

HARDWARE SECURITY

Design, Threats, and Safeguards

Debdeep Mukhopadhyay
Rajat Subhra Chakraborty



CRC Press
Taylor & Francis Group

A CHAPMAN & HALL BOOK

HARDWARE SECURITY

Design, Threats,
and Safeguards

HARDWARE SECURITY

Design, Threats,
and Safeguards

Debdeep Mukhopadhyay
Rajat Subhra Chakraborty

Indian Institute of Technology Kharagpur
West Bengal, India



CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

© 2015 by Taylor & Francis Group, LLC
CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works
Version Date: 20140908

International Standard Book Number-13: 978-1-4398-9584-9 (eBook - PDF)

This book contains information obtained from authentic and highly regarded sources. Reasonable efforts have been made to publish reliable data and information, but the author and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, please access www.copyright.com (<http://www.copyright.com/>) or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. CCC is a not-for-profit organization that provides licenses and registration for a variety of users. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation without intent to infringe.

Visit the Taylor & Francis Web site at
<http://www.taylorandfrancis.com>

and the CRC Press Web site at
<http://www.crcpress.com>

Contents

Foreword	xvii
Preface	xix
List of Figures	xxiii
List of Tables	xxxi
I Background	1
1 Mathematical Background	3
1.1 Introduction	3
1.2 Modular Arithmetic	4
1.3 Groups, Rings, and Fields	6
1.4 Greatest Common Divisors and Multiplicative Inverse	8
1.4.1 Euclidean Algorithm	9
1.4.2 Extended Euclidean Algorithm	10
1.4.3 Chinese Remainder Theorem	11
1.5 Subgroups, Subrings, and Extensions	13
1.6 Groups, Rings, and Field Isomorphisms	14
1.7 Polynomials and Fields	16
1.8 Construction of Galois Field	17
1.9 Extensions of Fields	18
1.10 Cyclic Groups of Group Elements	19
1.11 Efficient Galois Fields	21
1.11.1 Binary Finite Fields	21
1.12 Mapping between Binary and Composite Fields	24
1.12.1 Constructing Isomorphisms between Composite Fields	25
1.12.1.1 Mapping from $GF(2^k)$ to $GF(2^n)^m$, where $k = nm$	25
1.12.1.2 An Efficient Conversion Algorithm	26
1.13 Conclusions	28
2 Overview of Modern Cryptography	29
2.1 Introduction	30
2.2 Cryptography: Some Technical Details	31
2.3 Block Ciphers	36
2.3.1 Inner Structures of a Block Cipher	38
2.3.2 The Advanced Encryption Standard	39
2.3.3 The AES Round Transformations	40
2.3.3.1 SubBytes	41
2.3.3.2 ShiftRows	42
2.3.3.3 MixColumns	42

2.3.3.4	AddRoundKey	43
2.3.4	Key-Scheduling in AES	43
2.4	Rijndael in Composite Field	44
2.4.1	Expressing an Element of $GF(2^8)$ in Subfield	44
2.4.2	Inversion of an Element in Composite Field	45
2.4.3	The Round of AES in Composite Fields	46
2.5	Elliptic Curves	48
2.5.1	Simplification of the Weierstraß Equation	50
2.5.2	Singularity of Curves	51
2.5.3	The Abelian Group and the Group Laws	51
2.5.4	Elliptic Curves with Characteristic 2	52
2.5.5	Projective Coordinate Representation	57
2.6	Scalar Multiplications: LSB First and MSB First Approaches	58
2.7	Montgomery's Algorithm for Scalar Multiplication	59
2.7.1	Montgomery's Ladder	60
2.7.2	Faster Multiplication on EC without Pre-computations	60
2.7.3	Using Projective co-ordinates to Reduce the Number of Inversions .	62
2.8	Conclusions	62
3	Modern Hardware Design Practices	65
3.1	Introduction	65
3.1.1	FPGA Architecture	66
3.1.2	The FPGA Design Flow	68
3.2	Mapping an Algorithm to Hardware: Components of a Hardware Architecture	72
3.3	Case study: Binary gcd Processor	73
3.4	Enhancing the Performance of a Hardware Design	75
3.5	Modelling of the Computational Elements of the gcd Processor	78
3.5.1	Modeling of an Adder	78
3.5.2	Modeling of a Multiplexer	79
3.5.3	Total LUT Estimate of the gcd Processor	80
3.5.4	Delay Estimate of the gcd Processor	80
3.6	Experimental Results	80
3.6.1	LUT utilization of the gcd processor	81
3.6.2	Delay Estimates for the gcd Processor	81
3.7	Conclusions	81
II	Hardware Design of Cryptographic Algorithms	83
4	Hardware Design of the Advanced Encryption Standard (AES)	85
4.1	Introduction	85
4.2	Algorithmic and Architectural Optimizations for AES Design	86
4.3	Circuit for the AES S-Box	86
4.3.1	Subfield Mappings	87
4.3.2	Circuit Optimizations in the Polynomial Basis	87
4.3.3	Circuit Optimizations in the Normal Basis	91
4.3.4	Most Compact Realization: Polynomial vs. Normal basis	94
4.3.5	Common Subexpressions	96
4.3.6	Merging of Basis Change Matrix and Affine Mapping	97
4.3.7	Merged S-Box and Inverse S-Box	97
4.4	Implementation of the MixColumns Transformation	98
4.4.1	The AES S-Box and MixColumns	99

4.4.2	Implementing the Inverse MixColumns	100
4.5	An Example Reconfigurable Design for the Rijndael Cryptosystem	101
4.5.1	Overview of the Design	101
4.5.1.1	Encryption Unit	102
4.5.1.2	Pipelining and Serializing in FPGAs	102
4.5.1.3	Back to the Design	103
4.5.2	Key-Schedule Optimization	106
4.5.3	The Control Unit	107
4.6	Experimental Results	109
4.7	Single Chip Encryptor/Decryptor	110
4.8	Conclusions	111
5	Efficient Design of Finite Field Arithmetic on FPGAs	115
5.1	Introduction	116
5.2	Finite Field Multiplier	116
5.3	Finite Field Multipliers for High Performance Applications	117
5.4	Karatsuba Multiplication	118
5.5	Karatsuba Multipliers for Elliptic Curves	118
5.6	Designing for the FPGA Architecture	119
5.7	Analyzing Karatsuba Multipliers on FPGA Platforms	120
5.7.1	The Hybrid Karatsuba Multiplier	123
5.8	Performance Evaluation	126
5.9	High-Performance Finite Field Inversion Architecture for FPGAs	126
5.10	Itoh-Tsujii Inversion Algorithm	127
5.11	The Quad ITA Algorithm	128
5.11.1	Clock Cycles for the ITA	129
5.11.2	The Quad ITA Generalization	130
5.11.3	Hardware Architecture of Quad ITA	133
5.11.3.1	Addition Chain Selection Criteria	136
5.11.3.2	Design of the Quadblock	136
5.12	Experimental Results	137
5.13	Generalization of the ITA for 2^n Circuit	138
5.14	Hardware Architecture for 2^n Circuit Based ITA	140
5.15	Area and Delay Estimations for the 2^n ITA	141
5.15.1	Area and Delay of a x Variable Boolean Function	141
5.15.2	Area and LUT Delay Estimates for the Field Multiplier	142
5.15.3	Area and LUT Delay Estimates for the Reduction Circuit	145
5.15.4	Area and LUT Delay Estimates of a 2^n Circuit	146
5.15.5	Area and LUT Delay Estimates of a Multiplexer	146
5.15.6	Area and LUT Delay Estimates for the Powerblock	146
5.15.7	Area Estimate for the Entire ITA Architecture	147
5.15.8	LUT Delay Estimate for the Entire ITA Architecture	147
5.16	Obtaining the Optimal Performing ITA Architecture	148
5.17	Validation of Theoretical Estimations	149
5.18	Conclusions	151
6	High-Speed Implementation of Elliptic Curve Scalar Multiplication on FPGAs	153
6.1	Introduction	153
6.2	The Elliptic Curve Cryptoprocessor	155
6.2.1	Register Bank	155

6.2.2	Finite Field Arithmetic Unit	156
6.2.3	Control Unit	156
6.3	Point Arithmetic on the ECCP	157
6.3.1	Point Doubling	157
6.3.2	Point Addition	159
6.4	The Finite State Machine (FSM)	160
6.5	Performance Evaluation	163
6.6	Further Acceleration Techniques of the ECC Processor	163
6.7	Pipelining Strategies for the Scalar Multiplier	164
6.8	Scheduling of the Montgomery Algorithm	168
6.8.1	Scheduling for Two Consecutive Bits of the Scalar	169
6.8.2	Data Forwarding to Reduce Clock Cycles	171
6.9	Finding the Right Pipeline	171
6.9.1	Number of Clock Cycles	171
6.9.2	Analyzing Computation Time	172
6.10	Detailed Architecture of the <i>ECM</i>	172
6.11	Implementation Results	174
6.12	Conclusion	175

III Side Channel Analysis 177

7	Introduction to Side Channel Analysis	179
7.1	Introduction	179
7.2	What Are Side Channels?	180
7.2.1	Difference of Side Channel Analysis and Conventional Cryptanalysis	181
7.2.2	Types of Side Channel Attacks	182
7.3	Kocher's Seminal Works	182
7.3.1	Timing Attack on Modular Exponentiation	182
7.3.1.1	Timing Measurement	183
7.3.1.2	The Attack Methodology	183
7.4	Power Attacks	184
7.4.1	Simple Power Analysis	185
7.4.2	An Example SPA of the ECCP Designed	186
7.4.3	Differential Power Attacks	188
7.5	Fault Attacks	191
7.5.1	Fault Analysis of the RSA Cipher	193
7.6	Cache Attacks	193
7.6.1	Working Principle of Cache-Timing Attacks	194
7.7	Scan Chain Based Attacks	196
7.7.1	Working Principle of Scan-Chain-Attacks on Block Ciphers	196
7.8	Conclusions	199
8	Differential Fault Analysis of Ciphers 201	
8.1	Introduction to Differential Fault Analysis	202
8.1.1	General Principle of DFA of Block Ciphers	203
8.1.1.1	Fault Models	204
8.1.1.2	The Effect of Faults on a Block Cipher	204
8.2	DFA and Associated Fault Models	206
8.2.1	Fault Models for DFA of AES	206
8.2.1.1	Faults Are Injected in Any Location and Any Round	207
8.2.1.2	Faults Are Injected in AddRoundKey in Round 0	207

