-1} , the sum of all the elements located in each even number of the row i is equal to 1 while the sum of all the elements in each odd number of the row i is equal to -1 .

Property 4.4.2 also applies to the inverse transform matrices LPa^{-1} and LNa^{-1} . The following property gives more detail on LPH^{-1} and LNH^{-1} transform matrices.

Property 4.4.4. In the inverse transform matrices LPa^{-1} and LNa^{-1} , the number of ‘ -1 ’ element in each row is equal to 2^d , where d is the number of ‘ 1 ’s in the ternary representation of the row number i and $d \geq 1$. When $d=0$, there is no ‘ -1 ’ element in the row.

Property 4.4.5. In the forward transform matrices RPa and RNa , the sum of all the elements located in each even number of the row i is equal to 1 while the sum of all the elements in each odd number of the row i is equal to 2.

Property 4.4.6. In the inverse transform matrices RPa^{-1} and RNa^{-1} , the sum of all the elements located in each row i is equal to 0 except that the sum of all the elements located in the central row is equal to 1.

Property 4.4.6 describes that in each row, the number of ‘ 1 ’s is equal to the number of ‘ -1 ’s and only one ‘ 1 ’ exists in the central slot of the central row.

Property 4.4.7. In the forward transform matrices LPa and LNa , the sum of all the elements located in each even number of the column j is equal to 1 while the sum of all the elements in each odd number of the column j is equal to 2.

Property 4.4.8. In the inverse transform matrices LPa^{-1} and LNa^{-1} , the sum of all the elements located in each column is equal to 0 except that the sum of all the elements located in the central column is equal to 1.

Using Property 4.4.6 it can be shown that half non-zero elements in each column are '1's and the other half non-zero elements are '-1's. The central column only contains one non-zero element '1' in the central slot.

Property 4.4.9. In the forward transform matrices RPa and RNa , the sum of all the elements located in each even number of the column j is equal to 1 while the sum of all the elements in each odd number of the column j is equal to 0 except that the sum in the central column is also equal to 0.

The row number j of the helix transforms can also be represented as ternary number. The total number of non-zero elements in each column is associated with the number of '1's in the ternary representation of the column number similar to Property 4.4.2.

Property 4.4.10. In the forward transform matrices RPa and RNa , the number of non-zero elements in each column is equal to $3^{d'}$, where d' is the number of '1's in the ternary representation of the column number j .

Property 4.4.11. In the inverse transform matrices RPa^{-1} and RNa^{-1} , the sum of all the elements located in each even number of the column j is equal to 1 while the sum of all the elements in each odd number of the column j is equal to -1.

Property 4.4.10 also applies to the inverse transform matrices RPa^{-1} and RNa^{-1} . The number of '-1' element in each column of RPa^{-1} and RNa^{-1} follows Property 4.4.12.

Property 4.4.12. In the inverse transform matrices RPa^{-1} and RNa^{-1} , the number of '-1' element in each column is equal to $2^{d'}$, where d' is the number of '1's in the ternary representation of the column number j and $d' \geq 1$. When $d'=0$, there is no '-1' element in this column.

Due to the recursive defined structure of the ternary arithmetic helix transforms and from the properties presented above, some interesting results for ternary arithmetic helix transforms can also be derived. The truth vector $\vec{F}$ of any ternary switching functions must be a $3^n \times 1$ vector and it can be separated into n consecutive 3×1 sub-vectors. Let $\vec{A}_L$ represent the spectral coefficients' vectors of LPa^{-1} and $LN a^{-1}$ and $\vec{A}_R$ represent the spectral coefficients' vectors of RPa^{-1} and RNa^{-1} . If the truth vector $\vec{F}$ does not contain zero elements and the value of all the elements are all '1's or all '2's, then, Properties 4.4.13 – 4.4.15 can be derived.

Property 4.4.13. In the spectral coefficients' vector $\vec{A}_L$, the values of the elements in even rows are positive integers and others are negative integers.

Property 4.4.14. The absolute values of the elements in the spectral coefficients' vector $\vec{A}_L$ are equal to the values of the elements in the truth vector $\vec{F}$ corresponding to the same positions that they are located and can be described by the following equation,

$$|\vec{A}_L| = |\vec{F}|. \quad (4.47)$$

Property 4.4.15. The elements of the spectral coefficients' vector $\vec{A}_R$ are all equal to zero except that the central element is the same as any element in the truth vector $\vec{F}$.

Example 4.4.2. When $n=2$, then the inverse matrix of LPa transform for the

truth vector $\vec{F} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}^T$ is

$$\begin{aligned} \vec{A} &= LPa_2^{-1} \cdot \vec{F} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ -2 \\ 2 \\ -2 \\ 2 \\ -2 \\ 2 \\ -2 \\ 2 \end{bmatrix} \end{aligned}$$

□

Property 4.4.16. Let the truth vector $\vec{F}$ be separated into three $3^{n-1} \times 1$ sub-vectors as $\vec{F} = \begin{bmatrix} \vec{F}_0 & \vec{F}_1 & \vec{F}_2 \end{bmatrix}^T$. In every sub-vector, all the elements are non-zero and have the same value. Let the spectral coefficients vector $\vec{A}$ be also separated into three $3^{n-1} \times 1$ sub-vectors as $\vec{A} = \begin{bmatrix} \vec{A}_0 & \vec{A}_1 & \vec{A}_2 \end{bmatrix}^T$, then the following two relations exist

1. If $\vec{F}_0 = \vec{F}_1 \neq \vec{F}_2$, then $|\vec{A}_0| \neq |\vec{A}_1| = |\vec{A}_2|$;
2. If $\vec{F}_0 \neq \vec{F}_1 = \vec{F}_2$, then $|\vec{A}_0| = |\vec{A}_1| \neq |\vec{A}_2|$.

Example 4.4.3. When $n=2$, then the inverse matrix of LPa transform for the

truth vector $\vec{F} = \begin{bmatrix} 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}^T$ is

$$\begin{aligned} \vec{A} &= LPa_2^{-1} \cdot \vec{F} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1 \\ 2 \\ -2 \\ 2 \end{bmatrix} \end{aligned}$$

□

Looking into the structure of the helix transform LPa , it is clear that all the non-zero elements are distributed symmetrically to the diagonal, which contains all '1's. Due to this special structure, a symmetry property can also be derived below.

Property 4.4.17. If all the elements of the truth vector $\vec{F}$ are distributed in a symmetric way along the central element, then all the elements of the spectral coefficients' vector $\vec{A}$ have similar symmetry along the central element.

There are still some permutation relations among the four ternary arithmetic helix transforms. The following properties are based on the definitions of horizontal and vertical permutation operators.

Property 4.4.18. Based on Definitions 4.1.4, 3.1.7 and 3.1.8, $\mu_{EH} [LPy] = LNy$ and $\mu_{EV} [LPy] = LNy$, where the symbol y can be replaced by anyone from the set $a - e$. The same relations also exist between the inverse transform matrices LPy^{-1} and LNy^{-1} .

Property 4.4.19. Based on Definitions 4.1.4, 3.1.7 and 3.1.8, $\mu_{EH} [RPy] = RNy$ and $\mu_{EV} [RPy] = RNy$, where the symbol y can be replaced by anyone from the set $a - e$. The same relations also exist between the inverse transform matrices RPy^{-1} and RNy^{-1} .

The following property is derived from Properties 4.4.18 and 4.4.19.

Property 4.4.20. For any ternary arithmetic helix transform H_n and its inverse transform H_n^{-1} ,

$$\begin{aligned}\mu_{EH} [\mu_{EV} (H_n)] &= \mu_{EV} [\mu_{EH} (H_n)] \\ &= H_n,\end{aligned}\tag{4.48}$$

and

$$\begin{aligned}\mu_{EH} [\mu_{EV} (H_n^{-1})] &= \mu_{EV} [\mu_{EH} (H_n^{-1})] \\ &= H_n^{-1}.\end{aligned}\tag{4.49}$$

Let $\vec{A}$ be a $3^n \times 1$ spectral coefficients' vector for any ternary switching functions. If the operator r is defined as vertical flipping the vector $\vec{A}$ to get the result $\vec{A}^r$, which is also a $3^n \times 1$ spectral coefficients' vector, then the following property can be obtained.

Property 4.4.21. For any spectral coefficients' vector $\vec{A}$ and its vertical per-

mutation $\vec{A}^r$,

$$LP_y \cdot \vec{A} = LN_y \cdot \vec{A}^r, \quad (4.50)$$

and

$$RP_y \cdot \vec{A} = RN_y \cdot \vec{A}^r. \quad (4.51)$$

where the symbol y can be replaced by anyone from the set $a - e$.

Proof. Based on the general matrix form presented in (4.46),

$$LP_{y_n} = \begin{bmatrix} H_{0,0} & H_{0,1} & \cdots & H_{0,3^n-1} \\ H_{1,0} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ H_{3^n-1,0} & \cdots & \cdots & H_{3^n-1,3^n-1} \end{bmatrix},$$

and

$$\vec{A}_n = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{3^n-1} \end{bmatrix}.$$

The vector $\vec{A}^r$ is vertical flipping of the vector $\vec{A}$ and from Property 4.4.18,

$$\begin{aligned}
 LNy \cdot \vec{A}^r &= \mu_{EH}[LPy] \cdot \vec{A}^r \\
 &= \begin{bmatrix} H_{0,3^n-1} & H_{0,3^n-2} & \cdots & H_{0,0} \\ H_{1,3^n-1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ H_{3^n-1,3^n-1} & \cdots & \cdots & H_{3^n-1,0} \end{bmatrix} \cdot \begin{bmatrix} a_{3^n-1} \\ a_{3^n-2} \\ \vdots \\ a_0 \end{bmatrix} \\
 &= \begin{bmatrix} \sum_{i=0}^{3^n-1} (H_{0,i} \cdot a_i) \\ \sum_{i=0}^{3^n-1} (H_{1,i} \cdot a_i) \\ \vdots \\ \sum_{i=0}^{3^n-1} (H_{3^n-1,i} \cdot a_i) \end{bmatrix} = LPy \cdot \vec{A}.
 \end{aligned}$$

Similar approach can be used to prove (4.51). $\square$

4.4.2 Computational complexity

Based on similar matrix decomposition methods, the forward and inverse butterfly diagrams of all ternary arithmetic helix transforms are also given in Tables 4.12 – 4.15 for the cases $n = 2$ and $x = 1$. The solid and broken lines denote the values 1 and -1 , respectively.

It can be observed that every butterfly diagrams in Classes LN and RN has a permutation part in the left side of the diagram. In fact this part can be located once in any position of the butterfly diagrams, such as in the middle, or in the right side, without affecting the final outputs of the butterfly diagrams. It also should be noticed that the inputs and outputs of the butterfly diagram are exchangeable without changing the matrices for the first helix transform in

each class (LPa , RPa , LNa and RNa).

The first helix transform in each class has the same low computational cost S_n . Based on the matrix decomposition method, the number of additions is

$$S_{(a)n} = 2n \cdot 3^{n-1}. \quad (4.52)$$

The second and third helix transforms in each class have the same reduced computational costs based on the matrix decomposition method both for the forward and inverse transform matrices, as shown in the following equations.

$$S_{(b)n} = S_{(c)n} = 2^n + 6 \cdot (3^{n-1} - 2^{n-1}). \quad (4.53)$$

The last two helix transforms in each class also have the same reduced computational costs as shown below.

$$S_{(d)n} = S_{(e)n} = 3^n - 1. \quad (4.54)$$

4.4.3 Experimental results

The first helix transforms (LPa , RPa , LNa and RNa) in each class have been implemented using Microsoft Visual C++ and run on Pentium IV 2.8 GHz computer with 512 MB RAM. They are run on some benchmark functions that have been modified to represent ternary functions instead of original MCNC, IWLS'93 and ISCAS'95 binary benchmark functions.

Table 4.16 gives the numbers of non-zero spectral coefficients obtained from the four ternary arithmetic helix transforms. The experimental results of best

polarity as well as zero polarity ternary arithmetic (TAR) transform are also given in the last two columns of Table 4.16. It is obvious that the experimental results of LPa and LNa are exactly the same for every ternary benchmark function (TBF), as well as for RPa and RNa . Right helix transforms (RPa and RNa) always obtain smaller number of non-zero spectral coefficients than left helix transforms (LPa and LNa). The reason is that for right helix transforms, an additional permutation operation has been added for each of them before calculating their spectra. Comparing to the results of TAR transform, right helix transforms (RPa and RNa) can obtain better results than both zero polarity and the best polarity TAR transform for some cases.

Table 4.16: Number of non-zero spectral coefficients for ternary arithmetic helix transforms.

TBF	LPH	RPH	LNH	RNH	TAR^{-1}	$TAR_{<0>}^{-1}$
					Best polarity	Zero polarity
9sym	75	16	75	16	16	24
alu4	2183	1141	2183	1141	1362	1978
apex4	242	160	242	160	142	241
clip	234	107	234	107	109	235
con1	60	17	60	17	17	45
ex5	81	46	81	46	55	67
ex1010	237	241	237	241	236	242
inc	81	54	81	54	44	81
misex1	70	37	70	37	31	53
rd84	57	16	57	16	5	5
squar5	24	18	24	18	10	24
xor5	18	7	18	7	5	8
z5xp1	80	39	80	39	27	68

It should be noticed that in order to obtain the best polarity result for TAR transform all 3^n spectra have to be calculated. Hence, it always takes a lot of time to search through all the polarities of TAR transform. On the other hand, the four helix transforms do not have polarity so the calculation needs to

be performed only once for each of them and due to that fact the calculation processes for the four helix transforms are always much faster than that for TAR transform. Table 4.17 shows the calculation time for the four helix transforms and TAR transform. It is obvious that the calculation times to search for the best polarity results for TAR transform are much longer than the corresponding times for calculating the four helix transforms.

Table 4.17: Calculation times for ternary arithmetic helix transforms.

TBF	LPH	RPH	LNH	RNH	TAR^{-1}	$TAR^{-1}_{<0>}$
					Best polarity	Zero polarity
9sym	0.01	< 0.01	< 0.01	< 0.01	4.63	0.07
alu4	0.39	0.41	0.44	0.42	3500.32	4.18
apex4	0.02	0.02	0.01	0.02	21.77	0.15
clip	0.01	< 0.01	< 0.01	0.02	8.32	0.03
con1	< 0.01	< 0.01	< 0.01	< 0.01	1.50	0.02
ex5	0.01	0.02	0.03	0.02	18.25	0.40
ex1010	0.01	0.01	0.02	0.01	12.50	0.12
inc	< 0.01	0.02	0.02	< 0.01	3.88	0.03
misex1	< 0.01	< 0.01	0.01	< 0.01	3.13	0.03
rd84	< 0.01	< 0.01	< 0.01	< 0.01	2.28	0.03
squar5	< 0.01	< 0.01	< 0.01	< 0.01	1.53	0.03
xor5	< 0.01	< 0.01	< 0.01	< 0.01	0.46	0.02
z5xp1	< 0.01	< 0.01	< 0.01	< 0.01	3.62	0.04

4.5 Fastest linearly independent ternary arithmetic transforms

Two categories of fastest Linearly Independent Ternary Arithmetic (LITA) transforms, which possesses forward and inverse butterfly diagrams with lowest computational complexity have been identified and their various properties have been presented in this section. All the fastest LITA transforms are recursively

defined and have consistent formulas relating forward and inverse transform matrices. Computational costs of the calculation for the fastest LITA transforms are also discussed. Some experimental results for standard ternary benchmark functions and comparison with multi-polarity ternary arithmetic transform are also presented.

4.5.1 Properties of fastest LITA transform matrices

In this section, some properties that are useful in calculations and applications of the fastest LITA transform matrices will be discussed. The matrix decomposition method discussed in previous sections will also be used to derive the forward and inverse butterfly diagrams of all the fastest LITA transforms.

Two fastest LITA transforms $T_n^{(1)}$ and $T_n^{(2)}$ are recursively defined as

$$T_n^{(1)} = \begin{bmatrix} T_{n-1}^{(1)} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1}^{(1)} & O_{n-1} & T_{n-1}^{(1)} \end{bmatrix}, \quad (4.55)$$

and

$$T_n^{(2)} = \begin{bmatrix} T_{n-1}^{(2)} & O_{n-1} & Y_{n-1}^{(2)} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & T_{n-1}^{(2)} \end{bmatrix}, \quad (4.56)$$

where the submatrices T_{n-1} are square matrices with the dimension $3^{n-1} \times 3^{n-1}$, and O_{n-1} is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements equal to 0, Y_{n-1} is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements equal to 0 except one that is equal to 1 at the bottom left corner with the superscript 1 or top right corner with the superscript 2. Their inverse transforms matrices are obtained by simply

multiply -1 over their submatrices Y_{n-1} .

The following general matrix form is shown to present the properties of all fastest LITA transforms discussed in the following sections.

$$M_n = \begin{bmatrix} M_{0,0} & M_{0,1} & \cdots & M_{0,3^n-1} \\ M_{1,0} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ M_{3^n-1,0} & \cdots & \cdots & M_{3^n-1,3^n-1} \end{bmatrix}. \quad (4.57)$$

The following properties of $T_n^{(1)}$ and $T_n^{(2)}$ are applied to general matrix form given in (4.57).

Property 4.5.1. The total numbers of nonzero elements in T_n and its inverse matrix T_n^{-1} are equal to $3^n + 2^n - 1$. There are 3^n '1's and $2^n - 1$ '-1's in T_n^{-1} while T_n has all '1's.

Property 4.5.2. The row i of $T_n^{(1)}$ has $(1+d)$ '1' elements, where d follows the rules:

d is the number of '2' elements behind the last 0 of the ternary representation of i if it does not contain '1', otherwise d is equal to zero.

The row i of $(T_n^{(1)})^{-1}$ has one '1' element and d '-1' elements.

If D is defined as $\{1, 2, \dots, d\}$, then the '-1's inside the row i are located at the columns: $i - (3^1 - 1)$, $i - (3^2 - 1)$, $\dots$, $i - (3^d - 1)$.

Property 4.5.3. The column j of $T_n^{(1)}$ has $(1+h)$ '1' elements, where h follows the rules:

h is the number of consecutive '0' elements from the right to the left of the ternary representation of j if it does not contain '1' and the last element of the representation is '0', otherwise h is equal to zero.

The column j of $(T_n^{(1)})^{-1}$ has one '1' element and h '-1' elements.

If H is defined as $\{1, 2, \dots, h\}$, then the '-1's inside the column j are located at the rows: $j + (3^1 - 1), j + (3^2 - 1), \dots, j + (3^h - 1)$.

Property 4.5.4. $T_n^{(1)} = [T_n^{(2)}]^T$, where the superscript T indicates matrix transpose operation.

Due to Property 4.5.4, the position of non-zero elements in rows and columns of the matrix $T_n^{(2)}$ can be described in a similar manner as Properties 4.5.2 and 4.5.3.

Property 4.5.5. The sum W_T of spectral coefficients for both $T_n^{(1)}$ and $T_n^{(2)}$ LITA transforms can be calculated by using $\vec{F}_1$, a subset of $\vec{F}$. For an n -variable ternary function f , $\vec{F}_1$ has 2^{n-1} elements.

Proof.

$$W_T = \sum_{i=0}^{3^n-1} c_i = \sum_{i=0}^{3^n-1} \sum_{j=0}^{3^n-1} T_{i,j}^{-1} f_j = \sum_{j=0}^{3^n-1} \sum_{i=0}^{3^n-1} T_{i,j}^{-1} f_j = \sum_{j=0}^{3^n-1} f_j \left(\sum_{i=0}^{3^n-1} T_{i,j}^{-1} \right).$$

It is clear that only the rows that contain '-1' element need to be calculated. From Property 4.5.2, for $T_n^{(1)}$, rows that contain '-1' must follow the rules: its ternary representation of i do not contain '1' and the last element of its representation must not be '0'.

The ternary representation of i has n elements for n -variable. The last one must be '2' and the other $n - 1$ elements can be any combination of '0' and '2'. There are all together 2^{n-1} combinations. Hence, there are 2^{n-1} rows that contain '-1' elements to be calculated. $\square$

Similar approach can be used to prove Property 4.5.5 for $T_n^{(2)}$.

Figure 4.7: Forward butterfly diagram of $T_3^{(1)}$.

Property 4.5.6. The sum $W_T^{(k)}$ that is the sum of the upper 3^{n-k} spectral coefficients of both transforms $T_n^{(1)}$ and $T_n^{(2)}$ can be obtained by using $\vec{F}_2$, a subset of $\vec{F}$. For an n -variable ternary function f , $\vec{F}_2$ has 2^{n-k-1} elements.

Proof. Similar approach used to prove Property 4.5.5 can be applied to obtain Property 4.5.6 for the two fastest LITA transforms. $\square$

The computational complexity of a transform depends on the way of the construction of the higher matrices from the basic submatrices. With the introduction of submatrix Y_{n-1} in the recursive construction, the introduced matrices are the fastest and most efficient LITA transform matrices.

Figs. 4.7 and 4.8 show the forward and inverse butterfly diagram based on the decomposition matrices discussed in Example 4.3.1. The solid and broken lines in the figures represent addition and subtraction, respectively.

There are also two other fastest LITA transforms that have the same computational costs as $T_n^{(1)}$ and $T_n^{(2)}$. They have different structure in their recursive matrix definitions and butterfly diagrams, and they are denoted by $T_n^{(3)}$ and

Figure 4.8: Inverse butterfly diagram of $(T_3^{(1)})^{-1}$.

$T_n^{(4)}$. The following two equations give the recursive matrix definitions for $T_n^{(3)}$ and $T_n^{(4)}$, respectively.

$$T_n^{(3)} = \begin{bmatrix} Y_{n-1}^{(3)} & O_{n-1} & T_{n-1}^{(3)} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ T_{n-1}^{(3)} & O_{n-1} & O_{n-1} \end{bmatrix}, \quad (4.58)$$

and

$$T_n^{(4)} = \begin{bmatrix} O_{n-1} & O_{n-1} & T_{n-1}^{(4)} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ T_{n-1}^{(4)} & O_{n-1} & Y_{n-1}^{(4)} \end{bmatrix}, \quad (4.59)$$

where the submatrices T_{n-1} are square matrices with the dimension $3^{n-1} \times 3^{n-1}$, Y_{n-1} is a $3^{n-1} \times 3^{n-1}$ submatrix with all its elements equal to 0 except one that is equal to 1 at the top left corner with the superscript 3 or bottom right corner with the superscript 4.

Property 4.5.7. From Definitions 3.1.7 and 3.1.8, and the recursive matrix

definitions of the four discussed fastest LITA transforms,

$$T_n^{(3)} = \mu_{EV} [T_n^{(1)}] \quad (4.60)$$

and

$$T_n^{(4)} = \mu_{EH} [T_n^{(2)}]. \quad (4.61)$$

Property 4.5.8. From Definitions 3.1.7 and 3.1.8, and the recursive matrix definitions of the four discussed fastest LITA transforms,

$$(T_n^{(3)})^{-1} = \mu_{EH} [(T_n^{(1)})^{-1}] \quad (4.62)$$

and

$$(T_n^{(4)})^{-1} = \mu_{EV} [(T_n^{(2)})^{-1}]. \quad (4.63)$$

Similar properties as Properties 4.5.1 – 4.5.6 of $T_n^{(3)}$ and $T_n^{(4)}$ can be derived by applying Properties 4.5.7 and 4.5.8 to these fastest LITA transforms.

By rotating submatrices in four presented fastest LITA transforms, other four fastest LITA transforms can be derived, which have permutation parts in the left side of their forward and inverse butterfly diagrams. They also have the same computational costs as other transforms and they are defined by recursive equations.

The recursive matrix definitions of the four new LITA transforms are given below.

$$T_{*n}^{(1)} = R \left(\begin{bmatrix} T_{n-1}^{(1)} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1}^{(1)} & O_{n-1} & T_{n-1}^{(1)} \end{bmatrix} \right), \quad (4.64)$$

and

$$T'_{*n}{}^{(2)} = R \left(\begin{bmatrix} T_{n-1}^{(2)} & O_{n-1} & Y_{n-1}^{(2)} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & T_{n-1}^{(2)} \end{bmatrix} \right); \quad (4.65)$$

$$T_{*n}^{(3)} = R^{-1} \left(\begin{bmatrix} Y_{n-1}^{(3)} & O_{n-1} & T_{n-1}^{(3)} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ T_{n-1}^{(3)} & O_{n-1} & O_{n-1} \end{bmatrix} \right), \quad (4.66)$$

and

$$T_{*n}^{(4)} = R^{-1} \left(\begin{bmatrix} O_{n-1} & O_{n-1} & T_{n-1}^{(4)} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ T_{n-1}^{(4)} & O_{n-1} & Y_{n-1}^{(4)} \end{bmatrix} \right). \quad (4.67)$$

Fig. 4.9 shows the forward butterfly diagram of the fastest LITA transform $T_{*3}^{(1)}$.

The inverse transform matrices of fastest LITA transforms $T^{*(i)}$, $i = (1, 2, 3, 4)$ are shown below, while Fig. 4.10 gives the butterfly diagram of the inverse transform $(T_{*3}^{(1)})^{-1}$.

$$(T_{*n}^{(1)})^{-1} = R^{-1} \left(\begin{bmatrix} (T_{*n-1}^{(1)})^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ -Y_{n-1}^{(1)} & O_{n-1} & (T_{*n-1}^{(1)})^{-1} \end{bmatrix} \right), \quad (4.68)$$

Figure 4.9: Forward butterfly diagram of $T_{*3}^{(1)}$.

and

$$(T'_{*n}{}^{(2)})^{-1} = R^{-1} \left(\begin{bmatrix} (T_{*n-1}^{(2)})^{-1} & O_{n-1} & -Y_{n-1}^{(2)} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (T_{*n-1}^{(2)})^{-1} \end{bmatrix} \right); \quad (4.69)$$

$$(T_{*n}^{(3)})^{-1} = R \left(\begin{bmatrix} O_{n-1} & O_{n-1} & (T_{*n-1}^{(3)})^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ (T_{*n-1}^{(3)})^{-1} & O_{n-1} & -Y_{n-1}^{(3)} \end{bmatrix} \right), \quad (4.70)$$

and

$$(T_{*n}^{(4)})^{-1} = R \left(\begin{bmatrix} -Y_{n-1}^{(4)} & O_{n-1} & (T_{*n-1}^{(4)})^{-1} \\ O_{n-1} & J_{n-1} & O_{n-1} \\ (T_{*n-1}^{(4)})^{-1} & O_{n-1} & O_{n-1} \end{bmatrix} \right). \quad (4.71)$$

There are also eight other LITA transforms derived from the transforms $T_n^{(1)}$ to $T_n^{(4)}$, and $T_{*n}^{(1)}$ to $T_{*n}^{(4)}$, by replacing the central submatrix of the recursive matrix definition by the matrix itself in the stage $n-1$. They are denoted as $S_n^{(1)}$

Figure 4.10: Inverse butterfly diagram of $(T_{*3}^{(1)})^{-1}$.

to $S_n^{(4)}$, and $S_{*n}^{(1)}$ to $S_{*n}^{(4)}$. The four LITA transforms $S_n^{(1)}$ to $S_n^{(4)}$ are

$$S_n^{(1)} = \begin{bmatrix} S_{n-1}^{(1)} & O_{n-1} & O_{n-1} \\ O_{n-1} & S_{n-1}^{(1)} & O_{n-1} \\ Y_{n-1}^{(1)} & O_{n-1} & S_{n-1}^{(1)} \end{bmatrix}, \quad (4.72)$$

and

$$S_n^{(2)} = \begin{bmatrix} S_{n-1}^{(2)} & O_{n-1} & Y_{n-1}^{(2)} \\ O_{n-1} & S_{n-1}^{(2)} & O_{n-1} \\ O_{n-1} & O_{n-1} & S_{n-1}^{(2)} \end{bmatrix}; \quad (4.73)$$

$$S_n^{(3)} = \begin{bmatrix} Y_{n-1}^{(3)} & O_{n-1} & S_n^{(3)} \\ O_{n-1} & S_n^{(3)} & O_{n-1} \\ S_n^{(3)} & O_{n-1} & O_{n-1} \end{bmatrix}, \quad (4.74)$$

and

$$S_n^{(4)} = \begin{bmatrix} O_{n-1} & O_{n-1} & S_{n-1}^{(4)} \\ O_{n-1} & S_{n-1}^{(4)} & O_{n-1} \\ S_{n-1}^{(4)} & O_{n-1} & Y_{n-1}^{(4)} \end{bmatrix}. \quad (4.75)$$

The following properties are based on the general matrix in (4.57). Some properties of LITA transforms $S_n^{(1)}$ and $S_n^{(2)}$ will be presented, while similar approach can be easily applied to the other two LITA transforms $S_n^{(3)}$ and $S_n^{(4)}$.

Property 4.5.9. The total numbers of non-zero elements in S_n and its inverse matrix S_n^{-1} are equal to $(3^{n+1} - 1)/2$. All the elements in S_n are '1's and there are $(3^n - 1)/2$ '-1's and 3^n '1's in S_n^{-1} .

Property 4.5.10. The row i of $S_n^{(1)}$ has $(1 + d')$ '1' elements, where d' follows the rules:

d' is the number of '0' elements behind the last non-zero element of the ternary representation of $(i + 1)$.

The row i of $(S_n^{(1)})^{-1}$ has one '1' element and d' '-1' elements.

If D' is defined as $\{1, 2, \dots, d'\}$, then the '-1's inside the row i are located at the columns: $i - (3^1 - 1)$, $i - (3^2 - 1)$, $\dots$, $i - (3^{d'} - 1)$.

Property 4.5.11. The column j of $S_n^{(1)}$ has $(1 + h')$ '1' elements, where h' follows the rules:

h' is the number of consecutive '0' elements from the right to the left of the ternary representation of j if the last element of the representation is '0', otherwise h' is equal to zero.

The column j of $(S_n^{(1)})^{-1}$ has one '1' element and h' '-1' elements.

If H' is defined as $\{1, 2, \dots, h'\}$, then the '-1's inside the column j are located at the rows: $j + (3^1 - 1)$, $j + (3^2 - 1)$, $\dots$, $j + (3^{h'} - 1)$.

Property 4.5.12. $S_n^{(1)} = [S_n^{(2)}]^T$, where the superscript T indicates matrix transpose operation.

Due to Property 4.5.12, the position of non-zero elements in rows and columns

of the matrix $S_n^{(2)}$ can be described in a similar manner as in Properties 4.5.10 and 4.5.11.

Property 4.5.13. The sum W_S of spectral coefficients for both fastest $S_n^{(1)}$ and $S_n^{(2)}$ LITA transforms can be calculated by using $\vec{F}_1'$, a subset of $\vec{F}$. For an n -variable ternary function f , $\vec{F}_1'$ has 3^{n-1} elements.

Proof.

$$W_S = \sum_{i=0}^{3^n-1} c_i = \sum_{i=0}^{3^n-1} \sum_{j=0}^{3^n-1} S_{i,j}^{-1} f_j = \sum_{j=0}^{3^n-1} \sum_{i=0}^{3^n-1} S_{i,j}^{-1} f_j = \sum_{j=0}^{3^n-1} f_j \left(\sum_{i=0}^{3^n-1} S_{i,j}^{-1} \right).$$

It is clear that only the rows that contain '−1' element need to be calculated. From Property 4.5.10, for $S_n^{(1)}$, the rows that contain '−1' must follow the rules: the last one element of its ternary representation of $i + 1$ must be equal to '0'.

For n variables, the ternary representation of $i + 1$ has n elements. The last one must be '0' and the other $n - 1$ elements can be any combination of '0', '1' and '2'. There are all together 3^{n-1} combinations. Hence, there are 3^{n-1} rows that contain '−1' element to be calculated. $\square$

Similar approach can be used to prove Property 4.5.13 for $S_n^{(2)}$.

Property 4.5.14. The sum $W_S^{(k)}$ that is the sum of the upper 3^{n-k} spectral coefficients of both fastest $S_n^{(1)}$ and $S_n^{(2)}$ LITA transforms can be obtained by using $\vec{F}_2'$, a subset of $\vec{F}$. For an n -variable ternary function f , $\vec{F}_2'$ has 3^{n-k-1} elements.

Proof. Similar to the proof for Property 4.5.13. $\square$

The recursive matrix definitions of LITA transforms $S_{*n}^{(1)}$ to $S_{*n}^{(4)}$ are given in the following equations.

$$S_{*n}^{(1)} = R \left(\begin{bmatrix} S_{n-1}^{(1)} & O_{n-1} & O_{n-1} \\ O_{n-1} & S_{n-1}^{(1)} & O_{n-1} \\ Y_{n-1}^{(1)} & O_{n-1} & S_{n-1}^{(1)} \end{bmatrix} \right), \quad (4.76)$$

and

$$S'_{*n}{}^{(2)} = R \left(\begin{bmatrix} S_{n-1}^{(2)} & O_{n-1} & Y_{n-1}^{(2)} \\ O_{n-1} & S_{n-1}^{(2)} & O_{n-1} \\ O_{n-1} & O_{n-1} & S_{n-1}^{(2)} \end{bmatrix} \right); \quad (4.77)$$

$$S_{*n}^{(3)} = R^{-1} \left(\begin{bmatrix} Y_{n-1}^{(3)} & O_{n-1} & S_{n-1}^{(3)} \\ O_{n-1} & S_{n-1}^{(3)} & O_{n-1} \\ S_{n-1}^{(3)} & O_{n-1} & O_{n-1} \end{bmatrix} \right), \quad (4.78)$$

and

$$S_{*n}^{(4)} = R^{-1} \left(\begin{bmatrix} O_{n-1} & O_{n-1} & S_{n-1}^{(4)} \\ O_{n-1} & S_{n-1}^{(4)} & O_{n-1} \\ S_{n-1}^{(4)} & O_{n-1} & Y_{n-1}^{(4)} \end{bmatrix} \right). \quad (4.79)$$

Similar properties as Properties 4.5.9 – 4.5.14 can be easily derived for LITA transforms $S_{*n}^{(1)}$ to $S_{*n}^{(4)}$.

4.5.2 Computational costs

Due to the fact, that the presented fastest LITA transforms are recursively defined and they contained very few non-zero elements in their matrices, the procedure to obtain the spectra of a switching function could be simplified using the matrix decomposition method discussed in previous section. From

Figs. 4.7 – 4.10, it is obvious that only a few numbers of the inputs require additional operations while most of them could be directly transferred to the outputs without any computational costs.

Based on the matrix decomposition method, the number of arithmetic additions/subtractions required to compute these fastest LITA transforms $T_n^{(1)}$ to $T_n^{(4)}$ and $T_{*n}^{(1)}$ to $T_{*n}^{(4)}$, denoted by $C_{n(T)}$, for any n -variable ternary logic function is given by

$$C_{n(T)} = 2^n - 1. \quad (4.80)$$

The number of arithmetic additions/subtractions required to compute the transforms $S_n^{(1)}$ to $S_n^{(4)}$ and $S_{*n}^{(1)}$ to $S_{*n}^{(4)}$, denoted by $C_{n(S)}$, for any n -variable ternary logic function is given by

$$C_{n(S)} = (3^n - 1)/2. \quad (4.81)$$

Table 4.18 gives the comparison of the computational costs between these two categories of fastest LITA transforms, T_n and S_n , with the forward and inverse ternary arithmetic transforms calculated through the direct matrix computation method as well as using the fast transform algorithms.

4.5.3 Ternary logic function verification

There are two ternary functions that will be considered for verification problem in this section. They are

$$f_1(\vec{x}_3) = x_1^0 \cdot (2x_3^0x_2^2 + x_3^2x_2^0) + x_3^1x_2^1 \cdot (x_1^0 + x_1^1) + 2x_3^2x_2^2$$

Table 4.18: Computational costs of fastest LITA transforms.

n	T	S	TAR		TAR^{-1}	
			direct	fast	direct	fast
			$7^n - 3^n$	$4n \cdot 3^{n-1}$	$6^n - 3^n$	$3n \cdot 3^{n-1}$
1	1	1	4	4	3	3
2	3	4	40	24	27	18
3	7	13	316	108	189	81
4	15	40	2320	432	1215	324
5	31	121	16564	1620	7533	1215
6	63	364	116920	5832	45927	4374
7	127	1093	821356	20412	277749	15309

and

$$f_2(\vec{x}_3) = x_2^2 \cdot (x_1^0 + x_1^2) \cdot (x_3^0 + x_3^2) + x_3^1 x_2^1 x_1^1 + x_3^2 x_1^2 \cdot (x_2^2 + x_2^0) + x_1^0 \cdot (x_3^0 x_2^2 + x_3^2 x_2^0),$$

the symbol x_j^i represents ternary variable and the superscript i refers to the corresponding ternary value.

The truth vector of any ternary function is represented by

$$\vec{F} = \left[t_0^k \ t_1^k \ \dots \ t_{26}^k \right]^T,$$

k denotes the value of the element in the truth vector and $k \in \left(0, 1, 2 \right)$.

In the following equations, zero elements are not shown. The truth vectors of the two ternary functions are

$$\vec{F}_1 = \left[t_6^2 \ t_{12}^1 \ t_{13}^1 \ t_{18}^1 \ t_{24}^2 \ t_{25}^2 \ t_{26}^2 \right]^T$$

and

$$\vec{F}_2 = \left[t_6^2 \ t_7^1 \ t_{13}^1 \ t_{18}^1 \ t_{20}^1 \ t_{24}^1 \ t_{26}^2 \right]^T.$$

For $x_1 = x_2 = x_3 = 0$, both ternary functions evaluate to 0, for $x_1 = x_2 = x_3 = 1$, they evaluate to 1, and for $x_1 = x_2 = x_3 = 2$, they evaluate to 2. Such results will require to generate big number of random function testing. In this section, the fastest LITA transform $T^{(1)}$ is used to discover the difference in verification of these two ternary functions.

The following notation will be used, $\vec{A} = \begin{bmatrix} a_0^k & a_1^k & \dots & a_{26}^k \end{bmatrix}^T$, k denotes the value of the element in the spectrum column vector and $k \in \begin{pmatrix} -2, & -1, & 0 & 1 & 2 \end{pmatrix}$. According to Properties 4.5.2 and 4.5.3 discussed in Section 4.5.1, and the inverse butterfly diagram of $(T_3^{(1)})^{-1}$ shown in Fig. 4.8, the spectra of the two ternary functions are easily obtained.

$$\vec{A}_1 = \begin{bmatrix} a_6^2 & a_8^{-2} & a_{12}^1 & a_{13}^1 & a_{18}^1 & a_{20}^{-1} & a_{24}^2 & a_{25}^2 & a_{26}^{-1} \end{bmatrix},$$

and

$$\vec{A}_2 = \begin{bmatrix} a_6^2 & a_8^{-1} & a_{13}^1 & a_{18}^1 & a_{24}^1 \end{bmatrix}.$$

It is clear that the two ternary functions have different numbers of spectra coefficients. Hence, it can be concluded that the two ternary functions are not equivalent and no further calculation is necessary.

4.5.4 Experimental results

The fastest LITA transforms from two categories, T_n and S_n , as well as multipolarity ternary arithmetic (TAR) transform have been implemented using Microsoft Visual C++ and run on PIII 500 MHz computer with 128 MB RAM.

Table 4.19 presents the number of non-zero spectral coefficients for six fastest

Table 4.19: Number of non-zero spectral coefficients for T_n .

TBF		$T_n^{(1)}$	$T_n^{(2)}$	$T_{*n}^{(1)}$	$T_{*n}^{(2)}$	$T_{*n}^{(3)}$	$T_{*n}^{(4)}$	TAR^{-1}	$TAR_{<0>}^{-1}$
								Best polarity	Zero polarity
alu4	O ₁	1393	1390	1392	1389	1390	1392	442	1112
	O ₂	2150	2150	2150	2152	2151	2150	382	1045
	O ₃	1557	1550	1557	1550	1550	1557	592	1322
	O ₄	1130	1131	1130	1131	1131	1130	702	1287
clip	O ₁	106	105	106	105	105	106	62	200
	O ₂	127	121	127	121	121	127	75	157
	O ₃	64	66	64	66	66	64	41	130
apex4	O ₁	13	14	13	14	14	13	51	112
	O ₂	70	74	69	74	73	69	78	128
	O ₃	57	54	57	54	53	57	77	137
	O ₄	57	52	57	52	52	57	79	116
	O ₅	68	62	67	62	62	67	79	138
	O ₆	103	106	102	105	105	102	76	146
	O ₇	49	47	49	46	46	49	85	159
	O ₈	59	60	59	59	60	59	82	160
	O ₉	14	11	15	11	11	15	52	118
	O ₁₀	46	49	46	48	48	46	51	118
ex1010	O ₁	64	71	70	63	62	70	133	163
	O ₂	70	66	66	69	70	66	134	160
	O ₃	55	54	54	54	54	54	136	156
	O ₄	60	61	61	60	61	61	127	152
	O ₅	59	59	60	58	58	60	137	157
inc	O ₁	32	33	32	32	33	32	17	47
	O ₂	38	38	37	38	38	37	30	51
	O ₃	31	32	32	31	31	32	22	54
	O ₄	17	16	15	17	17	15	23	54
	O ₅	18	18	18	18	18	18	2	6

LITA transform matrices, $T_n^{(1)}$, $T_n^{(2)}$ and $T_{*n}^{(1)}$ to $T_{*n}^{(4)}$. Due to the fact that the transform matrices $T_n^{(3)}$ and $T_n^{(4)}$ are vertical and horizontal permutations of $T_n^{(1)}$ and $T_n^{(2)}$, respectively, the number of non-zero spectral coefficients for $T_n^{(3)}$ and $T_n^{(4)}$ are the same as $T_n^{(1)}$ and $T_n^{(2)}$. Hence the results for $T_n^{(3)}$ and $T_n^{(4)}$ are not shown in Table 4.19. The experimental results of other six LITA transform matrices from category S_n are given in Table 4.20. For the same reason, the results of $S_n^{(3)}$ and $S_n^{(4)}$ are not shown. The experimental results for the best polarity as well as zero polarity TAR transform are also given in the last two columns of Tables 4.19 and 4.20.

It can be seen that the results of T_n and S_n are better than the results of zero-polarity TAR transform for majority cases by getting smaller number of non-zero spectral coefficients. For almost all outputs of ternary benchmark functions apex4 and ex1010, the ternary fastest LITA transforms even can obtain better results when compared with the experimental results of the best polarity TAR transform.

Table 4.21 shows the calculation time for some T_n and S_n transforms, and the calculation time for searching through all the polarities and finding zero polarity for multi-polarity TAR transform. It should be noticed that in order to obtain the best polarity result for TAR transform all 3^n spectra have to be calculated what takes a lot of time. On the other hand, T_n and S_n transforms do not have polarity so the calculation needs to be performed only once for each of them what is the same as the calculation of zero polarity TAR transform. Hence the calculation time for T_n and S_n transforms as well as zero polarity TAR transform are always much faster than that for multi-polarity TAR transform. In addition, the calculation times of T_n and S_n are also shorter than the calculation time of

Table 4.20: Number of non-zero spectral coefficients for S_n .

TBF		$S_n^{(1)}$	$S_n^{(2)}$	$S_{*n}^{(1)}$	$S_{*n}^{(2)}$	$S_{*n}^{(3)}$	$S_{*n}^{(4)}$	TAR^{-1}	$TAR_{<0>}^{-1}$
								Best polarity	Zero polarity
alu4	O ₁	1214	1214	1213	1214	1213	1214	442	1112
	O ₂	1794	1816	1816	1794	1816	1794	382	1045
	O ₃	1510	1459	1459	1510	1459	1510	592	1322
	O ₄	1145	1111	1111	1145	1111	1145	702	1287
clip	O ₁	72	78	78	72	78	72	62	200
	O ₂	112	125	125	112	125	112	75	157
	O ₃	53	47	47	53	47	53	41	130
apex4	O ₁	19	20	20	19	20	19	51	112
	O ₂	81	69	68	80	68	80	78	128
	O ₃	60	66	66	59	66	59	77	137
	O ₄	57	66	66	57	66	57	79	116
	O ₅	65	73	72	65	72	65	79	138
	O ₆	95	89	88	94	88	94	76	146
	O ₇	47	45	45	46	45	46	85	159
	O ₈	62	55	55	62	55	62	82	160
	O ₉	11	16	17	11	17	11	52	118
	O ₁₀	41	35	35	40	35	40	51	118
ex1010	O ₁	78	77	76	76	76	76	133	163
	O ₂	80	77	77	80	77	80	134	160
	O ₃	63	65	65	62	65	62	136	156
	O ₄	73	72	72	74	72	74	127	152
	O ₅	71	71	72	71	72	71	137	157
inc	O ₁	24	30	29	25	29	25	17	47
	O ₂	32	36	35	32	35	32	30	51
	O ₃	27	25	25	27	25	27	22	54
	O ₄	20	16	15	20	15	20	23	54
	O ₅	12	12	12	12	12	12	2	6

Table 4.21: Calculation times in seconds for T_n and S_n .

TBF	$T_n^{(1)}$	$T_n^{(2)}$	$S_n^{(1)}$	$S_n^{(2)}$	TAR^{-1}	$TAR_{<0>}^{-1}$
					Best polarity	Zero polarity
9sym	0.01	0.01	0.02	0.01	4.63	0.07
alu4	1.84	1.73	1.83	1.88	3500.32	4.18
apex4	0.08	0.09	0.08	0.09	21.77	0.15
clip	0.03	0.03	0.03	0.03	8.32	0.03
con1	<0.01	0.01	0.01	<0.01	1.50	0.02
ex5	0.33	0.34	0.34	0.33	18.25	0.40
ex1010	0.03	0.04	0.04	0.05	12.50	0.12
inc	0.02	0.01	0.01	0.01	3.88	0.03
misex1	0.02	0.02	0.01	0.01	3.13	0.03
rd84	0.01	<0.01	<0.01	0.01	2.28	0.03
squar5	0.01	0.01	0.01	0.01	1.53	0.03
xor5	<0.01	<0.01	<0.01	<0.01	0.46	0.02
z5xp1	0.01	0.01	0.01	0.01	3.62	0.04

zero polarity TAR transform due to Properties 4.5.5 and 4.5.13.

Chapter 5

Generation of Linearly Independent Transforms over $GF(4)$

The concept of ternary linearly independent transforms from previous chapters has been extended to Galois Field (4) in this chapter. We are showing F – R type Quaternary Linearly Independent (QLI) transforms, Quaternary Linearly Independent Helix (QLIH) transforms and Fastest Quaternary Linearly Independent (FQLI) transforms, their fast calculating methods, butterfly diagrams, computational costs and experimental results in Sections 5.2 – 5.4, consecutively. In Section 5.5, another special family of QLI transforms are also presented, which shows their tight relations with binary LI transforms so that they can be generated directly from binary LI transforms.

5.1 Basic definition of linearly independent transforms over GF(4)

Definition 5.1.1. Let M_n be a $4^n \times 4^n$ matrix with columns corresponding to some quaternary functions of n variables over GF(4). If the set of columns is linearly independent with respect to digit by digit GF(4) additions, then M_n has one unique inverse M_n^{-1} , and is said to be linearly independent, i.e.

$$M_n \cdot M_n^{-1} = I_n \quad (5.1)$$

where I_n is a $4^n \times 4^n$ identity matrix and all the operations are performed over GF(4).

The linearly independent transform based on Definition 5.1.1 can be described by the following general formulae performed over GF(4):

$$M_n \cdot \vec{A} = \vec{F} \quad (5.2)$$

and

$$M_n^{-1} \cdot \vec{F} = \vec{A} \quad (5.3)$$

where $\vec{F} = \begin{bmatrix} f_0 & f_1 & \dots & f_{4^n-1} \end{bmatrix}^T$ is a column vector defining the truth vector of a quaternary function $f(x_n)$ in natural quaternary ordering, M_n is a matrix of size $4^n \times 4^n$ defined by any linearly independent set of n -variable quaternary functions and $\vec{A} = \begin{bmatrix} a_0 & a_1 & \dots & a_{4^n-1} \end{bmatrix}^T$ is the coefficient column vector for the particular transform matrix M_n with the inverse M_n^{-1} while T is the matrix transpose operator.

Formula (5.2) can be also expressed by

$$f(x_n) = \sum_{i=0}^{4^n-1} a_i g_i \quad (5.4)$$

where g_i represents the truth vector of an n -variable quaternary switching function and the matrix $M_n = \begin{bmatrix} g_0 & g_1 & \dots & g_{4^n-1} \end{bmatrix}^T$. The symbol $\sum$ is the addition performed over $GF(4)$.