8.2.1.3	Faults Are Injected between the Output of 7 th and the Input of 8 th MixColumns	207
8.2.1.4	Faults Are Injected between the Output of 8 th and the Input of 9 th MixColumns	208
8.2.2	Relationships between the Discussed Fault Models	209
8.2.2.1	Faults Are Injected in Any Location and Any Round	209
8.2.2.2	Faults Are Injected in AddRoundKey in Round 0	209
8.2.2.3	Faults Are Injected between the Output of 7 th and the Input of 8 th MixColumns	209
8.2.2.4	Faults Are Injected between the Output of 8 th and the Input of 9 th MixColumns	210
8.3	Principle of Differential Fault Attacks on AES	211
8.3.1	Differential Properties of AES S-Box	211
8.3.2	DFA of AES Using Bit Faults	212
8.3.3	Bit-Level DFA of Last Round of AES	212
8.3.4	Bit-Level DFA of First Round of AES	213
8.4	State-of-the-art DFAs on AES	214
8.4.1	Byte-Level DFA of Penultimate Round of AES	214
8.4.1.1	DFA Using Two Faults	216
8.4.1.2	DFA Using Only One Fault	216
8.4.1.3	DFA with Reduced Time Complexity	220
8.5	Multiple-Byte DFA of AES-128	222
8.5.1	DFA According to Fault Model <i>DM0</i>	222
8.5.1.1	Equivalence of Faults in the Same Diagonal	222
8.5.2	DFA According to Fault Model <i>DM1</i>	224
8.5.3	DFA According to Fault Model <i>DM2</i>	224
8.6	Extension of the DFA to Other Variants of AES	225
8.6.1	DFA on AES-192 States	226
8.6.2	DFA on AES-256 States	226
8.6.2.1	First Phase of the Attack on AES-256 States	226
8.6.2.2	Second Phase of the Attack on AES-256 States	228
8.7	DFA of AES Targeting the Key Schedule	230
8.7.1	Attack on AES-128 Key Schedule	231
8.7.1.1	First Phase of the Attack on AES-128 Key Schedule	231
8.7.1.2	Second Phase of the Attack on AES-128 Key Schedule	234
8.7.1.3	Time Complexity Reduction	234
8.7.2	Proposed Attack on AES-192 Key Schedule	236
8.7.2.1	First Phase of the Attack on AES-192 Key Schedule	236
8.7.2.2	Second Phase of the Attack on AES-192 Key Schedule	239
8.7.3	Proposed Attack on AES-256 Key Schedule	240
8.7.3.1	First Phase of the Attack of AES-256 Key Schedule	241
8.7.3.2	Second Phase of th Attack of AES-256 Key Schedule	243
8.8	DFA countermeasures	244
8.8.1	Hardware Redundancy	246
8.8.2	Time Redundancy	247
8.8.3	Information Redundancy	248
8.8.3.1	Parity-1	248
8.8.3.2	Parity-16	249
8.8.3.3	Parity-32	250
8.8.3.4	Robust Code	250
8.8.4	Hybrid Redundancy	251

8.8.5 Other Techniques	254
8.9 Invariance based DFA Countermeasures	254
8.9.1 An Infective Countermeasure Scheme	254
8.9.2 Infection Countermeasure	255
8.9.3 Attacks on the Infection Countermeasure	256
8.9.3.1 Further Loop Holes in the Countermeasure: Attacking the Infection Technique	257
8.9.4 Infection Caused by Compulsory Dummy Round	257
8.9.4.1 Attacking the Top Row	258
8.9.5 Piret & Quisquater's Attack on the Countermeasure	259
8.10 Improved Countermeasure	262
8.11 Conclusions	264
9 Cache Attacks on Ciphers	265
9.1 Memory Hierarchy and Cache Memory	266
9.1.1 Organization of Cache Memory	267
9.1.2 Improving Cache Performance for Superscalar Processors	269
9.2 Timing Attacks Due to CPU Architecture	269
9.2.1 Attacks on Block Ciphers using Data Cache Memories	269
9.2.1.1 Table-based Implementation of the AES	269
9.2.1.2 Software Implementations of AES	271
9.3 Trace-Driven Cache Attacks	272
9.3.1 A Theoretical Cache Trace Attack on DES	273
9.3.2 Trace-Driven Attacks on AES	273
9.3.3 Trace-Driven Attacks on the Last Round of AES	274
9.3.4 Trace-Driven Attack Based on Induced Cache Miss	275
9.4 Access-Driven Cache Attacks	275
9.4.1 Access-Driven Attacks on Block Ciphers	276
9.4.2 Second Round Access-Driven Attack on AES	277
9.4.3 A Last Round Access Driven Attack on AES	278
9.4.4 Asynchronous Access-Driven Attacks	278
9.4.5 Fine-Grained Access-Driven Attacks on AES	279
9.5 Time-Driven Cache Attacks	280
9.5.1 Remote Timing Attacks	280
9.5.2 Distinguishing between a Hit and Miss with Time	281
9.5.3 Second Round Time-Driven Attack on AES	282
9.5.4 Theoretical Analysis of Time-Driven Attacks	283
9.5.5 Profiled Time-Driven Attack on AES	283
9.6 Countermeasures for Timing Attacks	286
9.6.1 Application-Layer Countermeasures	287
9.6.2 Countermeasures Applied in the Hardware	288
9.7 Conclusions	291
10 Power Analysis of Cipher Implementations	293
10.1 Power Attack Setup and Power Traces	294
10.1.1 Power Traces of an AES hardware	295
10.1.2 Quality of Power Measurements	297
10.2 Power Models	298
10.2.1 Hamming Weight Power Model	298
10.2.2 Hamming Distance Power Model	298
10.2.3 Why Do Gates Leak?	300

10.3	Differential Power Analysis Using Difference of Mean	301
10.3.1	Computing DoM for Simulated Power Traces on AES	302
10.3.2	Simulation of the Power Traces and Computing the DoM	302
10.3.3	DOM and the Hamming Distance vs. Hamming Weight Model	303
10.4	PKDPA: An Improvement of the DoM technique	304
10.4.1	Application of PKDPA to AES	304
10.4.2	Choosing the Window-Size	306
10.5	Correlation Power Attack	309
10.5.1	Computing Correlation Coefficient for Simulated Power Traces for AES	310
10.5.2	Computation of the Correlation Matrix	311
10.5.3	Experimental Results on Simulated Traces	312
10.6	Metrics to Evaluate a Side Channel Analysis	312
10.6.1	Success Rate of a Side Channel Adversary	312
10.6.2	Guessing Entropy of an Adversary	313
10.7	CPA on Real Power Traces of AES-128	315
10.7.1	SNR of a Power Trace	315
10.7.2	DPA of an Iterative Architecture	317
10.7.3	DPA on a Serialized Architecture of AES	318
10.7.4	Shuffling Architecture	318
10.8	Popular Countermeasures against Power Analysis: Masking	321
10.8.1	Masking at the Algorithmic Level	322
10.8.1.1	Masking the AES S-Box	322
10.8.1.2	Security of the Masking Scheme	324
10.8.1.3	Circuit Complexity and Reuse of Masks	325
10.8.2	Masking at the Gate Level	327
10.8.3	The Masked AND Gate and Vulnerabilities due to Glitches	328
10.9	Conclusions	330
11	Testability of Cryptographic Hardware	333
11.1	Introduction	333
11.2	Scan Chain Based Attacks on Cryptographic Implementations	334
11.2.1	Scan Chain Based Attack on Stream Ciphers	335
11.3	Scan Attack on Trivium	336
11.3.1	Brief Description of Trivium	336
11.3.1.1	Key and IV Setup	336
11.3.2	KeyStream Generation	337
11.3.3	The Attack	337
11.3.3.1	Objective of the Attacker	337
11.3.3.2	Attack on Trivium	337
11.3.3.3	Ascertaining Bit Correspondence	338
11.3.4	Deciphering the Cryptogram	339
11.4	Testability of Cryptographic Designs	341
11.4.0.1	Reset Attack on Flipped-Scan Mechanism	342

11.5 Conclusions	345
IV Hardware Intellectual Property Protection	347
12 Hardware Intellectual Property Protection through Obfuscation	349
12.1 Introduction	350
12.2 Related Work	352
12.3 Functional Obfuscation through State Transition Graph Modification	353
12.3.1 Goals of the Obfuscation Technique	353
12.3.2 Hardware IP Piracy: Adversary's Perspective	353
12.3.2.1 Modification Scheme Employing Input Logic-cone Expansion	354
12.3.3 Obfuscation Metric	355
12.3.4 System-level Obfuscation	356
12.3.4.1 State Transition Function Modification	356
12.3.5 Embedding Authentication Features	357
12.3.6 Choice of Optimal Set of Nodes for Modification	358
12.3.7 The <i>HARPOON</i> Design Methodology	359
12.4 Extension of STG Modification for RTL Designs	362
12.5 Obfuscation through Control and Dataflow Graph (CDFG) Modification	364
12.5.1 CDFG Obfuscation Methodology	364
12.5.1.1 Embedding Authentication Features	367
12.5.2 Comparison between the Two Approaches	367
12.5.3 Obfuscation-based Secure SoC Design Flow	368
12.6 Measure of Obfuscation Level	368
12.6.1 Manual Attacks by Visual Inspection	368
12.6.2 Simulation-based Attacks	369
12.6.3 Structural Analysis-based Attack	369
12.6.3.1 Structural Analysis against STG Modification	369
12.6.3.2 Structural Analysis against CDFG Modification-based Approach	371
12.6.4 Quantitative Comparison	371
12.7 Results	372
12.7.1 Design Flow Automation	372
12.7.2 Implementation Results	374
12.7.2.1 Implementation Setup	374
12.7.2.2 Obfuscation Efficiency and Overheads	374
12.7.3 Effect of Key Length	375
12.7.3.1 Support for Multiple-length Keys	375
12.7.3.2 Effect of Increasing Key Length	376
12.8 Discussions	377
12.8.1 Using Unreachable States during Initialization	377
12.9 Conclusions	378
V Hardware Trojans	379
13 Overview of Hardware Trojans	381
13.1 Introduction	382
13.2 Trojan Taxonomy and Examples	383
13.3 Multi-level Attacks	386