Definition 5.1.2. Let M_n be a $4^n \times 4^n$ matrix following Definition 5.1.1. In what follows only matrices M_n will be considered, that additionally can be partitioned into sixteen $4^{n-1} \times 4^{n-1}$ submatrices M_{n-1} as shown in the following equation,

$$M_n = \begin{bmatrix} M_{n-1}^{(1,1)} & \dots & M_{n-1}^{(1,4)} \\ \dots & \dots & \dots \\ M_{n-1}^{(4,1)} & \dots & M_{n-1}^{(4,4)} \end{bmatrix} \quad (5.5)$$

where the superscripts of the submatrices are used to denote the location of the submatrices in M_n .

Definition 5.1.3. The μ_{EH} operator on M_n is defined as grouping the recursive equations in all the submatrices vertically and interchanging them in the submatrices horizontally.

$$\mu_{EH}[M_n] = \begin{bmatrix} M_{n-1}^{(1,4)} & \dots & M_{n-1}^{(1,1)} \\ \dots & \dots & \dots \\ M_{n-1}^{(4,4)} & \dots & M_{n-1}^{(4,1)} \end{bmatrix}. \quad (5.6)$$

Definition 5.1.4. The μ_{EV} operator on M_n is defined as grouping the recursive equations in all the submatrices horizontally and interchanging them in the

submatrices vertically.

$$\mu_{EV} [M_n] = \begin{bmatrix} M_{n-1}^{(4,1)} & \dots & M_{n-1}^{(4,4)} \\ \dots & \dots & \dots \\ M_{n-1}^{(1,1)} & \dots & M_{n-1}^{(1,4)} \end{bmatrix}. \quad (5.7)$$

5.2 Generation of linearly independent transforms over $GF(4)$

The new linearly independent transforms over $GF(4)$ are introduced in this section. This family is recursively defined and has consistent formulas relating forward and inverse transforms. The computational costs of the calculation for the new transforms is also discussed.

5.2.1 Classes of quaternary linearly independent transforms

In this section, new Quaternary Linearly Independent (QLI) transforms will be investigated. They are categorized into several classes according to their properties and relations. All presented QLI transforms are recursively defined and constructed from three basic types of submatrices, M_{n-1} , I_{n-1} and O_{n-1} . All of them are the matrices of size $4^{n-1} \times 4^{n-1}$. The submatrix I_{n-1} is an identity matrix containing all zero elements except only '1's in its diagonal while the submatrix O_{n-1} has all its elements equal to zero.

All QLI transforms presented here are derived from the general matrix shown

below:

$$M_n = \begin{bmatrix} M_{n-1}^{(1,1)} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{(2,1)} & M_{n-1}^{(2,2)} & O_{n-1} & O_{n-1} \\ M_{n-1}^{(3,1)} & O_{n-1} & M_{n-1}^{(3,3)} & O_{n-1} \\ M_{n-1}^{(4,1)} & M_{n-1}^{(4,2)} & M_{n-1}^{(4,3)} & M_{n-1}^{(4,4)} \end{bmatrix}, \quad (5.8)$$

where all the submatrices in the upper-right side of the diagonal as well as the submatrix in the location (3, 2) are zero submatrices.

The general QLI transform matrix is a self-inverse transform matrix over $GF(4)$. When one non-zero submatrix in the general QLI transform matrix is replaced by the identity submatrix I_{n-1} , nine new QLI transforms will be derived and they are categorized as Class A.

If the general QLI matrix is divided into four separated square areas, each including four submatrices, as shown below,

$$M_n = \begin{bmatrix} \left(\begin{matrix} M_{n-1}^{(1,1)} & O_{n-1} \\ M_{n-1}^{(2,1)} & M_{n-1}^{(2,2)} \end{matrix} \right)^1 & (O_{n-1}) \\ \left(\begin{matrix} M_{n-1}^{(3,1)} & O_{n-1} \\ M_{n-1}^{(4,1)} & M_{n-1}^{(4,2)} \end{matrix} \right)^3 & \left(\begin{matrix} M_{n-1}^{(3,3)} & O_{n-1} \\ M_{n-1}^{(4,3)} & M_{n-1}^{(4,4)} \end{matrix} \right)^2 \end{bmatrix}, \quad (5.9)$$

then there are two locations for the only one identity submatrix I_{n-1} in the matrix. One is the lower-left corner of each square area, such as (2, 1), (4, 1) and (4, 3); the other one is the location of all other non-zero submatrices.

By using symbolically calculation of the matrix in (5.9), a general inverse matrix can be obtained and also used to derive all the inverse transform matrices in this section.

Rule 5.2.1.

Let $M_n = \begin{bmatrix} A & O_{n-1} & O_{n-1} & O_{n-1} \\ B & C & O_{n-1} & O_{n-1} \\ D & O_{n-1} & E & O_{n-1} \\ F & G & H & K \end{bmatrix}$ represent the matrix defined in (5.9),

where $A-K$ denote $4^{n-1} \times 4^{n-1}$ submatrices, then the inverse transform matrix M_n^{-1} is given by

$$M_n^{-1} = \begin{bmatrix} A^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ C^{-1}BA^{-1} & C^{-1} & O_{n-1} & O_{n-1} \\ E^{-1}DA^{-1} & O_{n-1} & E^{-1} & O_{n-1} \\ Q & K^{-1}GC^{-1} & K^{-1}HE^{-1} & K^{-1} \end{bmatrix} \quad (5.10)$$

where

$$Q = K^{-1} (FA^{-1} + GC^{-1}BA^{-1} + HE^{-1}DA^{-1}). \quad (5.11)$$

The inverse QLI transform matrices of Class A can be derived directly from the forward QLI transform matrices by Rule 5.2.1.

One of the QLI transform matrices in Class A is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.12)$$

By Rule 5.2.1, the inverse QLI transform matrix is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.13)$$

When the number of the identity submatrix I_{n-1} in the forward QLI transform matrix increase to two, thirty six new QLI transforms can be derived from the general QLI transform in (5.9). They are categorized into Class B. In order to investigate the properties of the QLI transforms in Class B, all thirty six QLI transform matrices are divided into three sub-classes according to the locations of the two identity submatrices I_{n-1} in the forward transform matrices.

Class B.1 contains six QLI transforms, which have both two identity submatrices in the diagonal. One of them is shown below:

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.14)$$

Its two identity submatrices are in location (2, 2) and (3, 3). It can also be divided into four sub-areas as shown in (5.9).

In order to obtain the inverse transform matrix directly from (5.14), Rule

5.2.1 applies to (5.14) and the inverse QLI transform matrix is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & I_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.15)$$

If both identity submatrices are not in the diagonal of the forward QLI transform matrix, ten new QLI transforms will be derived, which are categorized into Class B.2. The following equation gives one example of the QLI transform in this class.

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.16)$$

The inverse QLI transform matrices of Class B.2 can be derived from the forward QLI transform matrices by the Rule 5.2.1 and given by

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.17)$$

There are still twenty QLI transforms that are categorized into Class B.3. Each QLI transform matrices in Class B.3 has two identity submatrices I_{n-1} and only one submatrix I_{n-1} is in the diagonal of the forward QLI transform matrix, which is denoted by I_{n-1}^1 , and the other submatrix I_{n-1} is denoted by

I_{n-1}^2 . One example is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1}^2 & O_{n-1} & I_{n-1}^1 & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.18)$$

Based on Rule 5.2.1, the inverse QLI transform matrix of (5.18) is given by

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.19)$$

Class C of QLI transforms is derived by increasing the number of identity submatrices I_{n-1} to three. 84 new QLI transforms are included in this class. They can be divided into four sub-classes according to the different number of identity submatrices located in the matrix diagonal.

$$M_n = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.20)$$

In (5.20), there is only one identity submatrix located in the diagonal. Its

inverse transform matrix is given by

$$M_n^{-1} = \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ I_{n-1} & (M_{n-1}^{-1})^2 & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}. \quad (5.21)$$

Forty QLI transforms similar to transform given in (5.20) are categorized into Class C.1.

Class C.2 contains thirty QLI transform matrices which have two identity submatrices I_{n-1} located in the diagonal of the forward transform matrices. One of the transforms is

$$M_n = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ M_{n-1} & I_{n-1}^3 & M_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.22)$$

In (5.22), two identity submatrices I_{n-1}^1 and I_{n-1}^2 are located in the diagonal (1, 1) and (3, 3). The third identity submatrix I_{n-1}^3 is in location (4, 2).

The inverse transform matrix of (5.22) are given in (5.23) following Rule 5.2.1 of Class C.2.

$$M_n^{-1} = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ Q & (M_{n-1}^{-1})^2 & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}, \quad (5.23)$$

where $Q = I_{n-1} + M_{n-1} + M_{n-1}^{-1}$.

When all the three identity submatrices I_{n-1} are located in the diagonal, four derivations are categorized into Class C.3. One example is given in (5.24).

$$M_n = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & I_{n-1}^3 \end{bmatrix}. \quad (5.24)$$

Its inverse transform matrix is given by

$$M_n^{-1} = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ M_{n-1}^2 & I_{n-1} & M_{n-1} & I_{n-1}^3 \end{bmatrix}. \quad (5.25)$$

The last ten QLI transforms in Class C have none of their three identity submatrices I_{n-1} in the diagonal of the forward matrices. They are included in Class C.4. One of the forward transform matrices is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.26)$$

In (5.26), none of the identity submatrices I_{n-1} is in the matrix diagonal.

Following Rule 5.2.1, the inverse transform matrix of (5.26) is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ Q & (M_{n-1}^{-1})^2 & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}, \quad (5.27)$$

where $Q = M_{n-1}^{-1} + (M_{n-1}^{-1})^2 + (M_{n-1}^{-1})^3$.

In this section, the concept of quaternary linearly independent logic was investigated. 130 new QLI transforms have been discussed with their properties and relations. They are also categorized into several classes according to their recursive matrix structures and due to the same reason, these QLI transforms have advantaged in calculation speed.

5.2.2 Complete example

Let a 2-variable quaternary switching function be

$$\vec{F} = \left[2 \ 0 \ 1 \ 3 \ 2 \ 0 \ 3 \ 3 \ 2 \ 1 \ 1 \ 3 \ 3 \ 0 \ 2 \ 3 \right]^T.$$

Four QLI transforms from each class presented in Section 5.2.1 will be used here. Their recursive definitions of the inverse transforms are given in equations (5.13), (5.15), (5.17) and (5.19), and they are denoted by symbols $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$.

The spectra of the four QLI transforms are presented below:

$$\begin{aligned}\overrightarrow{A_{(a)}} &= \begin{bmatrix} 2 & 2 & 3 & 0 & 0 & 0 & 2 & 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T; \\ \overrightarrow{A_{(b)}} &= \begin{bmatrix} 2 & 2 & 3 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 2 \end{bmatrix}^T; \\ \overrightarrow{A_{(c)}} &= \begin{bmatrix} 2 & 2 & 3 & 0 & 0 & 0 & 2 & 2 & 0 & 1 & 0 & 0 & 1 & 2 & 2 & 3 \end{bmatrix}^T; \\ \overrightarrow{A_{(d)}} &= \begin{bmatrix} 2 & 2 & 3 & 0 & 0 & 0 & 2 & 2 & 0 & 3 & 2 & 3 & 1 & 2 & 2 & 3 \end{bmatrix}^T.\end{aligned}$$

By (5.4), the quaternary functions of the four QLI transforms are

$$\begin{aligned}f_{(a)} &= 3 \cdot g_2 + 2 \cdot (g_0 + g_1 + g_6 + g_7) + (g_9 + g_{12}); \\ f_{(b)} &= 3 \cdot g_2 + 2 \cdot (g_0 + g_1 + g_6 + g_{15}) + (g_9 + g_{12}); \\ f_{(c)} &= 3 \cdot (g_2 + g_{15}) + 2 \cdot (g_0 + g_1 + g_6 + g_7 + g_{13} + g_{14}) + (g_9 + g_{12}); \\ f_{(d)} &= 3 \cdot (g_2 + g_9 + g_{11} + g_{15}) + 2 \cdot (g_0 + g_1 + g_6 + g_7 + g_{10} + g_{13} + g_{14}) + g_{12}.\end{aligned}$$

where g_i is obtained from the forward matrix of the QLI transforms. All the additions and multiplications are performed over GF(4).

5.2.3 Experimental results

The four QLI transforms, $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$, are implemented using Microsoft Visual C++ and run on Pentium IV 2.8GHz computer with 512MB RAM. The benchmark functions have been modified to represent quaternary functions instead of original MCNC, IWLS'93 and ISCAS'95 binary benchmark functions using the algorithm described in [FL04].

Table 5.1 presents the number of non-zero spectral coefficients obtained from

Table 5.1: Number of non-zero spectral coefficients for QLI transforms.

QBF	$M_{(a)}^{-1}$	$M_{(b)}^{-1}$	$M_{(c)}^{-1}$	$M_{(d)}^{-1}$	QRM^{-1}
9sym	302	154	571	538	758
apex4	649	512	1005	988	1022
clip	495	459	814	736	874
con1	53	56	149	141	148
ex5	236	230	255	256	256
ex1010	998	971	1021	1022	1024
inc	152	134	224	214	211
misex1	205	237	214	214	142
rd84	236	204	206	195	251
squar5	40	35	56	50	57
xor5	27	23	35	35	31
z5xpl	134	140	222	222	230

Quaternary Benchmark Functions (QBFs) for the four QLI transforms. It is obvious that the four QLI transforms can always obtain more zero spectral coefficients than QRM transform for most tested QBFs. The results of $M_{(a)}^{-1}$ and $M_{(b)}^{-1}$ are even better than that of $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$.

The calculation times of the four QLI transforms running on the QBFs have also been given in Table 5.2. It is shown that the calculation speed of QLI transforms is much faster than QRM transform.

5.3 Quaternary linearly independent helix transforms

Four classes of QLIH transforms are presented in this section according to their different matrix structures. They follow Definitions 5.1.1 – 5.1.2. All presented QLIH transforms are recursively defined and constructed by three kinds of basic sub-matrices, M_{n-1} , I_{n-1} and O_{n-1} . The submatrix M_{n-1} is the QLIH transform

Table 5.2: Calculation times in seconds for QLI transforms.

QBF	$M_{(a)}^{-1}$	$M_{(b)}^{-1}$	$M_{(c)}^{-1}$	$M_{(d)}^{-1}$	QRM^{-1}
9sym	0.02	0.03	0.03	0.03	0.05
apex4	0.17	0.17	0.19	0.19	0.34
clip	0.06	0.06	0.06	0.08	0.13
con1	0.01	<0.01	<0.01	<0.01	0.01
ex5	0.08	0.06	0.08	0.06	0.11
ex1010	0.09	0.11	0.09	0.09	0.19
inc	0.01	0.01	<0.01	0.02	0.12
misex1	<0.01	0.02	0.02	0.03	0.02
rd84	0.01	<0.01	<0.01	0.02	0.02
squar5	<0.01	<0.01	<0.01	<0.01	0.01
xor5	<0.01	<0.01	<0.01	<0.01	0.01
z5xp1	0.02	0.02	0.01	0.01	0.02

matrix with order $N - 1$. The submatrix I_{n-1} is an identity matrix, which contains all zero elements except only '1's in the diagonal. The submatrix O_{n-1} has all its elements equal to zero.

5.3.1 Classification of QLIH transforms

There are two QLIH transform matrices that are constructed only from the submatrices M_{n-1} and O_{n-1} given by the following equations:

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \quad (5.28)$$

and

$$M_n = \begin{bmatrix} M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}. \quad (5.29)$$

It can be seen that the QLIH transform matrix in (5.29) is the transpose of the matrix in (5.28), which results in similar properties. Hence in this section only the matrix in (5.28) will be analyzed.

The QLIH transform matrix in (5.28) is a self-inverse matrix as presented in (5.30).

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.30)$$

By using symbolically calculation of the matrix in (5.28), a general inverse matrix can be obtained and also used as the proof for all the following rules in this section.

Property 5.3.1. Let $M_n = \begin{bmatrix} A & O_{n-1} & O_{n-1} & O_{n-1} \\ B & C & D & O_{n-1} \\ O_{n-1} & O_{n-1} & E & O_{n-1} \\ O_{n-1} & O_{n-1} & F & G \end{bmatrix}$ represent the matrix defined in (5.28), where A - G denote $4^{n-1} \times 4^{n-1}$ submatrices, then the

inverse transform matrix M_n^{-1} is given by

$$M_n^{-1} = \begin{bmatrix} A^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ A^{-1}BC^{-1} & C^{-1} & E^{-1}DC^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & E^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & E^{-1}FG^{-1} & G^{-1} \end{bmatrix}. \quad (5.31)$$

The full class derived from (5.28) is generated by replacing different numbers of the non-zero submatrices by the identity submatrix I_{n-1} . Here, the number of the identity submatrix I_{n-1} in the forward matrices of this class is restricted up to two. This class is denoted as Class 1.

The sub-class A has the simplest relation between the forward and inverse matrices and presented in Table 5.3. The submatrix I_{n-1} only appears in the matrix locations (2, 1), (2, 3) or (4, 3) following Definition 5.1.2, which are not in the diagonal.

Property 5.3.2. For any forward matrix M_a of QLIH transform in sub-class A, its corresponding inverse matrix M_a^{-1} can be obtained from M_a directly by the following rule,

Replace I_{n-1} with $(M_{n-1}^{-1})^2$ ($I_{n-1} \Rightarrow (M_{n-1}^{-1})^2$) after self inversion of the forward matrix.

When the submatrix I_{n-1} is in the diagonal, the QLIH transform matrices will be no longer self-inverse. Hence in such a case, the relation between the forward and inverse matrices will be changed. Table 5.4 gives four pairs of QLIH transform matrices, which have only one submatrix I_{n-1} in the diagonal for each forward matrix.

Table 5.3: Sub-class A of QLH transforms.

a1	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
a2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & (M_{n-1}^{-1})^2 & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
a3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}$
a4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & (M_{n-1}^{-1})^2 & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
a5	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}$
a6	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & (M_{n-1}^{-1})^2 & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}$

Table 5.4: Sub-class B of QLIH transforms.

b1	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
b2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
b3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$

It can be seen that when the inverse matrices are calculated from the forward matrices in sub-class B, all the non-zero submatrices in the same column will change, as well as the non-zero submatrices in the same row. These rules are given in Property 5.3.3.

Property 5.3.3. For any QLIH transform matrix M_b in sub-class B, its inverse matrix M_b^{-1} can be obtained following the rule below:

Replace M_{n-1} with I_{n-1} ($M_{n-1} \Rightarrow I_{n-1}$) if and only if M_{n-1} and I_{n-1} are in the same column or in the same row of the forward QLIH transform matrix of order N .

When two submatrices I_{n-1} are in the diagonal of the forward matrix of QLIH transforms, some non-zero submatrices M_{n-1} are in the same columns or rows of both two submatrices I_{n-1} . This kind of QLIH transforms are categorized into sub-class C and Property 5.3.3 was revised to Property 5.3.4 to derive inverse matrices of QLIH transforms in this class.

Property 5.3.4. For any QLIH transform matrix M_c in sub-class C, its inverse matrix M_c^{-1} can be obtained by the following rules:

- $M_{n-1} \Rightarrow I_{n-1}$ if and only if M_{n-1} and only one of the two submatrices I_{n-1} are in the same column or in the same row of the forward QLIH transform matrix.
- M_{n-1} keeps unchanged with its symbol in the inverse matrix M_c^{-1} if and only if M_{n-1} shares the same column or row with both of the two submatrices I_{n-1} in the forward QLIH transform matrix.

There are five derivations of forward QLIH transform matrices that have two

Table 5.5: Sub-class C of QLIH transforms.

c1	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
c2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
c3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix}$
c4	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
c5	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$
c6	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$

Table 5.6: Sub-class D1 of QLIH transforms.

d1.1	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
d1.2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
d1.3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
d1.4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ (M_{n-1}^{-1})^2 & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$

identity sub-matrices I_{n-1} in the diagonal. All the forward and inverse QLIH transform matrices of sub-class C are presented in Table 5.5.

If one of the two identity submatrices I_{n-1} is not in the diagonal but in one of the locations of (2, 1), (2, 3) or (4, 3), and the other identity submatrix I_{n-1} is still in the diagonal, then twelve different QLIH transforms will be derived, which are categorized into sub-class D. According to different positions of the identity submatrix I_{n-1} in the QLIH transform matrices, these QLIH transforms are separated into three groups, which are called d1, d2 and d3. All the forward and inverse transforms of sub-class D are given in Tables 5.6 – 5.7.

In group d1, each forward QLIH transform matrix has one submatrix I_{n-1} in the position (2, 1) following Definition 5.1.2. Each forward QLIH transform matrix in group d2 has one submatrix I_{n-1} in the position (2, 3), and each forward matrix in group d3 has one submatrix I_{n-1} in (4, 3).

Property 5.3.5. For any QLIH transform matrix M_d in sub-class D, its inverse matrix M_d^{-1} can be obtained by using the following rules:

- $M_{n-1} \Rightarrow I_{n-1}$ or $I_{n-1} \Rightarrow M_{n-1}^{-1}$ if and only if the non-zero submatrix M_{n-1} or I_{n-1} are in the same column or in the same row as the submatrix I_{n-1} , which is not in the diagonal of the forward QLIH transform matrix.
- Replace I_{n-1} , which is not in the diagonal of the forward matrix, with $(M_{n-1}^{-1})^2$ ($I_{n-1} \Rightarrow (M_{n-1}^{-1})^2$), if and only if I_{n-1} is in neither the same column nor the same row as the other submatrix I_{n-1} .

As described before, the QLIH transform matrix in (5.29) is the transpose of the QLIH transform matrix in (5.28). Its inverse matrix is also the transpose of the matrix in (5.30) as shown below,

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (5.32)$$

Another class of QLIH transforms can be derived from (5.29) and (5.32) by replacing the non-zero submatrices by one or two identity submatrix I_{n-1} in different locations. This class is denoted as Class 2. Each forward or inverse

Table 5.7: Sub-class D2 of QLIH transforms.

d2.1	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & (M_{n-1}^{-1})^2 & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
d2.2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
d2.3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
d2.4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & (M_{n-1}^{-1})^2 & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix}$

Table 5.8: Sub-class D3 of QLIH transforms.

d3.1	$\begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} I_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}$
d3.2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & I_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & (M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}$
d3.3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
d3.4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & I_{n-1} & I_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & I_{n-1} \end{bmatrix}$

QLIH transform matrix in Class 2 has a corresponding forward or inverse QLIH transform matrix in Class 1, due to the following transpose relations.

Property 5.3.6. Each forward QLIH transform matrix $M_{(1)}$ and its inverse $M_{(1)}^{-1}$ in Class 1 has a corresponding forward QLIH transform matrix $M_{(2)}$ and the inverse $M_{(2)}^{-1}$ in Class 2 for which the transpose relations are described below:

$$[M_{(1)}]^T = M_{(2)} \quad (5.33)$$

and

$$[M_{(1)}^{-1}]^T = M_{(2)}^{-1}. \quad (5.34)$$

Due to Property 5.3.6, all the QLIH transforms in Class 2 can be categorized into four sub-classes as Class 1. Properties 5.3.2 – 5.3.5 also apply to the four sub-classes of Class 2, respectively.

There are also other two classes of QLIH transforms that have similar properties as the transforms in Classes 1 and 2. The two permutation operators given in Definitions 5.1.3 and 5.1.4 are useful for deriving these two classes of QLIH transforms. The new classes are denoted as Classes 3 and 4.

The relations between Classes 1 and 3 are presented in Properties 5.3.7 and 5.3.8.

Property 5.3.7. Each forward QLIH transform matrix $M_{(3)}$ in Class 3 has a corresponding QLIH transform matrix $M_{(1)}$ in Class 1 by the following relation:

$$M_{(3)} = \mu_{EH} [M_{(1)}]. \quad (5.35)$$

Property 5.3.8. Each inverse QLIH transform matrix $M_{(3)}^{-1}$ in Class 3 has a

corresponding inverse QLIH transform matrix $M_{(1)}^{-1}$ in Class 1 by the following relation:

$$M_{(3)}^{-1} = \mu_{EV} \left[M_{(1)}^{-1} \right]. \quad (5.36)$$

From (5.28) and by Property 5.3.7, the forward matrix of the basic QLIH transform in Class 3 is

$$M_n = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (5.37)$$

Also from (5.30) and by Property 5.3.8, the inverse matrix of the basic QLIH transform in Class 3 is

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (5.38)$$

The QLIH transforms in Class 4 are derived from Class 2 by horizontally permuting the forward matrices and vertically permuting the inverse matrices. Similar properties as Properties 5.3.7 and 5.3.8 also exist between Classes 2 and 4.

The forward and inverse QLIH transform matrices in Class 4 that are derived

from (5.29) and (5.30), are given by the following equations, respectively:

$$M_n = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}, \quad (5.39)$$

and

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (5.40)$$

It should be noticed that in Classes 3 and 4, the identity submatrix I_{n-1} is replaced by the reverse-identity submatrix J_{n-1} of size $4^{n-1} \times 4^{n-1}$, which has all its elements equal to zero except the elements in the reverse diagonal equal to 1. The reverse-identity submatrix J_{n-1} is the horizontal or vertical permutation result of the identity submatrix I_{n-1} .

When the number of the identity submatrix I_{n-1} in the forward transform matrices increase to three or more, some of them are overlapped with the QLIH transforms in the introduced four classes, but there are still some transform matrices that are not included in this thesis and they will not be discussed due to their more complicated structures and properties.

5.3.2 Experimental results

All QLIH transform matrices in the sub-class A to C of Class 1 have been implemented using Microsoft Visual C++ and tested by revised Quaternary

Table 5.9: Number of non-zero spectral coefficients for sub-class A.

QBF	$a1$	$a2$	$a3$	$a4$	$a5$	$a6$	QRM^{-1}
9sym	343	468	462	363	343	454	758
apex4	999	992	1004	971	999	985	1022
clip	930	676	940	750	930	814	874
con1	147	111	155	103	147	148	148
ex5	247	223	240	223	247	247	256
ex1010	1016	1014	1015	1014	1016	1015	1024
inc	198	198	206	206	198	196	211
misex1	186	175	171	172	186	186	142
rd84	206	229	205	212	206	208	251
squar5	59	58	60	60	59	59	57
xor5	33	34	46	39	33	39	31
z5xp1	238	209	256	230	238	235	230

Benchmark Functions (QBFs).

Tables 5.9 – 5.11 presents the experimental results for number of non-zero spectral coefficients for the six QLIH transforms in sub-class A as well as quaternary Reed-Muller (QRM) transform. It can be seen that for almost all tested QBFs, some or all of the QLIH transforms obtain smaller number of non-zero spectral coefficients than QRM. Only for functions squar5 and xor5, QRM has better results than all six QLIH transforms.

Their calculation speed is also much faster than for the QRM transform due to the fact that most of their elements in the matrices are zero. Only the calculation times for all QLIH transforms in sub-class A are presented in Table 5.12 compared with QRM transform.

Table 5.10: Number of non-zero spectral coefficients for sub-class B.

QBF	$b1$	$b2$	$b3$	$b4$	QRM^{-1}
9sym	312	468	378	492	758
apex4	673	1002	660	1010	1022
clip	527	705	562	922	874
con1	99	108	92	146	148
ex5	225	223	238	242	256
ex1010	940	1016	957	1015	1024
inc	166	206	163	220	211
misex1	163	173	169	162	142
rd84	247	229	239	231	251
squar5	48	59	46	60	57
xor5	30	34	32	44	31
z5xp1	169	198	168	256	230

Table 5.11: Number of non-zero spectral coefficients for sub-class C.

QBF	$c1$	$c2$	$c3$	$c4$	$c5$	$c6$	QRM^{-1}
9sym	355	388	356	333	510	369	758
apex4	673	660	665	543	1008	673	1022
clip	508	526	613	465	781	628	874
con1	102	95	119	93	133	118	148
ex5	236	243	251	244	248	248	256
ex1010	940	957	956	911	1014	939	1024
inc	160	161	169	141	217	166	211
misex1	184	186	181	178	184	174	142
rd84	242	240	237	250	218	242	251
squar5	47	45	46	38	59	48	57
xor5	25	25	33	23	39	33	31
z5xp1	149	149	182	138	224	182	230

Table 5.12: Calculation times for sub-class A.

QBF	<i>a1</i>	<i>a2</i>	<i>a3</i>	<i>a4</i>	<i>a5</i>	<i>a6</i>	QRM^{-1}
9sym	0.01	0.01	0.03	0.03	0.03	0.03	0.05
apex4	0.16	0.16	0.16	0.19	0.16	0.16	0.34
clip	0.15	0.06	0.06	0.05	0.06	0.05	0.13
con1	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.01
ex5	0.06	0.08	0.06	0.06	0.06	0.05	0.11
ex1010	0.09	0.08	0.09	0.08	0.09	0.09	0.19
inc	0.01	0.02	<0.01	0.02	0.01	0.02	0.12
misex1	0.02	0.01	<0.01	0.02	<0.01	0.03	0.02
rd84	<0.01	0.03	0.01	0.02	<0.01	0.02	0.02
squar5	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.01
xor5	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.01
z5xp1	0.03	0.03	0.02	0.02	0.02	<0.01	0.02

5.4 Classes of fastest quaternary linearly independent transforms

In this section, four new classes of fastest quaternary linearly independent (FQLI) transforms are presented as well as their properties and relations. All the FQLI transform matrices are recursively defined and follow Definitions 5.1.1 and 5.1.2. The FQLI transforms require the lowest computational costs when compared to Reed-Muller transform over $GF(4)$ and have fast transforms.

5.4.1 Properties of FQLI transform

The submatrix Y_{n-1} is important for deriving FQLI transforms. It is a $4^{n-1} \times 4^{n-1}$ square matrix with all its elements zero except one non-zero element. The position of the non-zero element is the same as the position of submatrix Y_{n-1} in M_n . Zero submatrix O_{n-1} is also used, which has all its elements equal to zero. With the introduction of submatrix Y_{n-1} , all the FQLI transform matrices

have the same simple recursive structure that leads to their properties of higher computational efficiency and lowest costs.

Table 5.13 gives all the forward and inverse FQLI transform matrices in Class 1. It is clear that in this class, the submatrix Y_{n-1} could be located in any of the six slots in the lower-left side of the diagonal. It means that all non-zero elements are also located in the lower-left side of the diagonal in the matrices. There are altogether six pairs of forward and inverse FQLI transform matrices in Class 1. The non-zero element of submatrix Y_{n-1} could be any value x , $x \in \left(1, 2, 3\right)$. The following property describes the relations between the forward and inverse transform matrices of FQLI transforms in Class 1.

Property 5.4.1. For any forward transform matrix M_n of FQLI in Class 1, its corresponding inverse matrix M_n^{-1} can be obtained from M_n directly by:

$$M_n^{-1} = M_n. \quad (5.41)$$

There are other six pairs of forward and inverse FQLI transform matrices, which are categorized into Class 2. For any matrix in Class 2, all non-zero elements are located in the upper-right side of the diagonal in the matrices. Table 5.14 presents all the FQLI transform matrices of Class 2.

Property 5.4.1 also applies to all the corresponding forward and inverse transform matrices in Class 2. The relations between the FQLI transform matrices of Classes 1 and 2 are presented in the following properties:

Property 5.4.2. For any forward matrix M_b that belongs to Class 2, there must be a corresponding forward matrix M_a from Class 1, which has the following

Table 5.13: FQLI transforms in Class 1.

a1	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ xY_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ xY_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
a2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & xY_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & xY_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
a3	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & xY_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & xY_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
a4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ xY_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ xY_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
a5	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & xY_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & xY_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
a6	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ xY_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ xY_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$

Table 5.14: FQLI transforms in Class 2.

b1	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & xY_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & xY_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b2	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & xY_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & xY_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b3	$\begin{bmatrix} M_{n-1} & xY_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & xY_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b4	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & xY_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & xY_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b5	$\begin{bmatrix} M_{n-1} & O_{n-1} & xY_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & xY_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
b6	$\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & xY_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & xY_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$

relation with M_b ,

$$M_b = \mu_{EH}\mu_{EV} [M_a]. \quad (5.42)$$

Property 5.4.3. If M_b^{-1} and M_a^{-1} are the inverse transform matrices of M_b and M_a , respectively, then the following relation also exists,

$$M_b^{-1} = \mu_{EH}\mu_{EV} [M_a^{-1}]. \quad (5.43)$$

If all non-zero elements in the forward FQLI transform matrices are restricted to be in the upper-left side of the reverse-diagonal, six new FQLI transforms will be derived and categorized into Class 3. Table 5.15 shows all the six pairs of forward and inverse FQLI transform matrices in Class 3. It is clear that all non-zero elements in the inverse FQLI transform matrices are located in the low-right side of the reverse-diagonal in the matrices.

The following property describes the relation between the forward and inverse transform matrices in Class 3.

Property 5.4.4. For any forward transform matrix M_n of FQLI in Class 3, its corresponding inverse transform matrix M_n^{-1} can be obtained from M_n by vertical and horizontal permutation,

$$M_n^{-1} = \mu_{EH}\mu_{EV} [M_n]. \quad (5.44)$$

The following two properties show the relations among matrices of three classes presented till now.

Property 5.4.5. For any forward matrix M_c that belongs to Class 3, there must be a corresponding forward matrix M_a from Class 1 and M_b from Class 2,

Table 5.15: FQLI transforms in Class 3.

c1	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ xY_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & xY_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
c2	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & xY_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & xY_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
c3	$\begin{bmatrix} O_{n-1} & O_{n-1} & xY_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & xY_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
c4	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ xY_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & xY_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
c5	$\begin{bmatrix} O_{n-1} & xY_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & xY_{n-1} & O_{n-1} \end{bmatrix}$
c6	$\begin{bmatrix} xY_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & xY_{n-1} \end{bmatrix}$

which have the following relations with M_c ,

$$M_c = \mu_{EV} [M_a] \quad (5.45)$$

and

$$M_c = \mu_{EH} [M_b]. \quad (5.46)$$

Property 5.4.6. If M_c^{-1} , M_b^{-1} and M_a^{-1} are the inverse transform matrices of M_c , M_b and M_a , respectively, then the following relations also exist,

$$M_c^{-1} = \mu_{EH} [M_a^{-1}] \quad (5.47)$$

and

$$M_c^{-1} = \mu_{EV} [M_b^{-1}]. \quad (5.48)$$

There are still six FQLI transforms that are categorized into Class 4. All non-zero elements are located in the lower-right side of the reverse-diagonal in the forward transform matrices and the upper-left side of the reverse-diagonal in the inverse transform matrices. They are given in Table 5.16. It is obvious that the forward transform matrices in Class 4 are the corresponding inverse transform matrices in Class 3 and vice versa.

Property 5.4.4 also applies to all the FQLI transform matrices in Class 4. Similar properties as Properties 5.4.5 and 5.4.6 can also be obtained for Class 4.

Table 5.16: FQLI transforms in Class 4.

d1	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & xY_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ xY_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d2	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & xY_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & xY_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d3	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & xY_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & xY_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d4	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & xY_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ xY_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d5	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & xY_{n-1} & O_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} O_{n-1} & xY_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
d6	$\begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & xY_{n-1} \end{bmatrix} \Leftrightarrow \begin{bmatrix} xY_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$

5.4.2 Computational costs

By introducing the submatrix Y_{n-1} , all FQLI transforms of the four presented classes have the same simplest structures and this results in very low computational costs when compared with quaternary Reed-Muller transform. If only 2-place addition is considered, the computational cost for any n -variable FQLI transform is given by

$$S_n = 4S_{n-1} + 1 \quad (5.49)$$

It is clear that as $S_1 = 1$, then from (5.49),

$$S_n = (4^n - 1)/3 \quad (5.50)$$

Example 5.4.1. Let a 2-variable quaternary switching function be

$$\vec{F} = \left[2 \ 0 \ 1 \ 1 \ 1 \ 0 \ 3 \ 3 \ 1 \ 1 \ 2 \ 0 \ 3 \ 2 \ 2 \ 3 \right]^T.$$

Let the FQLI transform a4 from Class 1 be applied to the given quaternary switching function, and $x=3$.

$$M_2^{-1} = \begin{bmatrix} M_1^{-1} & O_1 & O_1 & O_1 \\ O_1 & M_1^{-1} & O_1 & O_1 \\ 3Y_1 & O_1 & M_1^{-1} & O_1 \\ O_1 & O_1 & O_1 & M_1^{-1} \end{bmatrix}.$$

Then, the obtained spectral coefficients vector is

$$\vec{A} = \left[2 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 3 \ 1 \ 1 \ 0 \ 0 \ 3 \ 2 \ 0 \ 3 \right]^T.$$

During the calculation, only five 2-place additions have been used as shown by (5.50). The zero elements in $\vec{A}$ is extended by 4 when the quaternary switching function is converted to the spectral domain by FQLI transform a4.

By (5.4), the quaternary switching function can be presented as

$$f = 2 \cdot g_0 + g_3 + g_4 + 3 \cdot g_7 + g_8 \\ + g_9 + 3 \cdot g_{12} + 2 \cdot g_{13} + 3 \cdot g_{15},$$

where different g_i are obtained from the forward matrix of the FQLI transform a4. All the operations are performed over GF(4). $\square$

5.4.3 Experimental results

Table 5.17 presents the experimental results of non-zero spectral coefficients for the six FQLI transforms from Class 1 and quaternary Reed-Muller (QRM) transform. It is obvious that for all the tested benchmark functions, some or all of the FQLI transforms obtain less non-zero spectral coefficients than QRM transform.

Their calculation speed is also much faster than for the QRM transform due to the fact that most of their elements in the matrices are zero. The calculation times for all the transforms are also given in Table 5.18.

Table 5.17: Number of non-zero spectral coefficients for FQLI.

QBF	$a1$	$a2$	$a3$	$a4$	$a5$	$a6$	QRM^{-1}
9sym	677	488	606	313	466	597	758
apex4	853	597	763	566	575	764	1022
clip	824	579	743	502	555	737	874
con1	203	139	164	101	131	174	148
ex5	240	224	239	240	239	256	256
ex1010	864	853	836	856	841	848	1024
inc	174	121	155	93	118	154	211
misex1	98	104	109	97	109	112	142
rd84	255	191	250	255	250	247	251
squar5	48	35	45	33	34	45	57
xor5	26	19	23	26	17	22	31
z5xp1	213	149	192	149	144	192	230

Table 5.18: Calculation times for FQLI.

QBF	$a1$	$a2$	$a3$	$a4$	$a5$	$a6$	QRM^{-1}
9sym	0.02	0.03	0.03	0.03	0.01	0.02	0.05
apex4	0.14	0.14	0.16	0.14	0.14	0.14	0.34
clip	0.06	0.06	0.06	0.06	0.05	0.05	0.13
con1	0.01	0.01	0.01	0.01	0.01	0.02	0.01
ex5	0.05	0.05	0.06	0.06	0.06	0.06	0.11
ex1010	0.08	0.08	0.08	0.08	0.08	0.09	0.19
inc	0.01	0.01	0.01	0.01	0.02	0.02	0.12
misex1	0.02	0.01	0.01	0.02	0.02	0.01	0.02
rd84	0.01	0.01	0.01	0.01	0.01	0.01	0.02
squar5	0.01	0.01	0.01	0.01	0.01	0.01	0.01
xor5	0.01	0.01	0.01	0.03	0.01	0.01	0.01
z5xp1	0.01	0.01	0.01	0.02	0.01	0.02	0.02

5.5 New approach to generate QLI transforms from binary case

In this section, we developed new types of Quaternary Linearly Independent (QLI) transforms that are different than the known multiple-valued transforms for quaternary case with some special properties of generating their matrices directly from binary LI transforms. As there exist a huge number of different QLI expansions, there is a great interest in finding such QLI expansions that have fast transforms and this problem has been solved here. Since recursive equations for both forward and inverse transforms are provided, hence, the presented results can be extended easily to transform matrices with higher dimensions, giving the best sets of basis function for such cases as well.

5.5.1 Generation of QLI transforms

There are four basic linearly independent transform matrices over GF(2). They are shown below:

$$\begin{bmatrix} O_n & M_n \\ M_n & M_n \end{bmatrix} \quad \begin{bmatrix} M_n & M_n \\ M_n & O_n \end{bmatrix} \quad \begin{bmatrix} M_n & O_n \\ M_n & M_n \end{bmatrix} \quad \begin{bmatrix} M_n & M_n \\ O_n & M_n \end{bmatrix}$$

The recursive structure of the binary linearly independent transforms has four submatrices each with the size $2^{n-1} \times 2^{n-1}$. The binary linearly independent transform matrices of the order 2^N have the same matrix structure as the QLI transform matrices of the order N . All the matrices of QLI transforms have the size $4^n \times 4^n$ and thus they are closely related in their submatrices structures to the binary LI transforms as shown in Fig. 5.1.

$$\begin{bmatrix} O_n & M_n \\ M_n & M_n \end{bmatrix} \Rightarrow \begin{bmatrix} \begin{pmatrix} O_n & O_n \\ O_n & O_n \end{pmatrix} & \begin{pmatrix} O_n & M_n \\ M_n & M_n \end{pmatrix} \\ \begin{pmatrix} O_n & M_n \\ M_n & M_n \end{pmatrix} & \begin{pmatrix} O_n & M_n \\ M_n & M_n \end{pmatrix} \end{bmatrix}$$

Figure 5.1: Relation between binary and quaternary linearly independent transforms.

All the non-zero elements in the QLI transform presented in Fig. 5.1 are '1's. This matrix has the simplest structure among all QLI transforms. It can be separated into 4 submatrices, each with the size $2^{2n-1} \times 2^{2n-1}$. The upper-left submatrix is a zero submatrix, which has all its elements zero; other three submatrices can be view as the corresponding binary LI transform matrices, which have the order $2N - 1$. When the submatrices are multiplied by some factor (2 or 3) over GF(4), seven QLI transforms are generated efficiently and they are presented in Table 5.19. These QLI transforms are categorized as Class 1.

The inverse transform matrix of QLI transform matrix 1.a is

$$M_n^{-1} = \begin{bmatrix} \begin{pmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} \end{pmatrix} & \begin{pmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} \end{pmatrix} \\ \begin{pmatrix} M_{n-1}^{-1} & M_{n-1}^{-1} \\ M_{n-1}^{-1} & O_{n-1} \end{pmatrix} & \begin{pmatrix} O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} \end{pmatrix} \end{bmatrix}. \quad (5.51)$$

This inverse matrix can be directly derived from the forward transform matrix by flipping all the 16 submatrices along the reverse diagonal. This property also applies to other two QLI transforms 1.b and 1.c in Table 5.19, which are multiplied by factors 2 or 3 at the lower-right $2^{2n-1} \times 2^{2n-1}$ submatrix of the

Table 5.19: Forward and inverse QLI transforms in Class 1.

a	$\left[\begin{array}{c} \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ \begin{pmatrix} M & \\ M & M \end{pmatrix} \begin{pmatrix} M & M \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
b	$\left[\begin{array}{c} \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ \begin{pmatrix} M & \\ M & M \end{pmatrix} 2 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} 2 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
c	$\left[\begin{array}{c} \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ \begin{pmatrix} M & \\ M & M \end{pmatrix} 3 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
d	$\left[\begin{array}{c} 2 \times \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ \begin{pmatrix} M & \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
e	$\left[\begin{array}{c} \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ 2 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
f	$\left[\begin{array}{c} \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ 2 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} 2 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$
g	$\left[\begin{array}{c} 2 \times \begin{pmatrix} & M \\ M & M \end{pmatrix} \begin{pmatrix} M & \\ M & M \end{pmatrix} \\ \begin{pmatrix} M & \\ M & M \end{pmatrix} 2 \times \begin{pmatrix} M & \\ M & M \end{pmatrix} \end{array} \right] \Leftrightarrow \left[\begin{array}{c} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \\ 3 \times \begin{pmatrix} M^{-1} & M^{-1} \\ M^{-1} & \end{pmatrix} \end{array} \right]$

forward transform matrices.

For other four QLI transform matrices 1.d – 1.g in this class, the $2^{2n-1} \times 2^{2n-1}$ submatrices in the lower-left or upper-right corner are multiplied by factors 2 or 3. The inverse transform matrices are obtained by using the following steps:

1. Flipping all the sixteen submatrices along the reverse-diagonal of the forward transform matrix.
2. Multiplying all four $2^{2n-1} \times 2^{2n-1}$ submatrices by 3 over GF(4).

Similarly, from the second basic binary linearly independent transform matrices, other seven QLI transform matrices can be derived and they are grouped into Class 2. The inverse transform matrices of the QLI transforms in Class 2 can be derived directly from the forward transform matrices by the same steps as for the QLI transforms in Class 1. The following properties also exist between these two classes.

Property 5.5.1. Let $M_{(1)}$ and $M_{(2)}$ be any QLI transform matrices of order N in Classes 1 and 2, respectively. Then,

$$\mu_{EH}(\mu_{EV}[M_{(1)}]) = M_{(2)}. \quad (5.52)$$

Let $M_{(1)}^{-1}$ and $M_{(2)}^{-1}$ be the inverse matrices of $M_{(1)}$ and $M_{(2)}$, respectively. Then,

$$\mu_{EH}(\mu_{EV}[M_{(1)}^{-1}]) = M_{(2)}^{-1}. \quad (5.53)$$

For QLI transforms 2.b and 2.c in Class 2, their lower-left $2^{2n-1} \times 2^{2n-1}$ submatrices are multiplied by factors 2 and 3, respectively. The inverse transform

matrices of the last four QLI transform matrices in this class can be obtained by multiplying all four $2^{2n-1} \times 2^{2n-1}$ submatrices in every forward matrix by factor 3 over GF(4).

There are also other two classes of QLI transforms derived from the last two basic binary linearly independent transform matrices. They are denoted as Classes 3 and 4, each also containing seven QLI transforms. The first QLI transform in Class 3 is given below:

$$M_n = \left[\begin{array}{c} \left(\begin{array}{cc} M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} \end{array} \right) \\ \left(\begin{array}{cc} M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} \end{array} \right) \end{array} \right] \left[\begin{array}{c} \left(\begin{array}{cc} O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} \end{array} \right) \\ \left(\begin{array}{cc} M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} \end{array} \right) \end{array} \right]. \quad (5.54)$$

It is a self-inverse transform, which means that the inverse transform matrix is the same as the forward transform matrix itself.

Property 5.5.2. Let $M_{(3)}$ and $M_{(4)}$ be any QLI transform matrices of order N in Classes 3 and 4, respectively. Then,

$$\mu_{EH} (\mu_{EV} [M_{(3)}]) = M_{(4)}. \quad (5.55)$$

Let $M_{(3)}^{-1}$ and $M_{(4)}^{-1}$ be the inverse matrices of $M_{(3)}$ and $M_{(4)}$, respectively. Then,

$$\mu_{EH} (\mu_{EV} [M_{(3)}^{-1}]) = M_{(4)}^{-1}. \quad (5.56)$$

Property 5.5.3. Let $M_{(1)}$ and $M_{(3)}$ be any QLI transform matrices of order N

Table 5.20: Forward butterfly diagrams for QLI transforms 1.b – 4.b.

1.b	$\left[\begin{array}{c} \left(\begin{array}{cc} & M \\ M & \end{array} \right) \\ \left(\begin{array}{cc} M & \\ & M \end{array} \right) \end{array} \right] 2 \times \left(\begin{array}{cc} & M \\ M & \end{array} \right) \Rightarrow$
2.b	$\left[\begin{array}{c} 2 \times \left(\begin{array}{cc} M & \\ & M \end{array} \right) \\ \left(\begin{array}{cc} M & M \\ & \end{array} \right) \end{array} \right] \Rightarrow$
3.b	$\left[\begin{array}{c} \left(\begin{array}{cc} M & \\ & M \end{array} \right) \\ 2 \times \left(\begin{array}{cc} M & \\ & M \end{array} \right) \end{array} \right] \Rightarrow$
4.b	$\left[\begin{array}{c} \left(\begin{array}{cc} M & \\ & M \end{array} \right) \\ 2 \times \left(\begin{array}{cc} M & \\ & M \end{array} \right) \end{array} \right] \Rightarrow$

in Classes 1 and 3, respectively. Then,

$$\mu_{EH} [M_{(1)}] = M_{(3)}. \quad (5.57)$$

Let $M_{(1)}^{-1}$ and $M_{(3)}^{-1}$ be the inverse matrices of $M_{(1)}$ and $M_{(3)}$, respectively. Then,

$$\mu_{EV} [M_{(1)}^{-1}] = M_{(3)}^{-1}. \quad (5.58)$$

Similar relation as Property 5.5.3 also exists for Classes 2 and 4.