13.3.1	Difficulty of Recovering Encryption Key from Modified Faulty Cipher-text	389
13.3.2	Effectiveness of Multi-level Attacks	389
13.3.3	Results: Multi-level Attacks	390
13.3.4	Extension of the Multi-level Attack Model	391
13.3.5	Prevention Techniques against Multi-level Attacks	391
13.4	Effect of Hardware Trojan on Circuit Reliability	392
13.4.1	Hardware Trojan and Reliability: Analysis and Results	394
13.4.1.1	Combinational Trojans	394
13.4.2	Synchronous Counter-based Sequential Trojan	395
13.4.3	Asynchronous Counter-based Sequential Trojan	397
13.5	Hardware Trojan Insertion by Direct Modification of FPGA Configuration Bitstream	398
13.5.1	Xilinx Configuration Bitstream Structure	400
13.5.2	Xilinx Virtex-II FPGA Building Blocks	400
13.5.3	Xilinx Configuration Bitstream File Organisation	400
13.5.3.1	Configuration Memory Addressing	400
13.5.4	Configuration Process	401
13.5.5	Cyclic Redundancy Check (CRC)	401
13.5.6	Disabling CRC Check	401
13.5.7	Bitstream Modification Technique	401
13.5.8	Demonstration of the Bitstream Modification-based Attack	404
13.5.9	Prevention/Detection Techniques against the Proposed Attack	406
13.5.9.1	Grounding Unused Pins in a FPGA	406
13.5.9.2	Online Temperature Monitoring	406
13.5.9.3	Filling-up Unused Resources of the FPGA	406
13.5.10	Logic Distribution to Balance Power Consumption from the I/O Power Pins	406
13.5.10.1	Dedicated Hardware Logic to Check CRC Status	406
13.5.10.2	Scrambling and De-scrambling the Bitstream File	406
13.6	Conclusion	407
14	Logic Testing based Hardware Trojan Detection	409
14.1	Introduction	409
14.2	Statistical Approach for Trojan Detection	410
14.2.1	Mathematical Analysis	411
14.2.2	Test Generation	412
14.2.3	Coverage Estimation	413
14.2.4	Choice of Trojan Sample Size	413
14.2.5	Choice of N	413
14.2.6	Improving Trojan Detection Coverage	413
14.3	Results	415
14.3.1	Simulation Setup	415
14.3.2	Comparison with Random and ATPG Patterns	416
14.3.3	Effect of Number of Trigger Points (q)	417
14.3.4	Effect of Trigger Threshold (θ)	418
14.3.5	Sequential Trojan Detection	418
14.3.6	Application to Side-channel Analysis	420
14.4	Conclusions	420

15 Side-channel Analysis Techniques for Hardware Trojans Detection	421
15.1 Introduction	422
15.2 Motivation for the Proposed Approaches	423
15.3 Multiple-parameter Analysis based Trojan Detection	424
15.3.1 Theoretical Analysis	425
15.3.2 Improving Detection Sensitivity	428
15.3.2.1 Test Vector Selection	428
15.3.2.2 Power Gating and Operand Isolation	430
15.3.2.3 Use of Other Side-channel Parameters	431
15.3.2.4 Test Conditions	432
15.4 Results	433
15.4.1 Simulation Verification	433
15.4.1.1 Test Setup	433
15.4.1.2 Results	434
15.4.2 Hardware Validation	436
15.4.2.1 Test Setup	436
15.4.2.2 Results	438
15.5 Integration with Logic-testing Approach	439
15.5.1 Motivation of Self-Referencing Approach	441
15.5.2 Analysis of Process Variations on Self-Referencing	443
15.5.3 Self-Referencing Methodology	444
15.5.3.1 Functional Decomposition	445
15.5.3.2 Statistical Test Vector Generation	446
15.5.3.3 Side-channel Analysis Using Self-Referencing	447
15.5.3.4 Decision Making Process	448
15.5.4 Self-referencing Simulation Results	449
15.5.5 Experimental Results	450
15.5.6 Conclusions	451
16 Design Techniques for Hardware Trojan Threat Mitigation	453
16.1 Introduction	453
16.2 Obfuscation-based Trojan Detection/Protection	454
16.2.1 Methodology of Circuit Obfuscation-based Trojan Detection / Protection	454
16.2.2 Effect of Obfuscation on Trojan Insertion	456
16.2.3 Effect of Obfuscation on Trojan Potency	457
16.2.4 Effect of Obfuscation on Trojan Detectability	458
16.2.5 Effect of Obfuscation on Circuit Structure	459
16.2.6 Determination of Unreachable States	460
16.2.7 Test Generation for Trojan Detection	460
16.2.8 Determination of Effectiveness	461
16.3 Integrated Framework for Obfuscation	462
16.4 Results	464
16.4.1 Discussions	466
16.4.2 Application to Third-party IP Modules and SoCs	466
16.4.3 Protection against Malicious CAD Tools	466
16.4.4 Improving Level of Protection and Design Overhead	469
16.4.5 Protection against Information Leakage Trojans	469
16.5 A FPGA-based Design Technique for Trojan Isolation	470
16.6 A Design Infrastructure Approach to Prevent Circuit Malfunction	471

VI Physically Unclonable Functions	473
17 Physically Unclonable Functions: a Root-of-Trust for Hardware Security	475
17.1 Introduction	476
17.2 Physically Unclonable Function	476
17.3 Classification of PUFs	478
17.3.1 Classification based on Technology	478
17.3.2 Classification based on Response Collection Mechanism	479
17.3.3 Classification based on Difficulty of Modeling	480
17.3.4 Classification based on Reconfiguration Technique	480
17.4 Realization of Silicon PUFs	481
17.4.1 Delay-based PUF	481
17.4.1.1 Ring Oscillator PUF	481
17.4.1.2 Arbiter PUF	482
17.4.1.3 Bistable Ring PUF	483
17.4.1.4 Loop PUF	483
17.4.2 Memory-based PUF	484
17.4.2.1 SRAM PUF	484
17.4.2.2 Butterfly PUF	485
17.4.2.3 Latch PUF	485
17.4.2.4 Flip-flop PUF	485
17.5 PUF Performance Metrics for Quality Evaluation	486
17.5.1 Uniqueness	486
17.5.2 Reliability	487
17.5.3 Uniformity	488
17.5.4 Bit-aliasing	488
17.5.5 Bit-dependency	488
17.6 Secure PUF: What Makes a PUF Secure?	488
17.7 Applications of PUF as a Root-of-Trust	489
17.7.1 Low-cost Authentication	489
17.7.2 Cryptographic Key Generation	489
17.7.3 IP protection of FPGA Hardware Designs	491
17.7.4 Key Zeroization	491
17.8 Attacks Model: How PUF Security Could Be Compromised?	491
17.8.1 Fault Attacks on PUFs and Fuzzy Extractors	492
17.8.2 Side-channel Attacks on PUFs	492
17.8.3 Modeling Attacks on Delay-based PUFs	492
17.9 Looking Forward: What Lies Ahead for PUFs?	493
18 Genetic Programming-based Model-building Attack on PUFs	495
18.1 Introduction	495
18.2 Background: Genetic Programming and RO-PUFs	497
18.2.1 Genetic Programming	497
18.2.2 Ring Oscillator PUF (RO-PUF)	498
18.3 Methodology	499
18.3.1 Reproduction Schemes	500
18.3.1.1 Elitist Model	500
18.3.1.2 Tournament Selection Model	500
18.4 Results	500
18.4.1 Experimental Setup	500
18.4.2 Conclusions	502

Bibliography	505
Index	535

Foreword

In the past decade, the field of hardware security has grown into a major research topic, attracting intense interest from academics, industry, and governments alike. Indeed, electronic information technology now controls, documents, and supports virtually every aspect of our life. Be it electronic money, ID-cards, car electronics, industrial controllers, or Internet routers, this world is governed by bits as much as it is governed by physical resources. Hardware security addresses multiple key requirements in this information landscape: the need to securely handle and store electronic information any time, anywhere; and the need to do it very efficiently in terms of resource cost and energy cost.

Hardware security collects a rich field of research which combines multiple knowledge domains including, among others, discrete math, algorithm design and transformations, digital architecture design with analog twists, and controlled production technologies. Hardware security puts traditional notions of hardware design on their head. Examples are the beneficial use of noise to produce random numbers and to feed security protocols, and the design of hardware that has a constant, rather than minimal, power dissipation, to avoid side-channel leakage.

The origins of hardware security lie within cryptographic engineering, a field young enough to have new artifacts and methods still carry the inventor's last name. Indeed, although cryptography has been around for centuries, it's only since the last 50 years that it has established itself as a field of science that enables participation of a broad community. Cryptography is now a fundamental part of information technology. Cryptographic Engineering is concerned with design and implementation aspects of cryptography. It is a vibrant area with established conference venues such as CHES attracting hundreds of participants. New ideas in cryptography, such as privacy-friendly subscriptions and multi-party computation, provide a continuous influx of new design challenges to the hardware security engineer.