In previous sections, an efficient matrix decomposition method was intro-

Table 5.21: Number of non-zero spectral coefficients for QLI transforms in Class 1.

QBF	1.a	1.b	1.c	1.d	1.e	1.f	1.g	QRM^{-1}
9sym	420	932	824	824	420	824	420	758
apex4	1002	1000	1006	1006	1002	1006	1002	1022
clip	480	588	734	734	480	746	480	874
con1	40	44	84	84	40	88	40	148
ex5	256	256	256	256	256	256	256	256
ex1010	1023	1024	1023	1023	1023	1023	1023	1024
inc	182	240	238	238	182	240	182	211
misex1	60	104	120	120	60	120	60	142
rd84	107	201	202	202	107	223	107	251
squar5	46	58	60	60	46	56	46	57
xor5	10	26	40	40	10	40	10	31
z5xp1	122	162	224	224	122	226	122	230

duced to produce the forward and inverse butterfly diagrams of ternary LI transforms. Similar approach can also be applied to QLI transforms discussed here. As an example, Table 5.20 presents the basic forward butterfly diagrams of four QLI transforms 1.b – 4.b from Classes 1 – 4 based on the matrix decomposition method.

5.5.2 Experimental results

Table 5.21 gives the number of non-zero coefficients for the seven QLI transforms in Class 1 and for QRM. It is clear that QLI transforms can always provide smaller number of non-zero coefficients when compared to QRM for each revised quaternary benchmark function (QBF). The calculation speed of these QLI transform matrices is very fast due to their recursive structure.

Chapter 6

Quaternary Linearly Independent Arithmetic Transforms

Similar to the relation between the algebra over Galois field (3) and ternary standard algebra, linearly independent logic based on Galois field (4) can also be transferred into quaternary linearly independent standard algebra. In this chapter, Quaternary Arithmetic Helix (QAH) transforms and Fastest Quaternary Linearly Independent Arithmetic (FQLIA) transforms have been classified and their properties have been shown. Their computational costs and experimental results on quaternary benchmark functions are also shown and compared with Quaternary Arithmetic (QA) transform. Their applications in logic synthesis and circuit design are also discussed.

6.1 Basic definitions

Definition 6.1.1. Let M_n be a $4^n \times 4^n$ matrix with columns corresponding to some quaternary functions of n variables. If the set of columns is linearly independent with respect to quaternary standard algebra, then M_n has one unique inverse M_n^{-1} , and is said to be linearly independent, i.e.

$$M_n \cdot M_n^{-1} = I_n \quad (6.1)$$

where I_n is a $4^n \times 4^n$ identity matrix and all the operations are performed over standard algebra.

Definition 6.1.2. The linearly independent transform based on Definition 6.1.1 can be described by the following general formulae,

$$M_n \cdot \vec{A} = \vec{F} \quad (6.2)$$

and

$$M_n^{-1} \cdot \vec{F} = \vec{A} \quad (6.3)$$

where $\vec{F} = \begin{bmatrix} f_0, & f_1, & \dots, & f_{4^n-1} \end{bmatrix}^T$ is a column vector defining the truth vector of a quaternary function $f(x_n)$ in natural quaternary ordering, M_n is a matrix of order $N = 4^n$ defined by any linearly independent set of n -variable quaternary functions and $\vec{A} = \begin{bmatrix} a_0, & a_1, & \dots, & a_{4^n-1} \end{bmatrix}^T$ is the coefficient column vector for the particular transform matrix M_n with the inverse M_n^{-1} while T is the matrix transpose operator.

Definition 6.1.3. Formula (6.2) can be expressed as

$$f(x_n) = \sum_{i=0}^{4^n-1} a_i g_i \quad (6.4)$$

where g_i represents the truth vector of an n -variable quaternary function, such that the matrix $M_n = \begin{bmatrix} g_0 & g_1 & \dots & g_{4^n-1} \end{bmatrix}^T$, $0 \leq i \leq 4^n - 1$, and the symbol $\sum$ is the standard arithmetic addition.

Definition 6.1.4. Let M_n be a matrix of order N following Definition 6.1.1. Additionally, M_n can be partitioned into 16 submatrices M_{n-1} with matrix size $4^{n-1} \times 4^{n-1}$ as shown in the following equation,

$$M_n = \begin{bmatrix} M_{n-1}^{(1,1)} & \dots & M_{n-1}^{(1,4)} \\ \dots & \dots & \dots \\ M_{n-1}^{(4,1)} & \dots & M_{n-1}^{(4,4)} \end{bmatrix} \quad (6.5)$$

Definition 6.1.5. The μ_{EH} operator on M_n is defined as grouping the recursive equations in all the submatrices vertically and interchanging them in the submatrices horizontally.

$$\mu_{EH}[M_n] = \begin{bmatrix} M_{n-1}^{(1,4)} & \dots & M_{n-1}^{(1,1)} \\ \dots & \dots & \dots \\ M_{n-1}^{(4,4)} & \dots & M_{n-1}^{(4,1)} \end{bmatrix} \quad (6.6)$$

Definition 6.1.6. The μ_{EV} operator on M_n is defined as grouping the recursive equations in all the submatrices horizontally and interchanging them in the

submatrices vertically.

$$\mu_{EV}[M_n] = \begin{bmatrix} M_{n-1}^{(4,1)} & \dots & M_{n-1}^{(4,4)} \\ \dots & \dots & \dots \\ M_{n-1}^{(1,1)} & \dots & M_{n-1}^{(1,4)} \end{bmatrix} \quad (6.7)$$

6.2 Quaternary linearly independent arithmetic transforms

There are four types of general Quaternary Linearly Independent Arithmetic (QLIA) transforms which will be discussed in this section. Their recursively defined transform matrices and mutual relations are given below, which are separated by the superscript symbols $a - d$.

The four types of general QLIA transform matrices are

$${}^a M_n = \begin{bmatrix} M_{n-1}^{(1,1)} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{(2,1)} & M_{n-1}^{(2,2)} & O_{n-1} & O_{n-1} \\ M_{n-1}^{(3,1)} & O_{n-1} & M_{n-1}^{(3,3)} & O_{n-1} \\ M_{n-1}^{(4,1)} & M_{n-1}^{(4,2)} & M_{n-1}^{(4,3)} & M_{n-1}^{(4,4)} \end{bmatrix}; \quad (6.8)$$

$${}^b M_n = \begin{bmatrix} M_{n-1}^{(1,1)} & M_{n-1}^{(1,2)} & M_{n-1}^{(1,3)} & M_{n-1}^{(1,4)} \\ O_{n-1} & M_{n-1}^{(2,2)} & O_{n-1} & M_{n-1}^{(2,4)} \\ O_{n-1} & O_{n-1} & M_{n-1}^{(3,3)} & M_{n-1}^{(3,4)} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{(4,4)} \end{bmatrix}; \quad (6.9)$$

$${}^cM_n = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{(1,4)} \\ O_{n-1} & O_{n-1} & M_{n-1}^{(2,3)} & M_{n-1}^{(2,4)} \\ O_{n-1} & M_{n-1}^{(3,2)} & O_{n-1} & M_{n-1}^{(3,4)} \\ M_{n-1}^{(4,1)} & M_{n-1}^{(4,2)} & M_{n-1}^{(4,3)} & M_{n-1}^{(4,4)} \end{bmatrix}; \quad (6.10)$$

$${}^dM_n = \begin{bmatrix} M_{n-1}^{(1,1)} & M_{n-1}^{(1,2)} & M_{n-1}^{(1,4)} & M_{n-1}^{(1,4)} \\ M_{n-1}^{(2,1)} & O_{n-1} & M_{n-1}^{(2,3)} & O_{n-1} \\ M_{n-1}^{(3,1)} & M_{n-1}^{(3,2)} & O_{n-1} & O_{n-1} \\ M_{n-1}^{(4,1)} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (6.11)$$

These QLIA transforms contain only two kind of submatrices: M_{n-1} and the zero submatrix O_{n-1} . Hence they can be extended based on Kronecker product. It is clear that based on the permutation operators defined in Definitions 6.1.5 and 6.1.6, the following relations exist,

Property 6.2.1. For the QLIA transform matrices aM_n and cM_n with the same order N ,

$${}^cM_n = \mu_{EH} [{}^aM_n]. \quad (6.12)$$

Property 6.2.2. For the QLIA transform matrices aM_n and dM_n with the same order N ,

$${}^dM_n = \mu_{EV} [{}^aM_n]. \quad (6.13)$$

Property 6.2.3. For the QLIA transform matrices aM_n and bM_n with the same order N ,

$${}^bM_n = \mu_{EH}\mu_{EV} [{}^aM_n] \quad (6.14)$$

Their inverse transforms can be easily derived and are presented in Table

6.1. If the symbol “ $||$ ” denotes the operator to obtain absolute value on every element inside the matrix, the following properties are presented,

Property 6.2.4. For the QLIA transform matrices aM_n and bM_n with the same order N , the inverse transform matrices of their have the same absolute matrices’ values as their forward transform matrices as shown by

$$|{}^aM_n^{-1}| = {}^aM_n \quad (6.15)$$

and

$$|{}^bM_n^{-1}| = {}^bM_n \quad (6.16)$$

Property 6.2.5. For the QLIA transform matrices cM_n and dM_n with the same order N , the absolute matrix’ value of one of them is the same as the forward transform matrix of the other as shown by

$$|{}^cM_n^{-1}| = {}^dM_n \quad (6.17)$$

and

$$|{}^dM_n^{-1}| = {}^cM_n \quad (6.18)$$

From each of these general matrices, a family of QLIA transforms can be derived when replacing some of the non-zero submatrix M_{n-1} by either identity submatrix I_{n-1} (on aM_n or bM_n) or reverse identity submatrix J_{n-1} (on cM_n or dM_n). Due to the permutation relations presented in Properties 6.2.1 – 6.2.3, the properties of one type of QLIA transform can be easily extended to apply on other three types of QLIA transforms. Hence in this section, we will concentrate on discussing the properties of QLIA transforms derived from

Table 6.1: Four types of QLIA transforms.

${}^a M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}$	$\Leftrightarrow$	${}^a M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & -M_{n-1}^{-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$
${}^b M_n = \begin{bmatrix} M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}$	$\Leftrightarrow$	${}^b M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & -M_{n-1}^{-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & -M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & -M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}$
${}^c M_n = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & O_{n-1} & M_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} & M_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}$	$\Leftrightarrow$	${}^c M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & -M_{n-1}^{-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \\ -M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ -M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$
${}^d M_n = \begin{bmatrix} M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}$	$\Leftrightarrow$	${}^d M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & M_{n-1}^{-1} & -M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} & -M_{n-1}^{-1} \\ M_{n-1}^{-1} & -M_{n-1}^{-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}$

the general transform aM_n . All these QLIA transforms are classified based on the number of identity submatrix I_{n-1} contained in their forward transform matrices. When one non-zero submatrix in the general QLIA transform matrix is replaced by the identity submatrix I_{n-1} , nine new QLIA transforms will be derived and they are categorized as Class A.

If the general QLIA matrix is divided into four separated square areas, each including four submatrices, as shown below,

$$M_n = \left[\begin{array}{cc} \left(\begin{array}{cc} M_{n-1}^{(1,1)} & O_{n-1} \\ M_{n-1}^{(2,1)} & M_{n-1}^{(2,2)} \end{array} \right)^1 & \left(\begin{array}{cc} O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} \end{array} \right) \\ \left(\begin{array}{cc} M_{n-1}^{(3,1)} & O_{n-1} \\ M_{n-1}^{(4,1)} & M_{n-1}^{(4,2)} \end{array} \right)^3 & \left(\begin{array}{cc} M_{n-1}^{(3,3)} & O_{n-1} \\ M_{n-1}^{(4,3)} & M_{n-1}^{(4,4)} \end{array} \right)^2 \end{array} \right], \quad (6.19)$$

then there are two locations for the only one identity submatrix I_{n-1} in the matrix. One is the lower-left corner of each square area, such as (2, 1), (4, 1) and (4, 3); the other one is the location of all other non-zero submatrices.

By using symbolically calculation of the matrix in (6.19), a general inverse matrix can be obtained and also used to derive the entire inverse transform matrices in this section.

Rule 6.2.1.

$$\text{Let } M_n = \left[\begin{array}{cccc} A & O_{n-1} & O_{n-1} & O_{n-1} \\ B & C & O_{n-1} & O_{n-1} \\ D & O_{n-1} & E & O_{n-1} \\ F & G & H & K \end{array} \right] \text{ represent the matrix defined in (6.19),}$$

where A - K denote $4^{n-1} \times 4^{n-1}$ submatrices, then the inverse transform matrix

M_n^{-1} is given by

$$M_n^{-1} = \begin{bmatrix} A^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -A^{-1}BC^{-1} & C^{-1} & O_{n-1} & O_{n-1} \\ -A^{-1}DE^{-1} & O_{n-1} & E^{-1} & O_{n-1} \\ Q & -C^{-1}GK^{-1} & -E^{-1}HK^{-1} & K^{-1} \end{bmatrix} \quad (6.20)$$

where

$$Q = -A^{-1} (F + BC^{-1}G + DE^{-1}H) K^{-1}. \quad (6.21)$$

The inverse QLIA transform matrices of Class A can be derived directly from the forward QLIA transform matrices by Rule 6.2.1 as also presented in Table 6.2.

Example 6.2.1. One of the QLIA transform matrices in Class A is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & M_{n-1} \end{bmatrix}.$$

The submatrix I_{n-1} is in the sub-area 2 but not in the lower-left corner (4, 3). By Rule 6.2.1, the inverse QLIA transform matrix is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -I_{n-1} & O_{n-1} & I_{n-1} & O_{n-1} \\ I_{n-1} & -M_{n-1}^{-1} & -I_{n-1} & M_{n-1}^{-1} \end{bmatrix}.$$

□

Table 6.2 presents all the forward and inverse QLIA transforms in Class A. It can be seen that the inverse transform matrices of different QLIA transforms in Class A contain different number of identity submatrix I_{n-1} , hence, their computational costs can not be given in an unified formula. Nevertheless the direct computational cost of the forward transform matrices in Class A can be given by the following property.

Property 6.2.6. For a QLIA transforms M_A in Class A with order N , its computational cost based on direct matrix method is given by

$$S_n^A = 10 \cdot 8^{n-1} - 5 \cdot 4^{n-1} \quad (6.22)$$

When the number of the identity submatrix I_{n-1} in the forward QLIA transform matrix increases to two, thirty six new QLIA transforms can be derived from the general QLIA transform in (6.19). They are categorized into Class B. They can be divided into three sub-classes according to the locations of the two identity submatrices I_{n-1} in the forward transform matrices.

There are six QLIA transforms, which have both two identity submatrices in the diagonal of the matrices. They are categorized into Class B.1. Rule 6.2.1 also applies to obtain their inverse transform matrices of QLIA transforms in Class B.1.

Table 6.2: QLIA transforms in Class A.

[illegible]

Example 6.2.2. One of the QLIA transform in Class B.1 is given by:

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & I_{n-1} \end{bmatrix}.$$

The two identity submatrices are in location (2, 2) and (3, 3). It can also be divided into four sub-areas as shown in (6.19). Following Rule 6.2.1, the inverse QLIA transform matrix is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -I_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1} & -M_{n-1} & -I_{n-1} & I_{n-1} \end{bmatrix}.$$

The submatrices M_{n-1} in (4, 1) and (4, 2) of the inverse matrix are equal to the submatrix M_{n-1} in the forward matrix. $\square$

If both identity submatrices are not in the diagonal of the forward QLIA transform matrix, ten new QLIA transforms will be derived, which are categorized into Class B.2.

The inverse QLIA transform matrices of Class B.2 can be derived from the forward QLIA transform matrices by Rule 6.2.1.

There are still twenty QLIA transforms that are categorized into Class B.3. Each QLIA transform matrix in Class B.3 has two identity submatrices I_{n-1} and only one submatrix I_{n-1} is in the diagonal of the forward QLIA transform

matrix, which is denoted by I_{n-1}^1 , and the other submatrix I_{n-1} is denoted by I_{n-1}^2 .

Rule 6.2.1 also applies to the QLIA transforms in Class B.3 to derived their inverse transform matrices.

The forward transform matrices of all QLIA transform in Class B have the same computational costs based on direct matrix method due to the fact that the number of identity submatrix I_{n-1} is the same.

Property 6.2.7. For a QLIA transforms M_B in Class B with order N , its computational cost based on direct matrix method is given by

$$S_n^B = (31 \cdot 7^{n-1} - 4^{n+1})/3 \quad (6.23)$$

There are 84 new QLIA transforms that can be derived from the general matrix, which contain three identity submatrices I_{n-1} in each of their matrices. They are categorized into Class C which can be also divided into four sub-classes, C.1 – C.4, according to the different number of identity submatrices located in the matrix diagonal.

In (6.24), there is only one identity submatrix located in the diagonal.

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix}. \quad (6.24)$$

Its inverse transform matrix is given by

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -I_{n-1} & I_{n-1} & O_{n-1} & O_{n-1} \\ -(M_{n-1}^{-1})^2 & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ Q & -I_{n-1} & -(M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}, \quad (6.25)$$

where $Q = (M_{n-1}^{-1})^3 - M_{n-1}^{-1} + I_{n-1}$.

Forty QLIA transforms similar to the transform given in (6.24) are categorized into Class C.1.

Class C.2 contains thirty QLIA transform matrices which have two identity submatrices I_{n-1} located in the diagonal of the forward transform matrices. One of the transforms is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1}^1 & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ M_{n-1} & M_{n-1} & I_{n-1}^3 & M_{n-1} \end{bmatrix}. \quad (6.26)$$

In (6.26), two identity submatrices I_{n-1}^1 and I_{n-1}^2 are located in the diagonal (2, 2) and (3, 3). The third identity submatrix I_{n-1}^3 is in location (4, 3). The inverse transform matrix of (6.26) are given in (6.27) following Rule 6.2.1. .

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -I_{n-1} & I_{n-1}^1 & O_{n-1} & O_{n-1} \\ -I_{n-1} & O_{n-1} & I_{n-1}^2 & O_{n-1} \\ I_{n-1} & -I_{n-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (6.27)$$

When all the three identity submatrices I_{n-1} are located in the diagonal, four derivations of matrices are categorized into Class C.3. One example is given in (6.28).

$$M_n = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1}^2 & O_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & M_{n-1} & I_{n-1}^3 \end{bmatrix}. \quad (6.28)$$

Its inverse transform matrix is given by

$$M_n^{-1} = \begin{bmatrix} I_{n-1}^1 & O_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1} & I_{n-1}^2 & O_{n-1} & O_{n-1} \\ -I_{n-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ (M_{n-1})^2 & -M_{n-1} & -I_{n-1} & I_{n-1}^3 \end{bmatrix}. \quad (6.29)$$

The last ten QLIA transforms in Class C have none of their three identity submatrices I_{n-1} in the diagonal of the matrices. They are included in Class C.4. One of the forward transform matrices is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ M_{n-1} & M_{n-1} & O_{n-1} & O_{n-1} \\ I_{n-1} & O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & I_{n-1} & I_{n-1} & M_{n-1} \end{bmatrix}. \quad (6.30)$$

In (6.30), none of the identity submatrices I_{n-1} is in the matrix diagonal. Following Rule 6.2.1, the inverse transform matrix of (6.30) is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -(M_{n-1}^{-1})^2 & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ Q & -(M_{n-1}^{-1})^2 & -(M_{n-1}^{-1})^2 & M_{n-1}^{-1} \end{bmatrix}, \quad (6.31)$$

where $Q = (M_{n-1}^{-1})^3 + (M_{n-1}^{-1})^2 - M_{n-1}^{-1}$.

The forward transform matrices of all QLIA transform in Class C have the same computational costs based on direct matrix method which is given by Property 6.2.8.

Property 6.2.8. For a QLIA transforms M_C in Class C with order N , its computational cost based on direct matrix method is given by

$$S_n^C = 11 \cdot 6^{n-1} - 6 \cdot 4^{n-1} \quad (6.32)$$

There are altogether 130 new QLIA transforms that have been investigated in this section. They are derived from the general matrix aM_n . Based on Properties 6.2.1 – 6.2.3, all the presented QLIA transforms can be easily derived into other three families which are based on the general QLIA transform matrices from (6.9) to (6.11). All properties presented in this section also apply to the other three families of QLIA transforms.

6.2.1 Calculation of QLIA spectrum

In this subsection, a complete example is presented to show the calculation of quaternary spectral coefficients based on the four general QLIA transforms.

A random input quaternary switching function with $n = 2$ is given in the following truth vector

$$\vec{F} = \begin{bmatrix} 0 & 2 & 2 & 1 & 2 & 0 & 0 & 3 & 2 & 0 & 0 & 3 & 3 & 1 & 0 & 1 \end{bmatrix}^T. \quad (6.33)$$

From previous discussion, the inverse transform matrix of aM_n is given by

$${}^aM_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ -M_{n-1}^{-1} & O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & -M_{n-1}^{-1} & -M_{n-1}^{-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (6.34)$$

The spectral coefficients' vector can be obtained by direct matrix multiplication as

$$\begin{aligned} \vec{A}^a &= {}^aM_2^{-1} \cdot \vec{F} \\ &= \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \right) \cdot \vec{F} \\ &= \begin{bmatrix} 0 & 2 & 2 & -3 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & -1 & -2 \end{bmatrix}^T. \end{aligned}$$

Hence, from Definition 6.1.3 and (6.4), the switching function can be also

written as

$$f^a = 2 \cdot (g_1 + g_2 + g_4 + g_8 - g_{15}) - 3 \cdot (g_3 - g_{12}) - g_{14}. \quad (6.35)$$

When we look into the matrix structure of the inverse transform matrix of aM_n , the following property exists.

Property 6.2.9. For the inverse QLIA transform ${}^aM_n^{-1}$, the sum of all elements in row j is equal to zero, when $4^{n-1} \leq j \leq 4^n - 1$.

It can also be seen from (6.34) that the sum of the second row to the fourth row are equal to zero. It indicates that the input data of the truth vector can be separated into blocks with size 4^{n-1} and be calculated one by one.

The truth vector $\vec{F}$ can be written as

$$\vec{F} = \left[\vec{F}_1 \quad \vec{F}_2 \quad \vec{F}_3 \quad \vec{F}_4 \right]^T. \quad (6.36)$$

where

$$\begin{aligned} \vec{F}_1 &= \begin{bmatrix} 0 & 2 & 2 & 1 \end{bmatrix}^T; \\ \vec{F}_2 &= \begin{bmatrix} 2 & 0 & 0 & 3 \end{bmatrix}^T; \\ \vec{F}_3 &= \begin{bmatrix} 2 & 0 & 0 & 3 \end{bmatrix}^T; \\ \vec{F}_4 &= \begin{bmatrix} 3 & 1 & 0 & 1 \end{bmatrix}^T. \end{aligned}$$

If the spectral coefficients' vector $\vec{A}$ are written as

$$\vec{A} = \begin{bmatrix} \vec{A}_1 & \vec{A}_2 & \vec{A}_3 & \vec{A}_4 \end{bmatrix}^T, \quad (6.37)$$

Then, each block of $\vec{A}$ can be expressed by

$$\begin{aligned} \vec{A}_1 &= \vec{F}_1 \cdot M_{n-1}; \\ \vec{A}_2 &= (\vec{F}_2 - \vec{F}_1) \cdot M_{n-1}; \\ \vec{A}_3 &= (\vec{F}_3 - \vec{F}_1) \cdot M_{n-1}; \\ \vec{A}_4 &= (\vec{F}_4 - \vec{F}_3 - \vec{F}_2 + \vec{F}_1) \cdot M_{n-1}. \end{aligned}$$

It can be seen that the final results can be obtained from the products of each block with M_{n-1} , which are denoted by $\vec{A}_1^*$ to $\vec{A}_4^*$. For those blocks which have the same value, such as $\vec{F}_2$ and $\vec{F}_3$, the intermediates can be repeatedly used, hence, the calculation can be simplified.

$$\begin{aligned}
\vec{A}_1^* &= {}^a M_1 \cdot \vec{F}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 2 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 2 \\ -3 \end{bmatrix}; \\
\vec{A}_2^* &= {}^a M_{n-1} \cdot \vec{F}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 0 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}; \\
\vec{A}_3^* &= \vec{A}_2^* = \begin{bmatrix} 2 & 2 & 2 & 1 \end{bmatrix}^T; \\
\vec{A}_4^* &= {}^a M_{n-1} \cdot \vec{F}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 3 \\ 3 \end{bmatrix}.
\end{aligned}$$

Then, from the relations between the intermediates and sub-vectors of $\vec{A}$,

$$\begin{aligned}
\vec{A}_1 &= \vec{A}_1^* = \begin{bmatrix} 0 & 2 & 2 & -3 \end{bmatrix}^T; \\
\vec{A}_2 &= \vec{A}_2^* - \vec{A}_1^* = \begin{bmatrix} 2 & 0 & 0 & 0 \end{bmatrix}^T; \\
\vec{A}_3 &= \vec{A}_2 = \begin{bmatrix} 2 & 0 & 0 & 0 \end{bmatrix}^T; \\
\vec{A}_4 &= \vec{A}_4^* - \vec{A}_3^* - \vec{A}_2^* + \vec{A}_1^* = \begin{bmatrix} 3 & 0 & -1 & -2 \end{bmatrix}^T.
\end{aligned}$$

The spectral coefficients' vectors calculated from other three general QLIA

transforms are

$$\begin{aligned}\vec{A}^b &= \begin{bmatrix} 3 & 3 & 2 & 0 & -1 & -1 & 0 & 2 & -1 & -1 & 0 & 2 & 3 & 0 & -1 & 1 \end{bmatrix}^T; \\ \vec{A}^c &= \begin{bmatrix} 1 & -1 & 0 & 3 & 2 & 0 & -1 & -1 & 2 & 0 & -1 & -1 & 0 & 2 & 3 & 3 \end{bmatrix}^T; \\ \vec{A}^d &= \begin{bmatrix} -2 & -1 & 0 & 3 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & -3 & 2 & 2 & 0 \end{bmatrix}^T.\end{aligned}$$

6.3 Quaternary arithmetic helix transforms based on Kronecker product

In this section, properties of Quaternary Arithmetic Helix (QAH) transforms are investigated and compared with the best known quaternary transform, i.e. Quaternary Arithmetic (QA) transform. The introduced QAH transforms have very regular structure that results in their fast computation and some interesting properties. Relations and properties between different QAH transform matrices are also shown. As analyzed in computational costs, the introduced QAH transforms are much faster in calculation of their spectra than QA transform. The concept of QAH transforms is also extended into general case of transform matrices of order N .

6.3.1 Four basic types of QAH transforms

There are four types of Quaternary Arithmetic Helix (QAH) transforms. They are based on four different 4×4 kernel matrices and expended by Kronecker product. In this section, their important properties and some relations between them are discussed. All QAH transforms follow Definitions 6.1.1 – 6.1.3.

The 4×4 kernel matrix of the first QAH transform is given below, which is named as Right-Positive-Helix (*RPH*) transform,

$$RPH_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}. \quad (6.38)$$

RPH transform matrix of order N is derived from the kernel matrix by Kronecker product as shown in the following equation,

$$RPH_n = RPH_{n-1} \otimes RPH_1 = \overset{n-1}{\otimes} RPH_1. \quad (6.39)$$

The inverse transform matrix of *RPH* of order N can be derived by calculating their Kronecker product from the inverse transform matrix of (6.38).

$$RPH_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}. \quad (6.40)$$

The inverse transform matrix RPH^{-1} has the same matrix structure as the forward *RPH* transform. This relation is presented in Property 6.3.1.

Property 6.3.1. Let RPH^{-1} be the inverse of *RPH* transform of order N , then

$$|RPH_n^{-1}| = RPH_n. \quad (6.41)$$

where the symbol " $||$ " is the operator to obtain absolute value on every element inside the matrix.

In the RPH transform matrix of (6.38), the non-zero elements are all '1's. The three '1's that are not in the diagonal of the matrix can be multiplied together by the parameter x , $x \in (1, 2, 3)$, that is used in all of this section. The new matrices can be recognized as the derivations of RPH . They have the same relations between the forward and inverse transform matrices hence Property 6.3.1 also applies to them. The general RPH transform matrix is presented below,

$$RPH_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ x & 1 & x & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & x & 1 \end{bmatrix}. \quad (6.42)$$

From Definition 6.1.3 and by (6.2), the above matrix form can also be written as

$$\begin{aligned} f_{0\langle RPH \rangle} &= a_0 \\ f_{1\langle RPH \rangle} &= x \cdot a_0 + a_1 + x \cdot a_2 \\ f_{2\langle RPH \rangle} &= a_2 \\ f_{3\langle RPH \rangle} &= x \cdot a_2 + a_3. \end{aligned}$$

The inverse RPH transform is

$$RPH_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -x & 1 & -x & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -x & 1 \end{bmatrix}. \quad (6.43)$$

It can also be presented by the equations from Definition 6.1.2,

$$\begin{aligned} a_{0\langle RPH \rangle} &= f_0 \\ a_{1\langle RPH \rangle} &= -x \cdot f_0 + f_1 - x \cdot f_2 \\ a_{2\langle RPH \rangle} &= f_2 \\ a_{3\langle RPH \rangle} &= -x \cdot f_2 + f_3 \end{aligned}$$

Another QAH transform called Right-Negative-Helix (RNH) transform is given below,

$$RNH_1 = \begin{bmatrix} 1 & x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & x & 1 & x \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.44)$$

The above matrix form can also be represented by the following equations,

$$\begin{aligned} f_{0\langle RNH \rangle} &= a_0 + x \cdot a_1 \\ f_{1\langle RNH \rangle} &= a_1 \\ f_{2\langle RNH \rangle} &= x \cdot a_1 + a_2 + x \cdot a_3 \\ f_{3\langle RNH \rangle} &= a_3. \end{aligned}$$

For the matrix of order N ,

$$RNH_n = RNH_{n-1} \otimes RNH_1 = \overset{n-1}{\otimes} RNH_1. \quad (6.45)$$

The 4×4 basic matrix of the inverse transform RNH^{-1} is

$$RNH_1^{-1} = \begin{bmatrix} 1 & -x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -x & 1 & -x \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.46)$$

Property 6.3.1 also applies to RNH transform. The following properties present the relations between RPH and RNH transforms.

Property 6.3.2. For the forward transform matrices of order N ,

$$RNH_n = \mu_{EV} [\mu_{EH} (RPH_n)]. \quad (6.47)$$

Property 6.3.3. For the inverse transform matrices of order N ,

$$RNH_n^{-1} = \mu_{EV} [\mu_{EH} (RPH_n^{-1})]. \quad (6.48)$$

There are still two other QAH transforms. They are denoted as Left-Positive-Helix transform (LPH) and Left-Negative-Helix (LNH) transform.

The LPH transform can be expressed by the following equations:

$$\begin{aligned} f_{0(LPH)} &= a_3 \\ f_{1(LPH)} &= x \cdot a_1 + a_2 + x \cdot a_3 \\ f_{2(LPH)} &= a_1 \\ f_{3(LPH)} &= a_0 + x \cdot a_1. \end{aligned}$$

When comparing the equations of RPH and LPH , if the subscripts of the coefficients a of RPH are interchanged between 0 and 3 ($a_0 \Leftrightarrow a_3$), and between 1 and 2 ($a_1 \Leftrightarrow a_2$), the equations of RPH are converted into the equations of LPH . Due to this property, the 4×4 kernel matrix of LPH is the vertical permutation of RPH . The general LPH transform matrix is given by,

$$LPH_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & x & 1 & x \\ 0 & 1 & 0 & 0 \\ 1 & x & 0 & 0 \end{bmatrix}. \quad (6.49)$$

The 4×4 basic matrix of the inverse transform LPH^{-1} is

$$LPH_1^{-1} = \begin{bmatrix} 0 & 0 & -x & 1 \\ 0 & 0 & 1 & 0 \\ -x & 1 & -x & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}. \quad (6.50)$$

The LNH transform is given by the following equations:

$$\begin{aligned} f_{0\langle LNH \rangle} &= x \cdot a_2 + a_3 \\ f_{1\langle LNH \rangle} &= a_2 \\ f_{2\langle LNH \rangle} &= x \cdot a_0 + a_1 + x \cdot a_2 \\ f_{3\langle LPH \rangle} &= a_0. \end{aligned}$$

The general LNH transform matrix is given by,

$$LNH_1 = \begin{bmatrix} 0 & 0 & x & 1 \\ 0 & 0 & 1 & 0 \\ x & 1 & x & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \quad (6.51)$$

It is obvious that the inverse transform matrix of LPH transform has the same matrix structure as LNH transform. The same relation also exists between LPH transform and the inverse LNH transform. These relations are given in Properties 6.3.4 – 6.3.5.

When comparing the equations of RNH and LNH , it can be seen that the 4×4 kernel matrix of LNH is the vertical permutation of RNH similar to the relationship between RPH and LPH .

Property 6.3.4. For the forward transform matrix LPH and the inverse transform matrix LNH^{-1} ,

$$|LNH^{-1}| = LPH. \quad (6.52)$$

Property 6.3.5. For the forward transform matrix RPH and the inverse trans-

form matrix RNH^{-1} ,

$$|RNH^{-1}| = RPH. \quad (6.53)$$

The following two properties are also given.

Property 6.3.6. For the forward transform matrices of order N ,

$$LPH_n = \mu_{EH} (RPH_n). \quad (6.54)$$

Property 6.3.7. For the forward transform matrices of order N ,

$$LNH_n = \mu_{EV} (RPH_n). \quad (6.55)$$

When the matrix structures of the helix transform are detected, some properties are presented in the following discussion, which are useful in hardware realization.

Property 6.3.8. In the inverse transform matrices RPH^{-1} , the negative elements are located in every even number of columns or every odd number of rows.

Property 6.3.9. In the inverse transform matrices RNH^{-1} , the negative elements are located in every even number of rows or every odd number of columns.

Similar properties can be obtained for the inverse transform matrices of LPH and LNH based on Properties 6.3.4 – 6.3.7.

6.3.2 A complete example

Let a quaternary switching function be represented by its truth vector

$$\vec{F} = \left[0 \ 3 \ 3 \ 3 \ 0 \ 2 \ 3 \ 3 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 3 \ 3 \right]^T.$$

When $n = 2$, the inverse RPH transform is

$$\begin{aligned} \vec{A} &= RPH_2^{-1} \cdot \vec{F} \\ &= \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \right) \cdot \vec{F} \\ &= \left[0 \ 0 \ 3 \ 0 \ 0 \ -1 \ -1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ -2 \ -2 \ 0 \right]^T \end{aligned}$$

By Definition 6.1.3 and from (6.4),

$$f_{(RPH)} = 3 \cdot g_2 - g_5 - g_6 + g_{10} - 2 \cdot g_{13} - 2 \cdot g_{14}.$$

The number of non-zero elements in the quaternary truth vector $\vec{F}$ is 12 and it was reduced to 6 in the spectral coefficients' vector $\vec{A}$ of RPH .

6.3.3 Computational complexity

The number of non-zero elements in the matrix of QAH transforms determines the additions required to calculate the spectrum. The four QAH transforms have the same computational cost due to their close relations.

In Table 6.3, all four QAH transforms together with their forward butterfly diagrams for $n = 2$ are presented. The dashed lines correspond to the values of the parameter x .

The total number of 2-place additions to compute QAH transforms of any n -variable quaternary function by direct matrix computational method is

$$S_n = 7^n - 4^n. \quad (6.56)$$

Based on the fast transform method presented in previous sections, the additions are reduced to

$$S'_n = n \cdot (7 \cdot 4^{n-1} - 4^n) = 3n \cdot 4^{n-1}. \quad (6.57)$$

Table 6.4 gives the comparison of the computational costs between the QAH transforms and quaternary arithmetic (QA) transform both in the direct matrix computation method and through the fast transform algorithms as described in [Gre86, Gre90].

6.3.4 Experimental results

The four introduced QAH transforms have regular structure that makes their computationally efficient. In this section, *RPH* and *RNH* transforms have been implemented using Microsoft Visual C++ and tested by modified quaternary benchmark functions.

Table 6.5 presents the number of non-zero spectral coefficients obtained from quaternary benchmark functions for *RPH* and *RNH*. The parameter x of the

Table 6.3: QAH transforms and their corresponding butterfly diagrams when $n = 2$.

<i>RPH</i>	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ x & 1 & x & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & x & 1 \end{bmatrix}$
<i>LPH</i>	$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & x & 1 & x \\ 0 & 1 & 0 & 0 \\ 1 & x & 0 & 0 \end{bmatrix}$
<i>RNH</i>	$\begin{bmatrix} 1 & x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & x & 1 & x \\ 0 & 0 & 0 & 1 \end{bmatrix}$
<i>LNH</i>	$\begin{bmatrix} 0 & 0 & x & 1 \\ 0 & 0 & 1 & 0 \\ x & 1 & x & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

Table 6.4: Computational costs for QAH transforms.

n	<i>QA</i>		QA^{-1}		<i>QAH</i>	
	direct	fast	direct	fast	direct	fast
	$13^n - 4^n$	$9n \cdot 4^{n-1}$	$11^n - 4^n$	$7n \cdot 4^{n-1}$	$7^n - 4^n$	$3n \cdot 4^{n-1}$
1	9	9	7	7	3	3
2	153	72	105	56	33	24
3	2133	432	1267	336	279	144
4	28305	2304	14385	1792	2145	768
5	370269	11520	160027	8960	15783	3840
6	4822713	55296	1767465	43008	113553	18432
7	62732133	258048	19470787	200704	807159	86016

Table 6.5: Number of non-zero spectral coefficients for RPH and RNH transforms.

QBF	RPH^{-1}			RNH^{-1}			QA^{-1}
	$x = 1$	$x = 2$	$x = 3$	$x = 1$	$x = 2$	$x = 3$	
9sym	392	420	392	166	420	166	98
apex4	1005	965	1017	511	510	511	1014
clip	811	804	928	415	496	458	769
con1	132	154	191	73	108	97	90
ex5	208	256	250	226	256	256	254
ex1010	1017	973	1016	1009	966	1010	1024
inc	238	199	251	100	104	103	204
misex1	145	204	212	129	188	184	93
rd84	230	239	226	232	239	240	139
squar5	60	55	60	31	32	32	53
xor5	41	16	41	21	16	21	6
z5xp1	235	254	254	118	128	127	194

two QAH transforms changes from 1 to 3. It is clear that for most quaternary benchmark functions, the two QAH transforms obtain less non-zero spectral coefficients than QA transform. The calculation times of the two QAH transforms as well as for QA transform are also given in Table 6.6.

6.4 Fastest quaternary linearly independent arithmetic transforms

In this section, four types of Fastest Quaternary Linearly Independent Arithmetic (FQLIA) transforms will be presented. Some properties as well as the relations between different FQLIA transforms will also be discussed. The matrices of all FQLIA transforms are recursively defined and follow the basic definitions given in the following subsections. The four types of FQLIA transforms are denoted by using symbols aM to dM .

Table 6.6: Calculation times for RPH and RNH transforms.

QBF	RPH^{-1}			RNH^{-1}			QA^{-1}
	$x = 1$	$x = 2$	$x = 3$	$x = 1$	$x = 2$	$x = 3$	
9sym	0.02	0.02	0.02	0.02	0.02	0.02	0.01
apex4	0.16	0.16	0.16	0.14	0.16	0.16	0.19
clip	0.05	0.05	0.06	0.06	0.06	0.05	0.06
con1	0.02	<0.01	<0.01	0.01	0.02	<0.01	<0.01
ex5	0.05	0.06	0.06	0.04	0.06	0.06	0.06
ex1010	0.08	0.08	0.08	0.07	0.08	0.09	0.09
inc	0.02	0.02	0.02	0.01	0.01	<0.01	0.01
misex1	<0.01	0.01	0.02	<0.01	<0.01	0.02	0.02
rd84	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.01
squar5	<0.01	<0.01	<0.01	0.01	0.01	<0.01	<0.01
xor5	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.02
z5xp1	0.02	0.01	0.01	0.02	0.02	<0.01	<0.01

6.4.1 Classification and properties of FQLIA transforms

The FQLIA transforms are constructed using only three kinds of submatrices, M_{n-1} , O_{n-1} and Y_{n-1} . The submatrix O_{n-1} is a $4^{n-1} \times 4^{n-1}$ square submatrix with all its elements equal to zero. The submatrix Y_{n-1} is very important to define the new introduced FQLIA transforms. It is a $4^{n-1} \times 4^{n-1}$ square matrix with all its elements equal to zero except only one that is equal to 1. The non-zero element is based in one of the four corners of the submatrix Y_{n-1} , which follows the location of submatrix Y_{n-1} in M_n , and it is multiplied by the parameter x , where $x \in \left(1, \ 2, \ 3\right)$.

The forward matrix of FQLIA transform aM of order N is defined by the

recursive matrix in (6.58):

$${}^aM_n = \begin{bmatrix} {}^aM_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & {}^aM_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & {}^aM_{n-1} & O_{n-1} \\ xY_{n-1} & O_{n-1} & O_{n-1} & {}^aM_{n-1} \end{bmatrix}. \quad (6.58)$$

The submatrix Y_{n-1} is in the lower-left corner of the matrix aM_n . All other non-zero submatrices are along the diagonal of aM_n . The following equation gives the inverse matrix of (6.58):

$${}^aM_n^{-1} = \begin{bmatrix} {}^aM_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & {}^aM_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & {}^aM_{n-1}^{-1} & O_{n-1} \\ -xY_{n-1} & O_{n-1} & O_{n-1} & {}^aM_{n-1}^{-1} \end{bmatrix}. \quad (6.59)$$

The inverse matrix ${}^aM_n^{-1}$ in (6.59) is obtained from the forward matrix directly by simply multiplying the submatrix Y_{n-1} by -1 .

Property 6.4.1. Let ${}^aM_n^{-1}$ be the inverse matrix of the FQLIA transform matrix aM_n with n variables, then

$$|{}^aM_n^{-1}| = {}^aM_n, \quad (6.60)$$

where the symbol " $||$ " is the operator to obtain absolute value on every element inside the matrix.

The forward transform matrix of the second FQLIA transform bM is given

by the following equation:

$${}^bM_n = \begin{bmatrix} {}^bM_{n-1} & O_{n-1} & O_{n-1} & xY_{n-1} \\ O_{n-1} & {}^bM_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & {}^bM_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & {}^bM_{n-1} \end{bmatrix}. \quad (6.61)$$

Its inverse transform matrix is

$${}^bM_n^{-1} = \begin{bmatrix} {}^bM_{n-1}^{-1} & O_{n-1} & O_{n-1} & -xY_{n-1} \\ O_{n-1} & {}^bM_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & {}^bM_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & O_{n-1} & O_{n-1} & {}^bM_{n-1}^{-1} \end{bmatrix}. \quad (6.62)$$

Property 6.4.1 also applies to bM and its inverse matrix ${}^bM^{-1}$. Another property shows that aM and bM have very closed relations. The forward and inverse transform matrices of these two kinds of FQLIA transforms can be derived from each other by the permutation operations given in Definitions 6.1.5 and 6.1.6.

Property 6.4.2. Let aM_n and bM_n be the forward FQLIA transform matrices of order N ; Let ${}^aM_n^{-1}$ and ${}^bM_n^{-1}$ be the inverse transform matrices of aM_n and bM_n , respectively, then

$${}^bM_n = \mu_{EH} [\mu_{EV} ({}^aM_n)], \quad (6.63)$$

and

$${}^bM_n^{-1} = \mu_{EH} [\mu_{EV} ({}^aM_n^{-1})]. \quad (6.64)$$

When only the permutation operator μ_{EH} applies to aM , the third FQLIA transform cM is obtained. The forward transform matrix cM of order N is

$${}^cM_n = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & {}^cM_{n-1} \\ O_{n-1} & O_{n-1} & {}^cM_{n-1} & O_{n-1} \\ O_{n-1} & {}^cM_{n-1} & O_{n-1} & O_{n-1} \\ {}^cM_{n-1} & O_{n-1} & O_{n-1} & xY_{n-1} \end{bmatrix}. \quad (6.65)$$

Property 6.4.3. Let aM_n and cM_n be the forward FQLIA transform matrices of order N , then

$${}^cM_n = \mu_{EH} [{}^aM_n]. \quad (6.66)$$

The inverse transform matrix ${}^cM^{-1}$ is derived from the forward transform matrix cM by Rule 6.4.1.

Rule 6.4.1.

1. Interchanging the two $4^{n-1} \times 4^{n-1}$ submatrices $M_{n-1}^{(1,1)}$ and $M_{n-1}^{(4,4)}$;
2. Multiplying submatrix Y_{n-1} by -1 .

The inverse transform matrix of (6.65) is given below:

$${}^cM_n^{-1} = \begin{bmatrix} -xY_{n-1} & O_{n-1} & O_{n-1} & {}^cM_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & {}^cM_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & {}^cM_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ {}^cM_{n-1}^{-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (6.67)$$

Property 6.4.4. Let ${}^aM_n^{-1}$ and ${}^cM_n^{-1}$ be the inverse matrices of FQLIA trans-

forms aM_n and cM_n of order N , then

$${}^cM_n^{-1} = \mu_{EV} [{}^aM_n^{-1}]. \quad (6.68)$$

The last FQLIA transform can be derived from aM by applying the permutation operator μ_{EV} on the forward matrix.

$${}^dM_n = \begin{bmatrix} xY_{n-1} & O_{n-1} & O_{n-1} & {}^dM_{n-1} \\ O_{n-1} & O_{n-1} & {}^dM_{n-1} & O_{n-1} \\ O_{n-1} & {}^dM_{n-1} & O_{n-1} & O_{n-1} \\ {}^dM_{n-1} & O_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (6.69)$$

Property 6.4.5. Let aM_n and dM_n be the forward FQLIA transform matrices of order N , then

$${}^dM_n = \mu_{EV} [{}^aM_n]. \quad (6.70)$$

The inverse transform matrix ${}^dM^{-1}$ can be obtained directly from the forward matrix following the same steps as for ${}^cM^{-1}$. The inverse matrix of order N is

$${}^dM_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & O_{n-1} & {}^dM_{n-1}^{-1} \\ O_{n-1} & O_{n-1} & {}^dM_{n-1}^{-1} & O_{n-1} \\ O_{n-1} & {}^dM_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ {}^dM_{n-1}^{-1} & O_{n-1} & O_{n-1} & -xY_{n-1} \end{bmatrix}. \quad (6.71)$$

Property 6.4.6. Let ${}^aM_n^{-1}$ and ${}^dM_n^{-1}$ be the inverse matrices of FQLIA transforms aM_n and dM_n of order N , then

$${}^dM_n^{-1} = \mu_{EH} [{}^aM_n^{-1}]. \quad (6.72)$$

Table 6.7: Number of non-zero spectral coefficients for ${}^aM^{-1}$ and ${}^bM^{-1}$.

QBF	${}^aM^{-1}$			${}^bM^{-1}$			QA^{-1}
	$x = 1$	$x = 2$	$x = 3$	$x = 1$	$x = 2$	$x = 3$	
9sym	597	630	597	420	420	420	98
apex4	768	768	764	512	512	512	1014
clip	743	743	737	496	496	496	769
con1	181	177	173	118	118	118	90
ex5	256	256	256	240	240	240	254
ex1010	848	846	848	856	856	856	1024
inc	156	156	154	104	104	104	204
misex1	112	111	112	112	112	112	93
rd84	248	253	249	254	256	250	139
squar5	45	45	45	30	30	30	53
xor5	22	24	22	16	16	16	6
z5xp1	192	192	192	128	128	128	194

6.4.2 Computational complexity and experimental results

The four FQLIA transforms presented in this paper have the same simplest matrix structures that result in their lowest computational costs. If only 2-place addition is considered, the computational costs for any FQLIA transform of order $N = 4^n$ is given by the following equation,

$$S_n = 4S_{n-1} + 1. \tag{6.73}$$

When $n = 1$, $S_1 = 1$. Then, from (6.73),

$$S_n = (4^n - 1) / 3. \tag{6.74}$$

Table 6.7 presents the number of non-zero spectral coefficients obtained from

quaternary benchmark functions for aM and bM . The parameter x of the sub-matrix Y_{n-1} is increased from 1 to 3. It is shown that for 6 out of 12 benchmark functions aM can obtained better results comparing with QA transform, while bM has better performance for 7 out of 12 benchmark functions. It is also clear that that results of bM are always better than those of aM .