Besides a continuous innovation at the level of applications, secure hardware has also benefited, and is still benefiting, from tremendous improvements in technology. Although the impeding end-of-the-road for traditional CMOS has been predicted many years, the technology has propelled secure hardware into applications that would have been unthinkable just a decade ago. In fact, the 'Things' in the Internet of Things would be impossible without secure hardware, and this secure hardware would be impossible without the dense integration and high efficiencies offered by technology.

A fourth factor that makes secure hardware an exciting field of research is in its role as last-line-of-defense. Traditional software-based cyber-security has been plagued by security issues for many years, and hardware has been touted as a safe haven for our private electronic information. However, secure hardware is under attack, as well. Adversaries have learned to pick up side-channel information from secure processing using low-cost, simple measurement equipment; they have learned how faults can reveal the inner workings of algorithms that are conceived as a black box by cryptographers. Adversaries have even learned to interfere with the IC design flow itself, subverting the design with a Trojan and making the IC do something different than it was conceived for.

It is with great excitement that I see this book on 'Hardware Security Design' published.

In recent years, many schools have started graduate courses on this topic, often relying on advanced research papers. There is a clear need for a general, comprehensive text that can be used as the basis for a course. Furthermore, there is a great need for a reference text that will help designers and practitioners in the field to understand the main issues. This text stands out among others in that it has been written by a team of just two authors. They have not only done an admirable effort, but they have also ensured a consistent presentation and discussion.

The authors have done an excellent job at systematically introducing the field of hardware security. They start with the fundamentals, touch upon practical aspects of hardware design, discuss the mapping of symmetric-key and public-key algorithms, and finally provide an in-depth treatment of the last-line-of-defense aspects of secure hardware. The book is supported by an extensive collection of references, testifying to the intensive level of research ongoing in this exciting, innovative field. I hope the book will encourage new researchers and practitioners to join the effort in building the fundamentals of secure information technology that our future calls for.

Patrick Schaumont
Blacksburg, August 2014

Preface

With the ever-increasing proliferation of e-business practices, great volumes of secure business transactions and data transmissions are routinely carried out in an encrypted form in devices ranging in scale from personal smartcards to business servers. These cryptographic algorithms are often computationally intensive and are designed in hardware and embedded systems to meet the real-time requirements. Developing expertise in this field of *hardware security* requires background in a wide range of topics. On one end is understanding of the mathematical principles behind the cryptographic algorithms and the underlying theory behind their working. These principles are necessary to innovate in the process of developing efficient implementations, which is central to this discourse. This also needs to be backed by exposure in the field of Very Large Scale Integration (VLSI) and embedded systems. Understanding the platforms on which the designs are developed is needed to develop high performance and compact solutions to the cryptographic algorithms. On the other end are the threats which arise from the hardware implementations. History has taught us that strong cryptographic algorithms and their efficient designs are just the beginning. Naïve optimizations to develop efficient hardware devices can lead to embarrassing attacks which can have catastrophic implications. Often these attacks are based on exploitation of *side-channels*, which are covert channels leaking information which the designers of the cryptographic algorithms or the conventional hardware engineer will be unaware of. Further, the complexity of hardware designs also makes it increasingly difficult to detect bugs and modifications. These modifications can be malicious, as they can be the seat of *hardware Trojans*, which when triggered can lead to disastrous system failures and/or security breaches. Another pressing issue in the world of cyber-security arises from the threats of counterfeit integrated circuits (ICs). Detecting and protecting against these vulnerabilities requires “unclonable” novel hardware security primitives, which can act as *fingerprint generators* for the manufactured IC instances.

This book thus attempts to bring on a single platform a treatment of all these aspects of hardware security which we believe are quite challenging and unique. The book is targeted for a senior undergraduate or post-graduate course on hardware security. The book is suitable for students from not only CS and EE backgrounds, but also from mathematics. The book is also suitable for self-study of the practising professional who requires an exposure to state-of-the-art research on hardware security. Although we have strived to provide a contemporary overview on the design, threats and safeguards required for assurance in hardware security, we believe that the work, because of its constant evolution, will always be dynamic. However having said that, the fundamentals, which we have attempted to bring out, should stand out in the mind of the reader, assisting in future developments and research.

As mentioned, the content of the book covers modern day hardware security issues, along with fundamental aspects which are imperative for a comprehensive understanding. We briefly describe these aspects:

- **Mathematical Background and Cryptographic Algorithms:** Modern-day cryptographic systems rely heavily on field theory. Understanding of field arithmetic techniques, definitions, constructions, and inter-relations of several fields are required to

develop efficient crypto-designs. Modern ciphers come in wide-ranging variety, from symmetric algorithms like *Advanced Encryption Standard* (AES), to asymmetric algorithms like *Elliptic Curve Cryptography* (ECC). As we will see in the book, the internal structures of the algorithms and suitable manipulations can lead to facts which can be exploited in efficient implementations.

- **Modern Hardware Design:** A designer has to develop a thorough understanding of the hardware platforms on which the implementations are to be developed. In this book we focus on Field Programmable Gate Arrays (FPGAs), which are fast, programmable and low cost. With the rapid development of FPGA technology, they have become high-performance, and are suitable test beds for cryptographic engineering research and development. With their in-house nature they provide excellent test beds for learning the trade secrets of hardware security, while also providing avenues for developing complete end products. While depending on various algorithms the architecture of the designs vary, a systematic approach can lead to a suitable design with proper separation of the control and data-path. The Look-up-Table (LUT)-based structure of an FPGA also allows suitable modeling of the critical path and LUT, which enables the exploration of the design alternatives.
- **Hardware Design of Cryptographic Algorithms and Arithmetic Units in FPGAs:** It is a challenging task to transform the cryptographic algorithms into efficient implementations on a hardware platform. Further, FPGAs because of their underlying Lookup Table (LUT)-based structures provide new design challenges, which the architecture should be capable of leveraging. In these parts of the book, we study several algorithmic and architectural optimizations for arithmetic circuits useful in the cryptographic domain, like field multipliers, inverses and also for various ciphers, like AES and ECC.
- **Side Channel Analysis:** As discussed, naïve implementations and optimizations can lead to vulnerabilities in crypto-implementations through several information leakage channels, which are commonly referred to as *side-channels*. The book first attempts to present an overview on some of the common side channels and their usage in attacking ciphers. Side channels range from timing, power, behavior under faults, performance on platforms enabled with cache memories, and also Design-for-Testability (DFT) features. Subsequently, we deal in depth on each of the side-channels to understand the working principles of the attacks. How to protect against these attacks is a related question, which we also address with respect to each of the attacks.
- **Hardware Intellectual Property Protection:** Modern electronic design regularly employs high-performance, pre-verified hardware *intellectual property* (IP) cores to meet aggressive time-to-market demands. While this practice improves design quality and decreases design cost and time, this also exposes the IP vendor to IP piracy, thereby causing large loss of revenue. We develop a systematic design approach for hardware IP protection, based on the principle of *design obfuscation*. This technique is applicable for hardware IPs at both the *gate-level* and *Register Transfer Level* (RTL).
- **Hardware Trojans:** Modern semiconductor manufacturing follows a “horizontal” business model, where several external agencies are involved in the design and manufacturing phase. The resources and services procured from external agents include hardware IP cores (mentioned previously), *computer-aided design* (CAD) software tools, *standard cell libraries*, fabrication and testing services, etc. However, such practices decrease the amount of control that the design house used to traditionally exercise over their own products, thus making them vulnerable to malicious, hard-to-detect,

design modifications, commonly termed as *Hardware Trojan Horses* (“Hardware Trojans” in short). If undetected, these hardware Trojans can cause irreparable loss to critical infrastructure, human life and property. Traditional post-manufacturing testing methods are often unable to detect hardware Trojans. We explore the unique threat posed by hardware Trojans, and novel logic and side-channel analysis-based testing techniques to detect them. We also explore a design obfuscation based technique to resist hardware Trojan insertion, and to make inserted hardware Trojans more detectable.

- **Physically Unclonable Functions:** The threat posed by counterfeit and cloned ICs has become a major global issue, and is a consequence of the practice of remote fabrication of ICs. Silicon *Physically Unclonable Functions* (PUFs), proposed in recent years, constitute a class of hardware security primitives, which hold great promise in solving the problem of IC authentication. However, design of silicon PUFs comes with its unique set of challenges. It has also been shown that PUFs are vulnerable to the so-called *model-building attacks*. We explore PUF designs, metrics to judge the quality of a designed PUF, and state-of-the-art model building attacks on PUFs.

Acknowledgements: This preface would be incomplete without a mention of the people who made this effort possible. The authors would like to express their sincere gratitude to the Computer Science and Engineering Department of IIT Kharagpur, which provides an ideal environment for research on hardware security. The students of the *Secure Embedded Architecture Laboratory* (SEAL) in the CSE department of IIT Kharagpur read initial versions of the draft, and gave constructive criticisms. The authors appreciate the staff at CRC Press, especially Ms. Aastha Sharma, for their encouragement and understanding, in spite of missing several deadlines.