Chapter 7

Hardware Implementations

In this chapter, the multiple-valued LI transforms are using in to aspects of hardware implementations. First is the implementations of different LI transforms on systolic processors. The F – R type and fastest ternary LI transforms are given as examples. When the QAH transforms are implemented, it is shown that two sequentially connected systolic structures can solve this input data conflicting in time sequence problem and give compact hardware structures of QAH transforms. It is also shown that the new LI transform can be easily implemented using matrix decomposition method and they have advantages in terms of saving the number of the computing and storage cells of systolic processors. In the following, a universal logical module based on the first stage QLI transforms is proposed.

7.1 Implementation of multiple-valued LI transforms on systolic processor

In this section, the implementation of some types of ternary and quaternary LI transforms are discussed by using revised systolic processors. The systolic processor consists of two basic elements that are repeated n times, namely computing cells and storage cells. Each computing cell performs a particular number of iterations and has two quaternary arithmetic modules, the multiplication and the addition. The output of a computing cell is sent to the input of the next computing cell as well as to the corresponding storage cell, which consists of an FIFO register. The computing cells process different algebras according to different transforms implemented in the systolic processors.

7.1.1 F – R type ternary LI transforms

When the concept of systolic processors is extended to perform all its computation over $GF(3)$, there are some efficient ways to implement F – R type ternary LI transforms.

Let us consider one of the fast LI transforms over $GF(3)$ from Class A as shown in the following equation for the forward transform matrix,

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}. \quad (7.1)$$

The inverse transform matrix of (7.1) has been presented in Table 3.1 as well

as the forward and inverse butterfly diagrams.

Fig. 7.1 gives the systolic processor structure for this LI transform when $n = 2$ to calculate the spectral coefficients. It should be noticed that all the feed-back data from previous computing cells need to be multiplied by 2. All the multiplication and addition operators are performed over GF(3).

There are six addition operators required in this systolic processor structure. When compared with the inverse butterfly diagram of (7.1) which is presented in Table 3.1, it can be seen that the number of addition operators is equal to the number of crossing lines in the inverse butterfly diagram.

One of the F-R type fast LI transforms from Class C is given by

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & M_{n-1} \\ O_{n-1} & M_{n-1} & O_{n-1} \\ M_{n-1} & O_{n-1} & O_{n-1} \end{bmatrix}. \quad (7.2)$$

Its inverse transform matrix is

$$M_n^{-1} = \begin{bmatrix} O_{n-1} & O_{n-1} & M_{n-1}^{-1} \\ O_{n-1} & M_{n-1}^{-1} & O_{n-1} \\ M_{n-1}^{-1} & O_{n-1} & 2M_{n-1}^{-1} \end{bmatrix}. \quad (7.3)$$

Both the forward and inverse butterfly diagrams are presented in Table 3.2. It is clear that the matrix (7.3) can be derived by performing horizontal flipping on the inverse transform matrix of (7.1). This relation is the reason of adding a permutation part in the left side of the inverse butterfly diagram of (7.3). Hence, the same systolic processor structure can be used for implementation of

Figure 7.1: Systolic processor implementation of F – R types ternary LI transforms.

Figure 7.2: Permutation part.

calculating the spectral coefficients of (7.3), by reordering the sequence of input data. Fig. 7.2 shows a simple example of the permutation part.

7.1.2 Fastest ternary LI transforms

In Section 3.5, the concept of ternary fastest LI transforms was introduced. In this part, the hardware realization of the fastest LI transforms will be discussed based on systolic structure.

One forward ternary fastest LI transform is

$$M_n = \begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix}, \quad (7.4)$$

and its inverse transform matrix is

$$M_n^{-1} = \begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix}. \quad (7.5)$$

Figure 7.3: Butterfly diagram of $M_{(7.5)}$ when $n = 2$.

It contains one identity submatrix I_{n-1} and one submatrix Y_{n-1} . This kind of fastest LI transforms have the lowest computational cost and fastest processing speed. Its butterfly diagram is also shown in Fig. 7.3 when $n = 2$.

The systolic processor structure of (7.5) is presented in Fig. 7.4. It should be noticed that five inputs, f_1, f_3, f_4, f_5 and f_7 , are sent to the outputs as a_1, a_3, a_4, a_5 and a_7 directly without any computations. The five steps in the systolic processor structure are omitted in Fig. 7.4 and replaced by dashed lines.

Comparing with the butterfly diagram shown in Fig. 7.3, the number of additions in systolic processor structure which are operated over $\text{GF}(3)$ are equal to the number of crossing lines in the butterfly diagram as shown by Theorem 3.5.1. When $n = 2$, the FIFO structure required eight computing cells to store and process input and output data.

When considering the fastest LI transform in Class Z1 as given by

$$M_n = R \left(\begin{bmatrix} M_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & M_{n-1} \end{bmatrix} \right). \quad (7.6)$$

Figure 7.4: Systolic processor implementation of ternary fastest LI transform in (7.5).

Figure 7.5: Butterfly diagram of $M_{(7.7)}$ when $n = 2$.

Its inverse transform matrix is given by

$$M_n^{-1} = R^{-1} \left(\begin{bmatrix} M_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & I_{n-1} & O_{n-1} \\ 2Y_{n-1} & O_{n-1} & M_{n-1}^{-1} \end{bmatrix} \right). \quad (7.7)$$

It is obvious that the elements from every 3^{n-1} sub-vector of the input vector have to be permuted before they are sent to the computing cells. This property is also shown in its butterfly diagram in Fig. 7.5. So, when implementing the fastest LI transform (7.7) in systolic processor, the same structure shown in Fig. 7.4 can be used for this transform, but the sequence of the inputs has to be reordered. The complete structure map is given in Fig. 7.6.

7.1.3 Quaternary Kronecker arithmetic helix transforms

For the four types of QAH transforms introduced in Section 6.3, they need to be implemented in two parts of systolic structures that are connected in series. The *RPH* transform will be discussed in this section and its implementation in systolic structure will be shown later.

Figure 7.6: Systolic processor implementation of ternary fastest LI transforms in (7.7).

The general RPH transform matrix is given by

$$RPH_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ x & 1 & x & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & x & 1 \end{bmatrix}, \quad (7.8)$$

and the inverse RPH transform is

$$RPH_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -x & 1 & -x & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -x & 1 \end{bmatrix}. \quad (7.9)$$

It can be seen that the non-zero elements in the RPH transform are located along the two sides of the diagonal. It will cause conflicts when the inputs are sent to a FIFO system such as systolic processor. In order to avoid conflicts on the order of the inputs in time sequence, each of the QAH transforms needs to be decomposed into several matrices.

When $n = 2$ and $x = -1$, the inverse RPH transform can be decomposed by the multiplication of two matrices, $RPH_2^{-1} = Y \cdot X$, where the two matrices are given by

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \quad (7.10)$$

Figure 7.7: Inverse butterfly diagram of *RPH* transform when $n = 2$.

and

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (7.11)$$

The inverse butterfly diagram of the 2-variable *RPH* transform are presented in Fig. 7.7. The dashed lines correspond to the value -1 . The left half of the butterfly diagram represents the matrix X , and the right half represents the matrix Y . It can be seen that both the dashed lines in the left half and those in the right half of the butterfly diagram have the same direction.

Based on the decomposition method, the two matrices X and Y are implemented in two sequentially connected systolic processors as shown in Figs. 7.8 and 7.9, respectively. The outputs $t_i, i \in (0, 1, \dots, 15)$, of X are sent to the inputs of Y . The spectral coefficients are obtained from the outputs of Y . It should be noticed that all the feedback routes are performed using an extra

operation, i.e., to multiply -1 on the feedback data before adding it to the output from the previous computing cell. It is obvious that in sign binary notation this multiplication can be just performed by changing MSB.

Comparing Figs. 7.7 – 7.9, each connection point between the dashed line and horizontal line in the butterfly diagram directly corresponds to a quaternary addition operator in the systolic processor implementation, which means the butterfly diagrams of QAH transform can be mapped into systolic processor realization directly.

Fig. 7.10 gives the inverse butterfly diagram of 2-variable *LPH* transform with $x = -1$. It is clear that *LPH* transform can have the same systolic processor realization as shown in Figs. 7.8 and 7.9 for *RPH* transform with reordering of its input signals sent to the processor due to Property 6.3.6.

Similar approach could also be applied for the hardware implementations of *RNH* and *LNH* transforms.

7.2 Logic circuit module of QLI transforms

In Section 5.2.2, a complete example of four QLI transforms, $M_{(a)}$, $M_{(b)}$, $M_{(c)}$ and $M_{(d)}$, was discussed. The inverse transform matrices of the four QLI transforms are given in equations (5.13), (5.15), (5.17) and (5.19). In this section, the logic circuit module of them will be presented and a universal logical module based on first stage QLI transforms will be proposed.

In previous section, a fast algorithm was introduced based on decomposition of partial transform matrices. This method is also applied to the four QLI transforms discussed here and used to derive their butterfly diagrams. The

Figure 7.8: Part X systolic processor of RPH.

Figure 7.9: Part Y systolic processor of RPH.

Figure 7.10: Inverse butterfly diagram of LPH transform when $n = 2$.

inverse butterfly diagrams of QLI transforms $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$ are given in Fig. 7.11 from (a) – (d), respectively.

A universal logical module based on the 1-variable QLI transforms is proposed in Fig. 7.12. For the reason that all the QLI transform matrices only contain two kinds of non-zero submatrices, the presented universal logical module can be cascaded to realize higher level quaternary logic circuits.

When $n = 2$, a simplified logical structure can be derived for each of the four QLI transforms $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$ in Figs. 7.13 – 7.16, respectively. It should be noticed that in the figures, all 16 inputs and outputs are divided into small groups each contains four inputs/outputs in sequence and they are processed by small groups. The symbol $\mathbf{U}$ is used here to denote the universal logical module proposed in Fig. 7.12.

When the generalized universal module is used for the implementations of the QLI transforms, the logic structures presented in Figs. 7.13 – 7.16 can be used for any order of the QLI transforms.

Figure 7.11: Inverse butterfly diagrams for QLI transforms $M_{(a)}^{-1}$, $M_{(b)}^{-1}$, $M_{(c)}^{-1}$ and $M_{(d)}^{-1}$.

Figure 7.12: Universal logic module based on QLI transforms.

Figure 7.13: Logic structure of QLI transform $M_{(a)}^{-1}$.

Figure 7.14: Logic structure of QLI transform $M_{(b)}^{-1}$.

Figure 7.15: Logic structure of QLI transform $M_{(c)}^{-1}$.

Figure 7.16: Logic structure of QLI transform $M_{(d)}^{-1}$.

Chapter 8

Conclusion and Future Work

The main aims of the scientific work in this thesis are the developments of novel spectral transforms and fast algorithms for efficient representations and their hardware implementations of discrete switching functions for multiple-valued logic. The work done has concentrated on ternary and quaternary functions that are the most common among discrete functions. Families of linearly independent transforms have been introduced using Galois field and standard algebra. Their various useful properties and relations are presented. The experimental results of the novel spectral transforms are showing their advantages in terms of calculating spectrum and computational complexity when compared with corresponding ternary and quaternary Reed-Muller or arithmetic transforms. The multiple-valued linearly independent arithmetic transforms also show their application in logic function verification. Several types of the proposed spectral transforms are implemented by either systolic processors or through a universal logic module.

8.1 Conclusion

The following developments have been introduced in this thesis:

1. Fast Linearly Independent Transforms over Galois Field (3)

Two types of fast Linearly Independent (LI) transforms over $GF(3)$ are shown in Chapter 3, namely the $F - R$ type and F^2 type. They are categorized and classified into different classes. Fast calculating methods of these LI transforms are given.

2. Linearly Independent Ternary Arithmetic Transforms

The concept of fast LI transforms over $GF(3)$ is extended into standard ternary algebra. Correspondingly, $F - R$ and F^2 types of Linearly Independent Ternary Arithmetic (LITA) transforms are proposed in Chapter 4. Classification and properties, relations, computational complexities and experimental results of the LITA transforms are also presented.

3. Linearly Independent Transforms over Galois Field (4)

The extension of $F - R$ type fast LI transforms over Galois field (4) is shown in Chapter 5. These new transforms are known as Quaternary Linearly Independent (QLI) transforms. Hundreds of QLI transforms are derived from the generalized QLI transform matrix. Only those which possess fast transforms and have recursively defined matrices are classified in this thesis. Fast calculation algorithms apply to the QLI transforms.

4. Quaternary Linearly Independent Arithmetic Transforms

A novel family of Quaternary Linearly Independent Arithmetic (QLIA) transforms are presented in Chapter 6. It is the extension of QLI transforms, which is based on standard arithmetic transforms. Their properties give the novel transforms efficient method to extend to higher dimensions and also show their potential applications in representation of quaternary switching functions.

5. Fastest Linearly Independent Transforms

There exists a kind of LI transforms that possess the simplest matrix structure and the lowest computational cost, which are named as fastest LI transforms. The ternary and quaternary fastest LI transforms based on Galois field and standard algebra are proposed. The experimental results show their great advantage in calculation speed when compared with other known spectral transforms such as Reed-Muller and arithmetic transforms.

6. Helix Transforms over Galois Field and Standard Algebra

The helix transforms based on LI logic are proposed the first time by the author. They are named helix transforms due to their symmetrical structure along the diagonal in the transform matrices. These transforms possess some specific properties which give them advantages for the calculation of spectral coefficients. The butterfly diagrams of different helix transforms are also shown. Both ternary and quaternary cases over different fields and algebra are considered.

7. Ternary Logic Function Verification

Arithmetic transforms have been approved to be useful in logic circuit testify and function verification. In Chapter 4, the fastest LITA transforms are shown that they are useful in the application of ternary logic function verification. It can be expected that the quaternary fastest LIA transforms or other LITA/QLIA transforms have similar applications.

8. Implementation of Linearly Independent Transforms on Systolic Processor

It is shown in Chapter 7 that several types of ternary and quaternary LI transforms can be implemented on revised multiple-valued input and output systolic processors. The novel LI transforms also show their advantages in terms of requirement on the number of computing and storage cells in the structure. By using two joint processor structures, the implementation of helix transforms is also presented.

9. Logic Circuit Module of Quaternary Transforms

In Chapter 7, another application of LI transforms is also presented. The universal logic circuit module for the first stage QLI transforms is proposed. The presented universal logical module can be easily cascaded to realize higher level quaternary logic circuits for the reason that all the QLI transform matrices contain only two kinds of non-zero submatrices and are recursively defined. The approach can be extended to the implementation of all types ternary and quaternary LI transforms.

In conclusion, several new concepts of multiple-valued spectral transforms based on linearly independent logic that are useful for logic synthesis and representations of discrete switching functions are presented in this thesis. It derives new families of ternary and quaternary LI transforms on Galois field and standard algebra. Properties and relations of introduced transforms are investigated which give the transforms advantages in compact representations of discrete functions and in their calculation speed. Applications of the novel spectral transforms are also shown. More over, similar to know LI transforms for binary case, that can be used for the development of word-level decision diagrams, the new transforms can have the same applications as well. With the introduction of the novel LI transforms, it would open more choices in the design of multiple-valued logic circuit while taking into consideration of the tradeoff between efficiency of computation, final implementation and minimal number of non-zero coefficients. The advantage in terms of fast computation can be also used in other applications of introduced LI transforms such as finding symmetries in ternary and quaternary functions, image processing and CDMA coding where Reed-Muller and similar transforms have already been used.

8.2 Future work

A number of new open research problem arise from the developments presented in this thesis. Some of them are listed below.

1. Extending the concept of linearly independent logic into Galois field (5) and discovering the properties and mutual relations will be valuable in

efficient representation of multiple-valued logic functions in this area.

2. Application of different classes of multiple-valued LIA transforms developed in this thesis for efficient logic function verification and testing.
3. Development of universal logic circuit modules for various types of LI transforms.
4. Application of presented families of LI transforms in developing new word level decision diagrams and their comparison with the know ones.
5. Finding new helix transforms, which have specific matrix structure and properties, will be an interesting work.

Author's Publications

International Journals Publication

1. B. J. Falkowski and C. Fu, "Family of fast linearly independent ternary arithmetic transforms," *IEEE Trans. on Circuits and Systems I: Regular Papers*, vol. 51, no. 6, pp. 1132–1147, Jun. 2004.
2. B. J. Falkowski and C. Fu, "Family of ternary arithmetic polynomial expansions based on new classes of transforms," *Avtomatika i Telemekhanika, Journal of Russian Academy of Sciences, Institute of Transmission Control Problems, Maik Nauka/Interperiodica, Russia*, vol. 65, no. 6, pp. 22–36, Jun. 2004.
3. B. J. Falkowski and C. Fu, "Family of ternary arithmetic polynomial expansions based on new transforms," *Automation and Remote Control*, vol. 65, no. 6, pp. 857–870, Jun. 2004.
4. C. Fu and B. J. Falkowski, "Generation of multi-polarity helix transform over $GF(3)$," *IEICE Electronics Express*, vol. 1, pp. 211–216, Jul. 2004.
5. B. J. Falkowski and C. Fu, "Classification of new linearly independent transforms over $GF(3)$," *J. of Circuits, Systems, and Computers*, vol. 14, no. 2, pp. 395–421, Apr. 2005.
6. C. Fu and B. J. Falkowski, "Ternary fixed polarity linear Kronecker transforms and their comparison with ternary Reed-Muller transform," *J. of Circuits, Systems, and Computers*, vol. 14, no. 4, pp. 721–733, Aug. 2005.
7. B. J. Falkowski and C. Fu, "Fastest classes of linearly independent transforms over $GF(3)$ and their properties," *IEE Proc. Computers & Digital Techniques*, vol. 152, no. 5, pp. 567–576, Sep. 2005.

8. B. J. Falkowski and C. Fu, "Converting matrices between ternary helix and Reed-Muller transforms over Galois Field," *IEICE Electronics Express*, vol. 2, no. 21, pp. 530–535, Nov. 2005.
9. C. Fu and B. J. Falkowski, "Linearly independent ternary arithmetic helix transforms, their properties and relations," *IEE Proc. Vision, Image & Signal Processing*. Accepted for publication.
10. B. J. Falkowski and C. Fu, "Properties and experimental results of fastest linearly independent ternary arithmetic transforms," *IEEE Trans. on Circuits and Systems I: Regular Papers*. Accepted for publication.

International Conferences Publication

1. B. J. Falkowski and C. Fu, "Fastest linearly independent arithmetic transforms over $GF(3)$," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, vol. 2, (Hong Kong), pp. 669–672, Apr. 2003.
2. B. J. Falkowski and C. Fu, "Family of fast transforms over $GF(3)$ logic," in *Proc. 33rd IEEE Int. Symposium on Multiple-Valued Logic*, (Tokyo, Japan), pp. 323–328, May 2003.
3. B. J. Falkowski and C. Fu, "Polynomial expansions over $GF(3)$ based on fastest transformation," in *Proc. 33rd IEEE Int. Symposium on Multiple-Valued Logic*, (Tokyo, Japan), pp. 40–45, May 2003.
4. B. J. Falkowski and C. Fu, "Fast linearly independent ternary arithmetic transforms," in *Proc. 36th IEEE Int. Symposium on Circuits and Systems*, vol. 4, (Bangkok, Thailand), pp. 560–563, May 2003.
5. B. J. Falkowski and C. Fu, "Ternary arithmetic polynomial expansions based on new transforms," in *Proc. 36th IEEE Int. Symposium on Circuits and Systems*, vol. 4, (Bangkok, Thailand), pp. 564–567, May 2003.
6. B. J. Falkowski and C. Fu, "New expansions of linearly independent arithmetic transforms," in *Proc. European Conf. on Circuits Theory and Design*, vol. 1, (Krakow, Poland), pp. 361–369, Sep. 2003.

7. B. J. Falkowski and C. Fu, "Properties and experimental results of fastest linearly independent ternary arithmetic transforms," in *Proc. 13th Workshop on Post Binary Ultra-Large Scale Integration*, (Toronto, Canada), pp. 60–65, May 2004.
8. B. J. Falkowski and C. Fu, "Properties of fastest linearly independent transforms over $GF(3)$," in *Proc. 37th IEEE Int. Symposium on Circuits and Systems*, vol. 5, (Vancouver, Canada), pp. 285–288, May 2004.
9. C. Fu and B. Falkowski, "Multi-polarity helix transform over $GF(3)$," in *Proc. 37th IEEE Int. Symposium on Circuits and Systems*, vol. 5, (Vancouver, Canada), pp. 289–292, May 2004.
10. B. J. Falkowski and C. Fu, "Properties and relations of linearly independent ternary transforms over $GF(3)$," in *Proc. 4th TICSP Int. Workshop on Spectral Methods and Multirate Signal Processing*, (Vienna, Austria), pp. 107–110, Sep. 2004.
11. C. Fu and B. J. Falkowski, "New ternary fixed polarity linear Kronecker transforms and their comparison with ternary Reed-Muller transform," in *Proc. 4th TICSP Int. Workshop on Spectral Methods and Multirate Signal Processing*, (Vienna, Austria), pp. 111–114, Sep. 2004.
12. C. Fu and B. J. Falkowski, "Kronecker based fixed polarity transforms over $GF(3)$," in *Proc. IEEE Asia-Pacific Conf. on Circuits and Systems*, (Tainan, ROC), pp. 917–920, Dec. 2004.
13. C. Fu and B. J. Falkowski, "Linearly independent helix transforms over $GF(3)$," in *Proc. IEEE Asia-Pacific Conf. on Circuits and Systems*, (Tainan, ROC), pp. 921–924, Dec. 2004.
14. B. J. Falkowski and C. Fu, "Novel classes of linearly independent transforms for ternary logic functions," in *Proc. IEEE Asia-Pacific Conf. on Circuits and Systems*, (Tainan, ROC), pp. 1005–1008, Dec. 2004.
15. C. Fu and B. J. Falkowski, "Two classes of recursive fast transforms over $GF(3)$: properties, relations and computational costs," in *Proc. IEEE Asia-Pacific Conf. on Circuits and Systems*, (Tainan, ROC), pp. 1077–1080, Dec. 2004.
16. C. Fu and B. J. Falkowski, "Quaternary arithmetic helix transforms based on Kronecker product," in *Proc. 38th IEEE Int. Symposium on Circuits and Systems*, (Kobe, Japan), pp. 476–479, May 2005.

17. B. J. Falkowski and C. Fu, "Properties and relations of ternary linearly independent transforms," in *Proc. 38th IEEE Int. Symposium on Circuits and Systems*, (Kobe, Japan), pp. 4709–4712, May 2005.
18. B. J. Falkowski and C. Fu, "Generation of linearly independent transforms over $GF(4)$," in *Proc. 38th IEEE Int. Symposium on Circuits and Systems*, (Kobe, Japan), pp. 488–491, May 2005.
19. C. Fu and B. J. Falkowski, "Properties and relations of quaternary linearly independent helix transformations," in *Proc. 35th IEEE Int. Symposium on Multiple-Valued Logic*, (Calgary, Canada), pp. 200–205, May 2005.
20. B. J. Falkowski and C. Fu, "Classes of fastest quaternary linearly independent transformations," in *Proc. 35th IEEE Int. Symposium on Multiple-Valued Logic*, (Calgary, Canada), pp. 206–211, May 2005.
21. C. Fu and B. J. Falkowski, "Derivations of linearly independent ternary arithmetic helix transforms for higher dimensions," in *Proc. 48th IEEE Int. Midwest Symposium on Circuits and Systems*, (Cincinnati, Ohio), Aug. 2005.
22. B. J. Falkowski and C. Fu, "Fastest linearly independent arithmetic transforms and their applications in function verification," in *Proc. 48th IEEE Int. Midwest Symposium on Circuits and Systems*, (Cincinnati, Ohio), Aug. 2005.
23. B. J. Falkowski and C. Fu, "Mutual relations between helix and Reed-Muller transforms over $GF(3)$," in *Proc. 48th IEEE Int. Midwest Symposium on Circuits and Systems*, (Cincinnati, Ohio), Aug. 2005.