Debdeep Mukhopadhyay would like to thank all his former and present students for serving as a motivation and also window to this evolving subject. He would like also to thank all his colleagues, collaborators and funding agencies without whose support this work would not have been possible. Special thanks to Dr Chester Rebeiro (Columbia University), Dr Subidh Ali (NYU-Abu Dhabi), Sujoy Sinha Roy (K.U. Leuven), Dr Xiaofei (Rex) Guo (NYU-Poly), Dr Bodhisatwa Mazumdar (NYU-Abu Dhabi), Suvadeep Hajra (IIT Kharagpur), Prof Abhijit Das (IIT Kharagpur), Prof Dipanwita Roy Chowdhury (IIT Kharagpur), Prof Indranil Sengupta (IIT Kharagpur), Prof Partha Pratim Chakrabarti (IIT Kharagpur), Prof Ramesh Karri (NYU-Poly), Prof Ingrid Verbauwhede (K.U.Leuven), Prof C. Pandurangan (IIT Madras), Prof V. Kamakoti (IIT Madras), Dr Sanjay Burman (CAIR, DRDO), Dr P. K. Saxena (SAG, DRDO), Prof Bimal Roy (ISI Kolkata), Prof Sylvain Guille (Telecom Paris Tech), Dr Junko Takahashi (NTT Labs), Dr Toshinori Fukunaga (NTT Labs), for their contributions and encouragement in various stages of the book. He would like to say a thanks to his brother Rajdeep for being a constant companion and for emphasizing the importance of working with choice. He would like to take this occasion to express gratitude to family members for their constant support, prayers and encouragement. He would like to specially mention his father-in-law, Anup Chatterjee for his constant enquiry on the status of the book, which helped to get back to the book after the occasional gaps. This book would not have been possible without the *silent* support of his wife Ayantika through several highs and lows in life, and most importantly to help him dream big. Her help to provide a home after a tiring day to collect himself for a new challenge has been instrumental in this long journey. He would also like to take this moment to remember his Grand Mother, who passed away in the final moments of compiling this text, and pray for her soul’s eternal peace. Finally, he would like to thank *God* for being kind to gift with a lovely daughter, *Debanti* in the final phases of writing the book, to inspire him for his future

endeavours. Debdeep would like to dedicate the book to his parents, Niharendu and Dipa Mukhopadhyay, without whom nothing would be possible.

Rajat Subhra Chakraborty would like to thank his family members, especially his wife Munmun, for their love, patience and understanding, and for allowing him to concentrate on the book during the weekends and vacations. Rajat would like to dedicate this book to the memory of his father, Pratyush Kumar Chakraborty, who passed away while this book was being written. Rest in peace, *Baba*.

List of Figures

1.1	Homomorphism	15
1.2	Squaring Circuit	22
1.3	Modular Reduction with Trinomial $x^{233} + x^{74} + 1$	23
2.1	Secret Key Cryptosystem Model	32
2.2	The Diffie Hellman Key Exchange	35
2.3	Block Cipher: Encryption	37
2.4	Electronic Code Book	37
2.5	Cipher Block Chaining	37
2.6	Structure of a Block Cipher	38
2.7	SubBytes Transformation	41
2.8	MixColumn Transformation	43
2.9	Plot of y vs x	48
2.10	Point Addition	54
2.11	Point Doubling	54
3.1	FPGA island-style architecture	67
3.2	FPGA Logic Block	67
3.3	False path in a circuit	70
3.4	Multi-cycle path in a circuit	70
3.5	The FPGA Design Flow	71
3.6	Important Components of an Architecture	72
3.7	Data Path of a gcd Processor	76
3.8	LUTs in Critical Path vs. Delay for a Combinational Multiplier	78
3.9	Adder with Carry Chain	79
3.10	LUT utilization of the gcd processor (both theoretical and actual)	81
3.11	Delay Modeling of the gcd processor (both theoretical and actual)	82
4.1	Polynomial $GF(2^8)$ Inverter	88
4.2	Polynomial $GF(2^4)$ Inverter	89
4.3	Normal $GF(2^8)$ Inverter	92
4.4	Normal $GF(2^4)$ Inverter	93
4.5	Polynomial $GF(2^4)$ Multiplier Sharing a Common Input	96
4.6	MixColumns and Key XORing Combined to Save LUTs	99
4.7	The MixColumns architecture in a composite field	100
4.8	Top-Level Architecture [25]	102
4.9	Pipelining of the Encryption Unit	103
4.10	The Encryption Unit	103
4.11	Structure of Different S-Boxes	104
4.12	Structure of Inversion Calculation	105
4.13	Key-Scheduler	106
4.14	FSMD of Control Unit	108

4.15	Different Data-Flow of (a) Encryption and (b) Decryption Datapath	111
4.16	(a) Architecture of Encryption Datapath, (b) Architecture of Decryption Datapath	112
4.17	Single Chip Encryptor/Decryptor Core Design	113
5.1	Elliptic Curve Pyramid	117
5.2	Combining the Partial Products in a Karatsuba Multiplier	120
5.3	233-Bit Hybrid Karatsuba Multiplier [315]	124
5.4	m -Bit Multiplication vs. Area \times Time [315]	125
5.5	Circuit to Raise the Input to the Power of 2^k	129
5.6	Quad-ITA Architecture for $GF(2^{233})$ with the Addition Chain 5.24	134
5.7	Quadblock Design: Raises the Input to the Power of 4^k	134
5.8	Clock Cycles of Computation Time versus Number of Quads in Quadblock on a Xilinx Virtex 4 FPGA for $GF(2^{233})$	137
5.9	Performance of Quad-ITA vs Squarer-ITA Implementation for Different Fields on a Xilinx Virtex 4 FPGA	138
5.10	FPGA Architecture for 2^n ITA	140
5.11	Powerblock Design with Cascaded 2^n Circuits	140
5.12	LUTs in Critical Path vs Delay for a Combinational Multiplier	142
5.13	Three Stages of a Combinational Hybrid Karatsuba Multiplier	143
5.14	Generation of Overlapped Operands During Recursive Splitting of Inputs .	145
5.15	Threshold vs LUT Requirements for $GF(2^{163})$ Multiplier	145
5.16	Number of Clock Cycles vs. n	149
5.17	Performance Variation with n in 4-LUT-based FPGA	150
5.18	Performance Variation with n in 6-LUT-based FPGA	150
6.1	Block Diagram of the Elliptic Curve Crypto Processor	154
6.2	Register File for Elliptic Curve Crypto Processor	155
6.3	Finite Field Arithmetic Unit	157
6.4	The ECCP Finite State Machine	160
6.5	Block Diagram of the Processor Organization showing the critical path .	165
6.6	Different Stages in the HBKM	166
6.7	Example Critical Path (with $L = 4$ and $k = 4$)	167
6.8	Scheduling the Scalar Bit s_k	169
6.9	Schedule for Two Scalar Bits when $s_{k-1} = s_k$	170
6.10	Schedule for Two Scalar Bits when $s_{k-1} \neq s_k$	170
6.11	Effect of Data Forwarding in the <i>ECM</i> for $L = 4$	171
6.12	Detailed Architecture for a 4 Stage Pipelined <i>ECM</i>	173
6.13	Finite State Machine for 4-Stage <i>ECM</i>	173
7.1	Side Channel Attacks	181
7.2	Side Channel Attack Set Up	185
7.3	Power Trace of AES	185
7.4	Simple Power Analysis	186
7.5	Power Trace for a Key with all 1	187
7.6	Power Trace for a Key with all 0	187
7.7	Power Trace when $k = (B9B9)_{16}$	188
7.8	Distribution of Hamming Weights	189
7.9	Distribution of DOM when the Hamming weights are distributed based on an uncorrelated bit	190

7.10	Plot of DOM for a wrong guess of next bit of x compared to a correct guess with simulated power traces	191
7.11	Effect of Faults on Secrets	192
7.12	Memory Accesses in a Block Cipher Implementation	195
7.13	Generic Structure of a Block Cipher	197
8.1	Three steps of DFA.	203
8.2	Basic structure of SPN ciphers	205
8.3	DFA and diagonal fault models. The first state matrix is an example of DM0. Only diagonal D_0 is affected by a fault. The second state matrix is an example of DM1. Both D_0 and D_3 are corrupted by a fault. The third state matrix is an example of DM2. Three diagonals D_0 , D_1 , and D_2 are corrupted by a fault. The last state matrix is an example of DM3, all 4 diagonals are corrupted in the fourth state matrix.	208
8.4	Fault propagation of diagonal faults. The upper row shows the diagonals that faults are injected in. The lower row shows the corresponding columns being affected.	208
8.5	Relationships between DFA fault models when faults are injected between (a) the output of 7 th and the input of 8 th round MixColumns, (b) output of 8 th and the input of 9 th round MixColumns.	210
8.6	Difference Across S-box	211
8.7	Basic Structure of Feistel Ciphers	213
8.8	Differences across the last 2 rounds	215
8.9	Differences across the Last Three Rounds	217
8.10	Model for data-flow parallelization in the second phase	220
8.11	Equivalence of Different kinds of Faults induced in Diagonal D_0 at the Input of 8 th Round of AES	223
8.12	Byte Inter-relations at the end of 9 th Round corresponding to different Diagonals being Faulty	223
8.13	Fault Propagation if Diagonals D_0 and D_1 are Affected	224
8.14	Fault Propagation if diagonals D_0 , D_1 and D_2 are affected	225
8.15	Flow of Faults	227
8.16	Flow of faults in AES-128 key schedule	231
8.17	Flow of faults in the last three rounds of AES-128	232
8.18	Flow of Faults in AES-192 Key Schedule when Fault is Induced at $K_{0,0}^{10}$	236
8.19	Flow of Faults in AES-192 Key Schedule when Fault is Induced at $K_{1,0}^{10}$	237
8.20	Flow of faults in the last three rounds of AES-192	238
8.21	Flow of Faults in AES-256 Key Schedule when the Fault is Induced at $K_{0,0}^{12}$	241
8.22	Flow of Faults in AES-256 Key Schedule when the Fault is Induced at $K_{0,0}^{11}$	241
8.23	Flow of faults in the AES-256 rounds	242
8.24	Datapath mixing	246
8.25	Partial hardware redundancy.	247
8.26	Slid CED Architecture for Time Redundancy	247
8.27	Parity-1 CED.	248
8.28	Parity-16 CED.	249
8.29	Parity-32 CED.	250
8.30	Robust Code	251
8.31	Hybrid redundancy (a) Algorithm level (b) Round level	252
8.32	Hybrid redundancy. (a) Operation level. (b) Optimized version.	253
8.33	Piret & Quisquater's Attack on Algorithm 8.7	262

9.1	Memory Hierarchy Structure	266
9.2	Direct Mapped Cache Memory with $2^b = 4$	267
9.3	Organization of Direct Mapped Cache Memory	268
9.4	A Round of a Typical Iterated Block Cipher	270
9.5	Transformations of the state in a Round of AES ($1 \leq r \leq 9$)	271
9.6	Cache Sets Corresponding to Table \mathcal{T}_j for Evict+Time and Prime+Probe Access Driven Attacks	276
9.7	Remote Timing Attacks	281
9.8	Distributions of Encryption Time of OpenSSL AES Block Cipher on Intel Core 2 Duo (E7500)	281
9.9	Timing Profile for \mathcal{D}_0 vs x_0 for AES on an Intel Core 2 Duo (E7500)	284
10.1	Power Attack Set Up	294
10.2	Power Attack Platform using SASEBO-GII	295
10.3	The Rolled Encryption Unit subjected to Power Analysis	296
10.4	Power Trace of an AES-128 encryption	296
10.5	Power Trace of the last round of AES	296
10.6	Electronic Noise for 9 th and 10 th round of AES-128	297
10.7	Different Power Consumption for $0 \rightarrow 1$ and $1 \rightarrow 0$ transition for CMOS switch	299
10.8	Non-linear filter generator based stream cipher	299
10.9	Power profile of the implemented stream cipher as obtained from the setup	300
10.10	Power Models	300
10.11	Progress of Bias of keys for simulated Power Profiles of AES-128	304
10.12	AES Differential plots for some target bits of S-Box-15 with 2500 traces	305
10.13	Error probability and Confidence vs. Window-size for the AES circuit	308
10.14	Error probability, Confidence and # of Traces (S-Box - 15) vs. Window-size for AES	309
10.15	Computing Hamming Distance in the Registers due to Last Round Transformation of AES	310
10.16	Progress of Bias of keys for CPA on simulated Power Profiles of AES-128	312
10.17	Guessing Entropy Plots for CPA on Simulated Power Traces	314
10.18	Conceptual Frequency Plot of a power profile	316
10.19	An Iterative AES: Target for the Power Attack	317
10.20	Plot of Guessing Entropy for CPA on Parallel Architecture for AES	318
10.21	A Serialized Architecture for AES	319
10.22	Plot of Guessing Entropy for CPA on Serialized Architecture for AES	319
10.23	Frequency Plot of Signal Component of Power Traces for the zeroth S-Box of Serialized Architecture for AES	320
10.24	Comparison of SNR values of Serialized Architecture for AES vs the Shuffled Architecture	321
10.25	Plot of Guessing Entropy for CPA on Shuffled Architecture for AES (averaged over all the S-Boxes)	321
10.26	Masked AES S-Box using Field Isomorphism	326
10.27	Architecture of a masked multiplier	329
11.1	Design of a Scan Chain	334
11.2	Generic Structure of a Stream Cipher	335
11.3	Secure Scan Architecture with Mirror Key Register [55]	341
11.4	Example Demonstrating Attack on Flipped-Scan	343
11.5	XOR-Chain Architecture[261]	344

12.1 Schemes for Boolean function modification and modification cell.	354
12.2 The proposed functional and structural obfuscation scheme by modification of the state transition function and internal node structure.	356
12.3 Modification of the initialization state space to embed authentication signature.	357
12.4 Hardware obfuscation design flow along with steps of the iterative node ranking algorithm.	360
12.5 SoC design modification to support hardware obfuscation. An on-chip controller combines the input patterns with the output of a PUF block to produce the activation patterns.	360
12.6 Challenges and benefits of the <i>HARPOON</i> design methodology at different stages of a hardware IP life cycle.	361
12.7 Example of a Verilog RTL description and its obfuscated version [82]: a) original RTL; b) technology independent, unoptimized gate-level netlist obtained through RTL compilation; c) obfuscated gate-level netlist; d) decompiled obfuscated RTL.	362
12.8 Design transformation steps in course of the proposed RTL obfuscation process.	363
12.9 Transformation of a block of RTL code into CDFG [83].	364
12.10 Example of hosting the registers of the mode-control FSM [83].	365
12.11 Examples of control-flow obfuscation: (a) original RTL, CDFG; (b) obfuscated RTL, CDFG [83].	365
12.12 Example of datapath obfuscation allowing resource sharing [83].	366
12.13 Example of RTL obfuscation by CDFG modification: (a) original RTL; (b) obfuscated RTL [83].	366
12.14 Binary Decision Diagram (BDD) of a modified node.	370
12.15 Flow diagram for the proposed STG modification-based RTL obfuscation methodology [82].	372
12.16 Flow diagram for the proposed CDFG modification-based RTL obfuscation methodology [83].	373
12.17 Scheme with <i>initialization key sequences</i> of varying length (3, 4 or 5).	376
13.1 Vulnerable steps of a modern IC life cycle [107].	382
13.2 Trojan taxonomy based on trigger and payload mechanisms.	383
13.3 Examples of Trojans with various trigger mechanisms.	384
13.4 Examples of analog payload Trojans.	385
13.5 Examples of (a) combinational and (b) sequential hardware Trojans that cause malfunction conditionally; Examples of Trojans leaking information (c) through logic values and (d) through side-channel parameter [221].	386
13.6 Example of multi-level attacks for both ASIC and FPGA realizations of cryptographic hardware.	387
13.7 Example of a designer-embedded Trojan in the hardware implementation of the AES cryptographic hardware.	388
13.8 Power traces of circuits with and without Trojan.	390
13.9 Experimental setup to simulate multi-level attack.	391
13.10 The <i>c17 ISCAS-85</i> circuit and three different inserted combinational Trojans. Trojan-1 and Trojan-3 are triggered and cause malfunction for 2 out of 32 possible input vectors. Trojan-2 is triggered relatively more often and cause malfunction for 4 out of the 32 possible input vectors.	392
13.11 Failure trends of the <i>c17</i> circuit with three types of inserted combinational Trojans (shown in Figs. 13.10(b)–(d)).	394

13.12 <i>c17</i> with an asynchronous binary counter-type Trojan. The reset signal for the Trojan counter has not been shown for simplicity.	396
13.13 Failure trends of the <i>c17</i> circuit with two types of inserted sequential Trojans (described in Sec. 13.4.3).	397
13.14 Xilinx Virtex-II configuration bitstream file organization [159, 257].	399
13.15 Proposed bitstream modification-based attack.	402
13.16 Examples of two possible cases of Type-I Trojan insertion.	403
13.17 Demonstration of successful configuration of the RO-based Trojan insertion. .	404
13.18 Effects of inserting 3-stage ring-oscillator based MHL in Virtex-II FPGA: (a) rise of temperature as a function of the number of inserted 3-stage ring oscillators; (b) theoretical aging acceleration factor as a function of percentage FPGA resource utilization by ring-oscillator Trojans.	405
14.1 Impact of sample size on trigger and Trojan coverage for benchmarks c2670 and c3540, $N = 1000$ and $q = 4$: (a) deviation of trigger coverage, and (b) deviation of Trojan coverage.	414
14.2 Impact of N (number of times a rare point satisfies its rare value) on the trigger/Trojan coverage and test length for benchmarks (a) c2670 and (b) c3540.	415
14.3 Integrated framework for rare occurrence determination, test generation using MERO approach, and Trojan simulation.	416
14.4 Trigger and Trojan coverage with varying number of trigger points (q) for benchmarks (a) c3540 and (b) c7552, at $N = 1000$, $\theta = 0.2$	418
14.5 Trigger and Trojan coverage with <i>trigger threshold</i> (θ) for benchmarks (a) c3540 and (b) c7552, for $N = 1000$, $q = 4$	418
14.6 FSM model with no loop in state transition graph.	419
15.1 (a) Average I_{DDT} values at 100 random process corners (with maximum variation of $\pm 20\%$ in inter-die V_{th}) for c880 circuit. The impact of Trojan (8-bit comparator) in I_{DDT} is masked by process noise. (b) Corresponding F_{max} values. (c) The F_{max} vs. I_{DDT} plot shows the relationship between these parameters under inter-die process variations. Trojan-inserted chips stand out from the golden trend line. (d) The approach remains effective under both inter-die and random intra-die process variations. A limit line is used to account for the spread in I_{DDT} values from the golden trend line. .	424
15.2 Effect of process variations on device threshold voltage in an IC.	426
15.3 Major steps in the multiple-parameter Trojan detection approach.	429
15.4 Schematic showing the functional modules of the AES cipher test circuit. The AES “Key Expand” module is clock-gated and operand isolation is applied to the “SBOX” modules to reduce the background current, thereby improving detection sensitivity. The Trojan instance is assumed to be in the logic block 2.	430
15.5 The correlation among I_{DDT} , I_{DDQ} and F_{max} can be used to improve Trojan detection confidence.	431
15.6 I_{DDT} vs. F_{max} relationship for both golden and tampered AES and IEU circuits showing the sensitivity of our approach for detecting different Trojan circuits.	434