Bibliography

- [AAE95] S. Aghaian, J. T. Astola, and K. Egiazarian. *Binary Polynomial Transforms and Nonlinear Digital Filters*. Marcel Dekker, New York, 1995.
- [AAV00] L. N. Aizenberg, N. N. Aizenberg, and J. Vandewalle. *Multi-Valued and Universal Binary Neurons*. Kluwer Academic, Boston, 2000.
- [AIS95] V. M. Antonenko, A. A. Ivanov, and V. P. Shmerko. Linear arithmetic forms of k-valued logic functions and their implementation on systolic arrays. *Automation and Remote Control*, 56(3):419–432, 1995. (A translation of *Avtomatika i Telemekhanika*, 56(3):139–151, 1995).
- [AKS88] V. L. Artyukhov, V. N. Kondratev, and A. A. Shalyto. Generating Boolean functions via arithmetic polynomials. *Automation and Remote Control*, 49(4):508–551, 1988. (A translation of *Avtomatika i Telemekhanika*, 49(4):138–147, 1988).
- [Al-04] A. Al-Rabadi. *Reversible Logic Synthesis : From Fundamentals to Quantum Computing*. Springer-Verlag, New York, 2004.
- [Bea84] K. G. Beauchamp. *Applications of Walsh and Related Functions*. Academic Press, London, 1984.
- [Boo54] G. Boole. *The Laws of Thought*. MacMillan, London, 1854. (reprint by Dover, New York, 1958).
- [BP81] D. Bochmann and Ch. Posthoff. *Binäre dynamische Systeme*. Oldenbourg-Verlag, München; Akademie-Verlag, Berlin, 1981. (in German).
- [Bro90] F. Brown. *Boolean Reasoning*. Kluwer Academic, Boston, 1990.

- [Bry86] R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. on Computers*, 35(8):677–691, Aug. 1986.
- [Bry91] R. E. Bryant. On the complexity of vlsi implementations and graph representations of Boolean functions with application to integer multiplication. *IEEE Trans. on Computers*, 40(2):206–213, Feb. 1991.
- [BS91] D. Bochmann and B. Steinbach. *Logikentwurf mit XBOOLE, Algorithmen und Programme*. Verlag Technik, Berlin, 1991. (in German).
- [BS95] J. A. Brzozowski and C-J. Seger. *Asynchronous Circuits*. Springer-Verlag, Berlin, 1995.
- [CCH⁺99] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNelly, and L. Todd. *Surviving the SOC Revolution, A Guide to Platform-Based Design*. Kluwer Academic, Boston, 1999.
- [CFZ96] E. M. Clarke, M. Fujita, and X. Zhao. Multi-terminal binary decision diagrams and hybrid decision diagrams. In T. Sasao and M. Fujita, editors, *Representations of Discrete Functions*. Kluwer Academic, Boston, 1996.
- [CH82] X. Chen and S. L. Hurst. A comparison of universal-logic-module realizations and their application in the synthesis of combinatorial and sequential logic networks. *IEEE Trans. on Computers*, 31(2):140–147, Feb. 1982.
- [Che92] T. H. Chen. *Fault Diagnosis and Fault Tolerance: Systematic Approach to Special Topics*. Springer-Verlag, Berlin, 1992.
- [CMZ⁺93] E. M. Clarke, K. L. McMillan, X. Zhao, M. Fujita, and J. Yang. Spectral transforms for large Boolean functions with applications to technology mapping. In *Proc. 30th ACM/IEEE Conf. on Design Automation*, pages 54–60, Dallas, Texas, Jun. 1993.
- [CRC94] S. Chattopadhyay, S. Roy, and P. Pal Chaudhuri. KGPMAP: library based technology mapping technique for antifuse based FPGAs. *IEE Proc. on Computers and Digital Techniques*, 141(6):361–368, Nov. 1994.
- [DDT78] P. Davio, J. P. Deschamps, and A. Thayse. *Discrete and Switching Functions*. George and McGraw-Hill, New York, 1978.

- [DM94] G. De Micheli. *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, New York, 1994.
- [DMSY02] P. Dziurzanski, V. Malyugin, V. P. Shmerko, and S. N. Yanushkevich. Linear models of circuits based on the multivalued components. *Automation and Remote Control*, 63(6):960–980, 2002. (A translation of *Avtomatika i Telemekhanika*, 63(6):99–119, 2002).
- [Dre00] R. Drechsler. *Formal Verification of Circuits*. Kluwer Academic, Boston, 2000.
- [ER82] D. F. Elliott and K. R. Rao. *Fast Transforms, Algorithms, Analyses, Applications*. Academic Press, London, 1982.
- [Fal94] B. J. Falkowski. Properties and ways of calculation of multipolarity generalized Walsh transforms. *IEEE Trans. on Circuits and Systems II, Analog and Digital Signal Processing*, 41(6):380–391, Jun. 1994.
- [Fal99] B. J. Falkowski. A note on the polynomial form of Boolean functions and related topics. *IEEE Trans. on Computers*, 48(8):860–864, Aug. 1999.
- [Fal04] B. J. Falkowski. Compact representations of logic functions for lossless compression of Grey scale images. *IEE Proc. on Computers and Digital Techniques*, 151(3):221–230, May 2004.
- [FC97] B. J. Falkowski and C. H. Chang. Properties and methods of calculating generalized arithmetic and adding transforms. *IEE Proc. on Circuits, Devices, and Systems*, 144(5):249–258, Oct. 1997.
- [FC98] B. J. Falkowski and C. H. Chang. Mutual conversions between generalized arithmetic expansions and free binary decision diagrams. *IEE Proc. on Circuits, Devices, and Systems*, 145(4):219–228, Aug. 1998.
- [FF] B. J. Falkowski and C. Fu. Classification of new linearly independent transforms over GF(3). *J. of Circuits, Systems, and Computers*. Accepted for publication.
- [FF03] B. J. Falkowski and C. Fu. Family of fast transforms over GF(3) logic. In *Proc. 33rd IEEE Int. Symposium on Multiple-Valued Logic*, pages 323–328, Tokyo, Japan, May 2003.

- [FF04] C. Fu and B. J. Falkowski. Linearly independent helix transforms over $GF(3)$. In *Proc. IEEE Asia-Pacific Conf. on Circuits and Systems*, pages 921–924, Tainan, ROC, Dec. 2004.
- [FF05a] B. J. Falkowski and C. Fu. Classes of fastest quaternary linearly independent transformations. In *35th IEEE Int. Symposium on Multiple-Valued Logic*, Calgary, Canada, May 2005. Accepted for publication.
- [FF05b] B. J. Falkowski and C. Fu. Generation of linearly independent transforms over $GF(4)$. In *38th IEEE Int. Symposium on Circuits and Systems*, Kobe, Japan, May 2005. Accepted for publication.
- [FF05c] C. Fu and B. J. Falkowski. Properties and relations of quaternary linearly independent helix transformations. In *35th IEEE Int. Symposium on Multiple-Valued Logic*, Calgary, Canada, May 2005. Accepted for publication.
- [FL04] B. J. Falkowski and C. C. Lozano. Efficient calculation of fixed polarity Reed-Muller transform over $GF(4)$ using disjoint cubes. In *Proc. 10th Int. Symposium on IC Technology, Systems and Applications*, Singapore, Sep. 2004. CD publication.
- [FR96] B. J. Falkowski and S. Rahardja. Walsh-like functions and their relations. *IEE Proc. on Vision, Image, Signal Processing*, 143(5):279–284, Oct. 1996.
- [FR97] B. J. Falkowski and S. Rahardja. Classification and properties of fast linearly independent logic transformations. *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, 44(8):646–655, Aug. 1997.
- [FR00] B. J. Falkowski and S. Rahardja. Fast transforms for multiple-valued input binary output PLI logic. In *Proc. 30th IEEE Int. Symposium on Multiple-Valued Logic*, pages 47–52, Portland, Oregon, May 2000.
- [FR01] B. J. Falkowski and S. Rahardja. PLI logic for multiple-valued functions. *IEE Proc. on Computers and Digital Techniques*, 148(1):7–14, Jan. 2001.
- [FSP92] B. J. Falkowski, I. Schaefer, and M. A. Perkowski. Effective computer methods for the calculation of Rademacher-Walsh spectrum

- for completely and incompletely specified Boolean functions. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 11(10):1207–1226, Aug. 1992.
- [FSY97] B. J. Falkowski, V. P. Shmerko, and S. N. Yanushkevich. Arithmetical logic - its status and achievements. In *Proc. 4th Int. Conf. on Applications of Computer Systems*, pages 208–223, Szczecin, Poland, Nov. 1997.
- [GE78] D. H. Green and M. Edkins. Synthesis procedures for switching circuits represented in generalized Reed-Muller form over a finite field. *IEE Proc. on Computers and Digital Techniques*, 1(1):27–35, Jan. 1978.
- [GES87] B. Golubov, A. Efimov, and V. Skvortsov. *Walsh Series and Transforms, Theory and Applications*. Kluwer Academic, Boston, 1987.
- [Gra81] A. Graham. *Kronecker Products and Matrix Calculus with Applications*. John Wiley, New York, 1981.
- [Gre86] D. H. Green. *Modern Logic Design*. Addison-Wesley, Wokingham, 1986.
- [Gre89] D. H. Green. Ternary Reed-Muller switching functions with fixed and mixed polarities. *Int. Journal Electronics*, 67(5):761–775, Nov. 1989.
- [Gre90] D. H. Green. Reed-Muller expansions with fixed and mixed polarities over GF(4). *IEE Proc. on Computers and Digital Techniques*, 137(5):380–388, Sep. 1990.
- [GT74] D. H. Green and I. S. Taylor. Modular representation of multiple-valued logic systems. *IEE Proc. on Computers and Digital Techniques*, 121(2):409–418, Feb 1974.
- [GT76] D. H. Green and I. S. Taylor. Multiple-valued switching circuit design by means of generalized Reed-Muller expansions. *Digital Processes*, 2:63–81, 1976.
- [Hei91] K. D. Heidtmann. Arithmetic spectrum applied to fault detection for combinational networks. *IEEE Trans. on Computers*, 40(3):320–324, Mar. 1991.

- [HM92] B. Harking and C. Moraga. Efficient derivation of Reed-Muller expansions of Boolean functions. In *Proc. 22nd Int. Symposium on Multiple-Valued Logic*, pages 436–441, Sendai, Japan, May 1992.
- [HS96] G. D. Hachtel and F. Somenzi. *Logic Synthesis and Verification Algorithms*. Kluwer Academic, Boston, 1996.
- [HSB02] S. Hassoun, T. Sasao, and R. K. Brayton, editors. *Logic Synthesis and Verification*. Kluwer Academic, Boston, 2002.
- [Hur84] S. L. Hurst. Multiple-valued logic - its status and its future. *IEEE Trans. on Computers*, 33(12):1160–1179, 1984.
- [Hur92] S. L. Hurst. *Custom VLSI Microelectronics*. Prentice Hall, Hertfordshire, 1992.
- [ISM04] Y. Iguchi, T. Sasao, and M. Matsuura. A method to evaluate logic functions in the presence of unknown inputs using LUT cascades. In *Proc. 34th IEEE Int. Symposium on Multiple-Valued Logic*, pages pp. 302–308, Toronto, Canada, May 2004.
- [JSD02] D. Jankovic, R. S. Stankovic, and R. Drechsler. Efficient calculation of fixed-polarity polynomial expressions for multiple-valued logic functions. In *Proc. 32nd IEEE Int. Symposium on Multiple-Valued Logic*, pages 76–82, Boston, Massachusetts, May 2002.
- [Kar76] M. G. Karpovsky. *Finite Orthogonal Series in Design of Digital Devices*. Wiley, New York, 1976.
- [Kar85] M. G. Karpovsky, editor. *Spectral Techniques and Fault Detection*. Academic, Orlando, 1985.
- [KB81] S. K. Kumar and M. A. Breuer. Probabilistic aspects of Boolean switching functions via a new transform. *J. ACM*, 28:502–520, Jul. 1981.
- [KEA96] P. Kuosmanen, K. Egiazarian, and J. T. Astola. Boolean derivatives, weighted Chow parameters and selection of stack filters. *IEEE Trans. on Signal Processing*, 44(7):1634–1641, Jul. 1996.
- [Ker01] P. Kerntopf. New generalizations of Shannon decomposition. In *Proc. 10th Int. Workshop on Logic and Synthesis*, pages 109–118, Tahoe City, California, Jun. 2001.

- [KP03] P. Kerntopf and M. A. Perkowski. Function-driven linearly independent expansions of Boolean functions and their applications to synthesis of reversible circuits. In *Proc. 12th Int. Workshop on Logic and Synthesis*, Laguna Beach, California, May 2003.
- [KS75] K. L. Kodandapani and R. V. Setlur. Reed-Muller canonical forms in multivalued logic. *IEEE Trans. on Computers*, 24(6):628–636, Jun. 1975.
- [KS93] V. N. Kondratev and A. A. Shalito. Realization of systems of Boolean functions by linear arithmetic polynomials. *Automation and Remote Control*, 54(3):472–487, 1993. (A translation of *Avtomatika i Telemekhanika*, 54(3):135–151, 1993).
- [KSA00] M. G. Karpovsky, R. S. Stankovic, and J. T. Astola. Spectral techniques for design and testing of computer hardware. In *Proc. 1st Int. Workshop on Spectral Techniques and Logic Design for Future Digital Systems*, TICSP Series 10, pages 9–42, Tampere, Finland, Dec. 2000.
- [KSA03] M. G. Karpovsky, R. S. Stankovic, and J. T. Astola. Reduction of sizes of decision diagrams by autocorrelation functions. *IEEE Trans. on Computers*, 52(5):592–606, May 2003.
- [KSM01] M. G. Karpovsky, R. S. Stankovic, and C. Moraga. Spectral techniques in binary and multiple-valued switching theory. In *Proc. 31st IEEE Int. Symposium on Multiple-Valued Logic*, pages 22–24, Warsaw, Poland, May 2001.
- [KSY91] G. A. Kukharev, V. P. Shmerko, and S. N. Yanushkevich. *Technique of Binary Data Parallel Processing for VLSI*. University Press, Minsk, 1991. (in Russian).
- [KSZ90] G. A. Kukharev, V. P. Shmerko, and E. N. Zaitseva. *Multiple-Valued Data Processing Algorithms and Systolic Processors*. Science and Engineering, Minsk, 1990. (in Russian).
- [LM98] J. S. Lee and L. E. Miller. *CDMA Systems Engineering Handbook*. Artech House, Boston, 1998.
- [LPV94] Y. T. Lai, M. Pedram, and S. B. K. Vrudhula. EVBDD-based algorithms for integer linear programming, spectral transformation, and functional decomposition. *IEEE Trans. on Computer-Aided*

- Design of Integrated Circuits and Systems*, 13(8):959–975, Aug. 1994.
- [LV99] J. Lafferty and A. Vardy. Ordered binary decision diagrams and minimal trellises. *IEEE Trans. on Computers*, 48(9):971–986, Sep. 1999.
- [Mal97] V. D. Malyugin. *Paralleled Calculation by Means of Arithmetic Polynomials*. Physical and Mathematical Publishing, Russian Acad. Sci., Moscow, 1997. (in Russia).
- [MF97] R. Murgai and M. Fujita. Some recent advances in software and hardware logic simulation. In *Proc. 10th Int. Conf. on VLSI Design*, pages 232–238, Jan. 1997.
- [Mil87] D. M. Miller, editor. *Developments in Integrated Circuit Testing*. Academic, London, 1987.
- [MSS02] C. Moraga, T. Sasao, and R. S. Stankovic. A unifying approach to edge-valued and arithmetic transform decision diagrams. *Automation and Remote Control*, 63(1):125–138, 2002. (A translation of *Avtomatika i Telemekhanika*, 63(1):140–153, 2002).
- [MST00] C. Meinel, F. Somenzi, and T. Theobald. Linear sifting of decision diagrams and its application in synthesis. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 19(5):521–533, May 2000.
- [MT98] C. Meinel and T. Theobald. *Algorithms and Data Structures in VLSI Design: OBDD - Foundations and Applications*. Springer-Verlag, Berlin, 1998.
- [Muk86] M. Mukaidono. Regular ternary logic functions - ternary logic functions suitable for treating ambiguity. *IEEE Trans. on Computers*, 35(2):179–183, Feb. 1986.
- [MW86] J. C. Muzio and T. C. Wesselkamper. *Multiple-Valued Switching Theory*. Adam Hilger Ltd., Bristol and Boston, 1986.
- [Per92] M. A. Perkowski. The generalized orthonormal expansion of functions with multiple-valued inputs and some of its applications. In *Proc. 22nd IEEE Int. Symposium on Multiple-Valued Logic*, pages 442–450, Sendai, Japan, May 1992.

- [Per93] M. A. Perkowski. A fundamental theorem for EXOR circuits. In U. Kebschull, E. Schubert, and W. Rosenstiel, editors, *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller expansion in Circuit Design*, pages 52–60, Hamburg, Germany, Sep. 1993.
- [PFCD02] M. A. Perkowski, B. J. Falkowski, M. Chrzanowska-Jeske, and R. Drechsler. Efficient algorithms for creation of linearly-independent decision diagrams and their mapping to regular layouts. *VLSI Design*, 14(1):35–52, Feb. 2002.
- [PM75] K. P. Parker and E. J. McCluskey. Analysis of logic circuits with faults using input signal probabilities. *IEEE Trans. on Computers*, 23(5):573–578, May 1975.
- [PRG⁺95] M. A. Perkowski, T. Ross, D. Gadd, J. A. Goldman, and N. Song. Application of ESOP minimization in machine learning and knowledge discovery. In *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 288–299, Chiba, Japan, Sep. 1995.
- [PSB93] M. A. Perkowski, A. Sarabi, and F. R. Beyl. XOR canonical forms of switching functions. In U. Kebschull, E. Schubert, and W. Rosenstiel, editors, *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 27–32, Hamburg, Germany, Sep. 1993.
- [Red72] S.M. Reddy. Easily testable realizations for logic functions. *IEEE Trans. on Computers*, 21(11):1183–1188, Nov. 1972.
- [RF89] T. R. N. Rao and E. Fujiwara. *Error-Control Coding for Computer Systems*. Prentice Hall, Englewood Cliffs, 1989.
- [RF99a] S. Rahardja and B. J. Falkowski. Application of linearly independent arithmetic transform in testing of digital circuits. *IEE Electronics Letters*, 35(5):363–364, Mar. 1999.
- [RF99b] S. Rahardja and B. J. Falkowski. Fast linearly independent arithmetic expansions. *IEEE Trans. on Computers*, 48(9):991–999, Sep. 1999.
- [RK95] M. Ryu and M. Kameyama. Design of a highly parallel multiple-valued linear digital system for k-ary operations based on extended representation matrices. In *Proc. 25th IEEE Int. Symposium on*

- Multiple-Valued Logic*, pages 20–25, Bloomington, Indiana, May 1995.
- [RZ00] K. Radecka and Z. Zilic. Using arithmetic transform for verification of datapath circuits via error modeling. In *Proc. 18th IEEE VLSI Test Symposium*, pages 271–277, Montreal, Canada, 2000.
- [RZ02] K. Radecka and Z. Zilic. Specifying and verifying imprecise sequential datapaths by arithmetic transforms. In *Proc. 21st IEEE/ACM Int. Conf. on Computer-Aided Design*, pages 128–131, San Jose, California, Nov. 2002.
- [SA02] R. S. Stankovic and J. T. Astola. Some remarks on linear transform of variables in representation of adders by word-level expressions and spectral transform decision diagrams. In *Proc. 32nd IEEE Int. Symposium on Multiple-Valued Logic*, pages 116–122, Boston, Massachusetts, May 2002.
- [SA03] R. S. Stankovic and J. T. Astola. *Spectral Interpretation of Decision Diagram*. Springer-Verlag, New York, 2003.
- [Sas93] T. Sasao, editor. *Logic Synthesis and Optimization*. Kluwer Academic, Boston, 1993.
- [Sas99] T. Sasao. *Switching Theory for Logic Synthesis*. Kluwer Academic, Boston, 1999.
- [Sch89] W. G. Schneeweiss. *Boolean Functions with Engineering Applications and Computer Programs*. Springer-Verlag, Berlin, 1989.
- [SF96] T. Sasao and M. Fujita, editors. *Representations of Discrete Functions*. Kluwer Academic, Boston, 1996.
- [Sha38] C. E. Shannon. A symbolic analysis of relay and switching circuits. *Trans. of American Institute of Electrical Engineers*, 57:713–723, Mar. 1938.
- [Shm89] V. P. Shmerko. Synthesis of arithmetic forms of Boolean functions using the Fourier transform. *Automation and Remote Control*, 50(5):684–691, 1989. (A translation of *Avtomatika i Telemekhanika*, 50(5):134–142, 1989).

- [SM92] R. S. Stankovic and C. Moraga. Methods for detection of some properties of multiple-valued functions. *IEE Proc. on Computers and Digital Techniques*, 139(5):421–429, Sep. 1992.
- [SM94] V. P. Shmerko and S. V. Mikhailov. Review of publications in the former Soviet Union on spectral methods of logic data processing and logic differential calculus. In *Proc. 5th Int. Workshop on Spectral Techniques*, pages 48–54, Beijing, China, Mar. 1994.
- [Smi81] K. C. Smith. The prospects for multi-valued logic: a technology and application view. *IEEE Trans. on Computers*, 30(9):619–634, 1981.
- [SSJ98] R. S. Stankovic, M. S. Stankovic, and D. Jankovic. *Spectral Transforms in Switching Theory, Definitions and Calculations*. Science Publisher, Belgrade, 1998.
- [SSS96] R. S. Stankovic, M. R. Stojic, and M. S. Stankovic. *Recent Developments in Abstract Harmonic Analysis with Applications in Signal Processing*. Science Publisher, Belgrade, 1996.
- [Sta95] R. S. Stankovic. Some remarks about spectral transform interpretation of MTBDDs and EVBDDs. In *Proc. 1st IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 385–390, Makuhari, Japan, Aug. 1995.
- [Sta00] R. S. Stankovic. Arithmetic transform TDDs for generation of arithmetic expressions. In *1st Int. Workshop on Spectral Techniques and Logic Design for Future Digital Systems*, TICSP Series 10, pages 79–94, Tampere, Finland, Dec. 2000.
- [Sta01] R. S. Stankovic. Word-level ternary decision diagrams and arithmetic expansions. In *Proc. 5th Int. Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 34–50, Starkville, Mississippi, Aug. 2001.
- [TDM01] M. A. Thornton, R. Drechsler, and D. M. Miller. *Spectral Techniques in VLSI CAD*. Kluwer Academic, Boston, 2001.
- [TH79] V. H. Tokmen and S. L. Hurst. A consideration of universal-logic-modules for ternary synthesis based upon Reed-Muller coefficients. In *Proc. 9th IEEE Int. Symposium on Multiple-Valued Logic*, pages 248–256, Bath, United Kingdom, May 1979.

- [TM97] C. C. Tsai and M. Marek-Sadowska. Boolean functions classification via fixed polarity Reed-Muller forms. *IEEE Trans. on Computers*, 46(2):173–186, Feb. 1997.
- [Tos72] Z. Tosić. *Analytical Representations of m -Valued Logical Functions over the Ring of Integers Modulo m* . PhD thesis, University of Beograd Press, Beograd, 1972.
- [WFMW00] Y. L. Wu, H. B. Fan, M. Marek-Sadowska, and C. K. Wong. OBDD minimization based on two-level representation of Boolean functions. *IEEE Trans. on Computers*, 49(12):1371–1379, Dec. 2000.
- [Wol98] W. Wolf. *Modern VLSI Design, Systems on Silicon*. Prentice Hall, Upper Saddle River, 1998.
- [Yan94] S. N. Yanushkevich. Systolic algorithms to synthesize arithmetical polynomial forms for k -valued logic functions. *Automation and Remote Control*, 55(5):715–729, 1994. (A translation of *Avtomatika i Telemekhanika*, 55(12):128–141, 1994).
- [Yan98] S. N. Yanushkevich. *Logic Differential Calculus in Multi-Valued Logic Design*. Technical University of Szczecin Press, Szczecin, 1998. Scientific Works of Technical University of Szczecin, Nr. 537, Institute of Computer Science and Information Systems.
- [Yar85] L. P. Yaroslavsky. *Digital Picture Processing, An Introduction*. Springer-Verlag, Berlin, 1985.
- [YDS02] S. N. Yanushkevich, P. Dziurzanski, and V. P. Shmerko. The word-level models for efficient computation of multiple-valued functions, part 1: LAR. In *Proc. 32nd IEEE Int. Symposium on Multiple-Valued Logic*, pages 202–208, Boston, Massachusetts, May 2002.
- [YTS02] S. N. Yanushkevich, A. M. Tomaszewska, and V. P. Shmerko. The word-level models for efficient computation of multiple-valued functions, part 2: LWL. In *Proc. 32nd IEEE Int. Symposium on Multiple-Valued Logic*, pages 209–215, Boston, Massachusetts, May 2002.