15.7	Effect of random process variations is observed by performing Monte Carlo simulations with inter-die $\sigma = 10\%$ and random intra-die $\sigma = 6\%$ for the 32-bit IEU circuit with Trojan IV inserted. Using a 2% sensitivity limit line to accommodate for the random process variation effects, we can obtain 99.8% detection accuracy and limit the false alarms to 0.9%.	435
15.8	Choosing a faster clock period (3ns) and lowering the supply voltage from 1V to 0.8V gives better Trojan detection accuracy.	436
15.9	(a) Test PCB schematic. (b) Test circuit schematic. (c) Experimental setup. (d) Snapshot of measured I_{DDT} waveform from oscilloscope.	437
15.10	Measurement results for 10 FPGA chips showing I_{DDT} vs. F_{max} trend for the IEU test circuit with and without a 16-bit sequential Trojan (0.14% of original design).	437
15.11	Measured I_{DDT} vs. F_{max} results for 8 golden and 2 Trojan chips for the IEU circuit with and without a 4-bit sequential Trojan (0.03% of original design).	438
15.12	Sensitivity of Trojan detection decreases with Trojan size but improves by proper test vector selection.	438
15.13	Sensitivity of Trojan detection can be improved by measuring current from multiple supply pins.	439
15.14	Improvement of Trojan coverage by logic testing approach with insertion of observable test points (5 and 10).	440
15.15	Complementary nature of <i>MERO</i> and Side-channel analysis for maximum Trojan detection coverage in 32-bit IEU.	441
15.16	(a) A simple test circuit: a 4-bit Arithmetic Logic Unit (ALU). (b) A combinational Trojan inserted into the subtractor.	442
15.17	(a) Comparison of supply current between golden and tampered chip for four regions of a 4-bit ALU. (b) Correlation of region currents at different process points for golden and tampered ICs.	442
15.18	The major steps of the proposed self-referencing methodology. The steps for test vector generation for increasing sensitivity and threshold limit estimation for calibrating process noise are also shown.	447
15.19	Self-referencing methodology for detecting Trojan in the 32-bit ALU and FIR circuits. Blue and red lines (or points) denote golden and Trojan chips, respectively.	449
15.20	Sensitivity analysis with (a) different number of regions, (b) different circuit sizes, and (c) different Trojan sizes.	450
15.21	Experimental results for 8 golden and 2 tampered FPGA chips. Region slope matrix for (a) 32-bit DLX processor; (b) 32-bit ALU. The limit lines are obtained by analyzing the 8 golden chips. The red points denote the values for the Trojan-containing test chips while the blue points denote the values for the golden chips.	451
16.1	The obfuscation scheme for protection against hardware Trojans: (a) modified state transition graph and (b) modified circuit structure.	455
16.2	Fractional change in average number of test vectors required to trigger a Trojan, for different values of average fractional mis-estimation of signal probability f and Trojan trigger nodes (q).	459
16.3	Comparison of input logic cones of a selected flip-flop in <i>s15850</i> : (a) original design and (b) obfuscated design.	459
16.4	Steps to find unreachable states for a given set of S state elements in a circuit.	460
16.5	Framework to estimate the effectiveness of the obfuscation scheme.	462

16.6	Variation of protection against Trojans in <i>s1196</i> as a function of (a), (b) and (c): the number of added flip-flops in state encoding (<i>S</i>); (d), (e) and (f): the number of original state elements used in state encoding (<i>n</i>). For (a), (b) and (c), four original state elements were selected for state encoding, while for (d), (e) and (f), four extra state elements were added.	463
16.7	Effect of obfuscation on Trojans: (a) 2-trigger node Trojans (<i>q</i> = 2), and (b) 4-trigger node Trojans (<i>q</i> = 4).	465
16.8	Improvement of Trojan coverage in obfuscated design compared to the original design for (a) Trojans with 2 trigger nodes (<i>q</i> = 2) and (b) Trojans with 4 trigger nodes (<i>q</i> = 4).	465
16.9	Comparison of conventional and proposed SoC design flows. In the proposed design flow, protection against malicious modification by untrusted CAD tools can be achieved through obfuscation early in the design cycle.	467
16.10	Proposed FPGA design flow for protection against CAD tools.	468
16.11	Obfuscation for large designs can be efficiently realized using multiple parallel state machines which are constructed with new states due to additional state elements as well as unreachable states of original state machine.	468
16.12	Functional block diagram of a crypto-SoC showing possible Trojan attack to leak secret key stored inside the chip. Obfuscation coupled with bus scrambling can effectively prevent such attack.	469
16.13	Module isolation through moats and drawbridges [156].	470
16.14	<i>DEFENSE</i> hardware infrastructure for runtime monitoring. [7].	471
17.1	Basic principle of Optical PUF [234].	478
17.2	Basic principle of Coating PUF [234].	479
17.3	Ring Oscillator PUF [372].	482
17.4	Arbiter PUF.	482
17.5	Bistable Ring PUF.	483
17.6	(a) Basic delay element. (b) Loop PUF structure.	484
17.7	Memory PUF cell variants: (a) SRAM PUF cell; (b) Butterfly PUF cell; (c) Latch PUF cell.	484
17.8	Configuration memory and its use to initialize D Flip-flop [233].	486
17.9	Overview of the “Helper data” algorithm.	490
17.10	Types of Cloning Attacks.	491
18.1	Model of Genetic Programming.	497
18.2	Example of expression trees and their <i>crossover</i> : (a) and (b) parent expression trees; (c) and (d) children expression trees resulting from the crossover of the parent trees.	498
18.3	Ring Oscillator PUF (7-input, 1 output) as implemented by us.	499
18.4	Flowchart of the proposed Genetic Programming Model.	501
18.5	Variation of prediction accuracy with generation number for two reproduction schemes – <i>Tournament Selection Model</i> and <i>Elitist Model</i> , averaged over the results from the six PUF instances.	502
18.6	Impact of population size on prediction accuracy for two reproduction schemes – <i>Tournament Selection Model</i> and <i>Elitist Model</i> , averaged over the results from the six PUF instances.	502
18.7	Best Prediction accuracy for the six different PUF instances across 100 models built through Genetic Programming.	503

List of Tables

1.1	Addition on the set \mathbb{S}	5
1.2	Multiplication on the set \mathbb{S}	5
1.3	Addition on the set \mathbb{S}	14
1.4	Additive inverse operation on the set \mathbb{S}	14
1.5	Multiplication on the set \mathbb{S}	14
1.6	An Example Isomorphic Mapping between the field $GF(2^4)$ and $GF(2^2)^2$	26
2.1	Shift offsets for different block lengths	42
2.2	Scalar Multiplication using Double and Add to find $22P$	57
2.3	Computations for performing ECC scalar multiplication (projective vs affine coordinates)	62
3.1	State Transition Matrix of the Controller	75
4.1	XOR count of Square and Scaling circuit for $GF(2^4)$ Polynomial Basis	95
4.2	XOR count of Square and Scaling circuit for $GF(2^4)$ Normal Basis	96
4.3	Numbers of rounds (N_r) as a function of the block and key length	106
4.4	C_2C_1 for different Key and Data	109
4.5	Throughput in FPGA	109
4.6	Throughput in ASIC	109
4.7	Performances of Compared Cores	110
5.1	Comparison of LUT Utilization in Multipliers	123
5.2	Comparison of the Hybrid Karatsuba Multiplier with Reported FPGA Implementations	125
5.3	Inverse of $a \in GF(2^{163})$ using ITA	128
5.4	Inverse of $a \in GF(2^{233})$ using generic ITA	128
5.5	Comparison of LUTs Required for a Squarer and Quad Circuit for $GF(2^9)$	131
5.6	Comparison of Squarer and Quad Circuits on Xilinx Virtex 4 FPGA	132
5.7	Inverse of $a \in GF(2^{233})$ using Quad-ITA	133
5.8	Control Word for $GF(2^{233})$ Quad-ITA for Table 5.7	135
5.9	Comparison for Inversion on Xilinx Virtex E	138
5.10	Theoretical and Experimental Comparison for Different Number of Cascades of a 2^2 based ITA in the field $GF(2^{233})$	150
5.11	Experimental Performance of ITA for Different Fields	151
5.12	Comparison for inversion in $GF(2^{193})$ on <i>XCV3200efg1156</i>	151
6.1	Utility of Registers in the Register Bank	156
6.2	Parallel LD Point Doubling on the ECCP	158
6.3	Inputs and Outputs of the Register File for Point Doubling	158
6.4	Parallel LD Point Addition on the ECCP	160
6.5	Inputs and Outputs of the Register Bank for Point Addition	160

6.6	Inputs and Outputs of Regbank for Every State	161
6.7	Control Words for ECCP	162
6.8	LUT delays of Various Combinational Circuit Components	167
6.9	Scheduling Instructions for the <i>ECM</i>	168
6.10	Computation Time Estimates for Various Values of L for an <i>ECM</i> over $GF(2^{163})$ and FPGA with 4 input LUTs	172
6.11	Comparison of the Proposed <i>ECM</i> with FPGA based Published Results	174
7.1	SPA for the key $B9B9_{16}$	187
7.2	Dependence of Hamming weight on a target bit	189
8.1	A Summary of DFA of AES.	206
8.2	Computation of Algorithm 8.8	264
9.1	Mapping of Table T0 to a Direct-Mapped Cache of size $4KB$ ($2^\delta = 64$ and $2^b = 64$)	268
10.1	Transitions of an AND gate	301
10.2	AES Probable Key Matrix for S-Box 15 with 2500 Traces	306
10.3	Frequency Matrix for unmasked AES S-Box 15 with 2500 Traces	306
10.4	Error Probabilities(ϵ) for different window-sizes (AES)	308
10.5	Target Ciphertext Byte wrt. Register Position to Negotiate Inverse Shift Row	311
10.6	Sequence of Masking Operations	327
11.1	Internal states of Trivium	340
11.2	Hardware Overhead Comparison	345
12.1	Comparison of Decompilation-based and CDFG-based Obfuscation Approaches	367
12.2	Functional and Semantic Obfuscation Efficiency and Overheads for IP Cores for 10% area overhead target (CDFG Modification based Results at iso-delay)	374
12.3	Overall Design Overheads for Obfuscated IP Cores (STG Modification based Results at Iso-delay)	375
12.4	Area Overhead for Multi-length Key Support	376
12.5	Comparison of Throughput with Increasing Key Length	376
12.6	Area and Power Overhead for ISCAS-89 Benchmarks Utilizing Unused States (at Iso-delay)	377
12.7	Area and Power Overhead for IP Cores Utilizing Unused States (at Iso-delay)	378
13.1	Simulated Increase in Average Power	390
13.2	Hardware Overhead	391
13.3	Combinational Trojans: Trigger Condition and Trojan Activation Probability	394
13.4	Combinational Trojans and Associated Mean Time to Failure	394
13.5	Asynchronous Sequential Trojan: Trigger Condition and Trigger Probability [†]	396
13.6	Asynchronous Sequential Trojans and Associated Mean Time to Failure [†]	397
13.7	Virtex-II (XC2V40) Column Types and Frame Count [159]	399
14.1	Comparison of Trigger and Trojan Coverage Among ATPG Patterns [246], Random (100K, input weights: 0.5), and MERO Patterns for $q = 2$ and $q = 4$, $N = 1000$, $\theta = 0.2$	416

14.2 Reduction in Test Length by MERO Approach Compared to 100K Random Patterns and Associated Runtime, $q = 2$, $N=1000$, $\theta=0.2$	417
14.3 Comparison of Sequential Trojan Coverage between Random (100K) and MERO Patterns, $N = 1000$, $\theta = 0.2$, $q = 2$	419
15.1 Trojan Detection Sensitivity for Different Trojan Sizes in AES	434
15.2 Trojan Detection Sensitivity for Different Trojan Sizes in IEU	435
15.3 Trojan Coverage for ISCAS-85 Benchmark Circuits	441
15.4 Probability of Detection and Probability of False Alarm (False Positives) .	450
16.1 Effect of Obfuscation on Security Against Trojans (100,000 random patterns, 20,000 Trojan instances, $q = 2$, $k = 4$, $\theta = 0.2$)	464
16.2 Effect of Obfuscation on Security Against Trojans (100,000 random patterns, 20,000 Trojan instances, $q = 4$. $k = 4$, $\theta = 0.2$)	464
16.3 Design Overhead (at iso-delay) and Run-time [†] for the Proposed Design Flow	466
17.1 PUF Timeline	477
17.2 List of Metrics Proposed in Literature	487
18.1 Impact of Population Size on Execution Time	503
18.2 The Best Boolean Expressions Modelling the 6 PUFs	503

Part I

Background

Chapter 1

Mathematical Background

1.1	Introduction	3
1.2	Modular Arithmetic	4
1.3	Groups, Rings, and Fields	6
1.4	Greatest Common Divisors and Multiplicative Inverse	8
1.4.1	Euclidean Algorithm	9
1.4.2	Extended Euclidean Algorithm	10
1.4.3	Chinese Remainder Theorem	11
1.5	Subgroups, Subrings, and Extensions	13
1.6	Groups, Rings, and Field Isomorphisms	14
1.7	Polynomials and Fields	15
1.8	Construction of Galois Field	17
1.9	Extensions of Fields	18
1.10	Cyclic Groups of Group Elements	19
1.11	Efficient Galois Fields	21
1.11.1	Binary Finite Fields	21
1.12	Mapping between Binary and Composite Fields	24
1.12.1	Constructing Isomorphisms between Composite Fields	25
1.12.1.1	Mapping from $GF(2^k)$ to $GF(2^n)^m$, where $k = nm$	25
1.12.1.2	An Efficient Conversion Algorithm	26
1.13	Conclusions	27

“*Mathematics is the language with which God wrote the universe.* ”

Galileo Galilei

1.1 Introduction

Mathematics is often referred to as the *mother of all sciences*. It is every where, and without it no scientific study would have progressed. Mathematics defines not only the laws of the universe, but also gives us insight to solutions to many unsolved mysteries around us. Frankly, mathematical discoveries have often given rise to many questions, a large share of which are *unknown* to our present knowledge. These unproven results often are the backbone of science. To quote Vonn Neumann, *In mathematics you don't understand things. You just get used to them.* However, we start from these results and discover by logical deduction results, which are used in several discourses. The study of any *engineering* discipline, thus

relies on the applications of these mathematical principles to solve practical problems for the benefit of mankind.

The study of *hardware security*, like many other engineering subjects, relies heavily on mathematics. To start with, it involves the implementation of complex cryptographic algorithms on various platforms in an efficient manner. By the term efficiency, we often imply resource utilization and the time required by a hardware circuit. Although there are other measures, like power consumption, just restricting ourselves to these two classical objectives of hardware circuits poses several challenges, solutions to which are often obtained in the tricks in mathematics. The choice of a suitable algorithm for a specific purpose implies one has to be aware of the contemporary algorithms in the crypto literature. But these algorithms often are based on deep theories in mathematics: number theory, field theory, and the like. Hence to obtain a proper understanding and, most importantly, to compare the algorithms, one needs to develop a grasp of these topics. Once an algorithm is chosen, the underlying primitives must be understood: for example, a given algorithm may employ a multiplication step, more specifically a *finite field* multiplier. So the question is which multiplier should be chosen? Each design has their positives and negatives, thus a designer equipped with proper mathematical training and algorithm analysis prowess can make the choices in a prudent manner, which leads to efficient architectures. Today hardware designs of cryptographic algorithms are threatened with attacks, that exploit the properties of the implementations rather than the algorithms themselves. These attacks, commonly referred to as *side channel attacks* rely heavily on *statistics*. Thus in order to develop suitable defenses against these attacks, the designer also needs to understand these statistical methods. Knowledge of these methods shall help the designer not only to improve the existing attacks, but finally develop sound counter-measures. In short, design of *efficient* and *secured* implementations of cryptographic algorithms needs not only prudent engineering practices and architectural knowledge, but also understanding of the underlying mathematical principles.

In this chapter, we present an overview of some of the important mathematical concepts that are often useful for the understanding of hardware security designs.

1.2 Modular Arithmetic

Modular arithmetic is central to the discussion of ciphering algorithms. Starting from prehistoric classical ciphers to the present-day cryptographic algorithms, modular arithmetic is heavily used.

Also from the point of view of devices performing these ciphering operations, the underlying operations need to be done with finite numbers. An 8-bit computer bus, for example, can store only 256 numbers. The question is can we define arithmetic as we are used to with real numbers. Can we define *addition*, *subtraction*, *multiplication*, *division* on a finite space of numbers? Here is the beauty of mathematics, that we can have our own rules!

Consider, the set of numbers, $\mathbb{S} = \{0, \dots, 5\}$. Suppose we consider the two operations (later we define these notions more formally), addition (denoted by $+$) and multiplication (denoted by $*$). We obtain sums and products, such as shown in the **Tables 1.1** and **1.2**.

The above results show that we can define the laws of addition and multiplication, just as we have over the set of real numbers. Closer investigation shows that all the numbers in the set have additive inverse, that is another number which if added with it gives the number 0, commonly referred to as the *additive identity*. Similarly, all numbers except zero have

TABLE 1.1: Addition on the set \mathbb{S}

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	1	0
3	3	4	0	1	2
4	4	0	1	2	3

TABLE 1.2: Multiplication on the set \mathbb{S}

*	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

multiplicative inverse, that is another number which multiplied with it gives the number 1, known as the *multiplicative identity*. Since, we can define arithmetic on a finite set, we can envisage to develop ciphering algorithms on these numbers. For this we have to generalize this observation, in particular answer questions, like is there any speciality of the number 5 that we chose. It turns out that the primality of the number 5 has a very significant role in the theory that follows. We gradually develop the results subsequently. It may be kept in mind that we shall often state important results without formal proofs, which can be found in more details in textbooks on number theory and algebra.

We first state some definitions in the following:

Definition 1.2.1 An integer a is said to be congruent to an integer b modulo m , when m divides $b - a$ and is denoted as $a \equiv b \pmod{m}$.

Congruence modulo m is an equivalence relation on the integers.

- (**Reflexivity**): any integer is congruent to itself modulo m .
- (**Symmetry**): $a \equiv b \pmod{m} \Rightarrow b \equiv a \pmod{m}$.
- (**Transitivity**): $(a \equiv b \pmod{m}) \wedge (b \equiv c \pmod{m}) \Rightarrow a \equiv c \pmod{m}$.

The expression $a \equiv b \pmod{m}$ can also be written as $\exists k \in \mathbb{Z}, st. a = b + km$. Equivalently, when divided by m , both a and b leave the same remainder.

Definition 1.2.2 The equivalence class of $a \pmod{m}$ consists of all integers that are obtained by adding a with integral multiples of m . This class is also called the residue class of $a \pmod{m}$.

Example 1 Residue class of $1 \pmod{4}$ is the set $\{1, 1 \pm 4, 1 \pm 2 * 4, 1 \pm 3 * 4, \dots\}$

The residue classes \pmod{m} is denoted by the symbol $\mathbb{Z}/m\mathbb{Z}$. Each class has a representative element, $0, 1, \dots, m - 1$. The equivalence class for a representative element, say 0, is denoted by $[0]$. The set $\{0, 1, \dots, m - 1\}$ formed of the m incongruent residues is also called a *complete system*.

Example 2 Complete systems for a given modulo m is not unique. For example for $m = 5$, the sets $\{0, 1, 2, 3, 4\}$ and $\{-12, -15, 82, -1, 31\}$ are both complete systems.

The following theorem is straightforward and is stated without any proof.

Theorem 1 $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, implies that $-a \equiv -b \pmod{m}$, $a + c \equiv b + d \pmod{m}$, and $a * c \equiv b * d \pmod{m}$, $\forall a, b, c, d, m \in \mathbb{Z}$.

This result is particularly useful as it shows that operations in modular arithmetic can be made much easier by performing intermediate modular reduction. The following example illustrates this point.

Example 3 Prove that $2^{2^5} + 1$ is divisible by 641.

We note that $641 = 640 + 1 = 5 * 2^7 + 1$. Thus,

$$\begin{aligned} 5 * 2^7 &\equiv -1 \pmod{641} \\ \Rightarrow (5 * 2^7)^4 &\equiv (-1)^4 \pmod{641} \\ \Rightarrow 5^4 * 2^{28} &\equiv 1 \pmod{641} \\ \Rightarrow (625 \pmod{641}) * 2^{28} &\equiv 1 \pmod{641} \\ \Rightarrow (-2^4) * 2^{28} &\equiv 1 \pmod{641} \\ \Rightarrow 2^{32} &\equiv -1 \pmod{641} \end{aligned}$$

This example shows that the complicated computation can be simplified by performing modular reductions and subsequently carrying on the computations. This fact holds true for all the ciphering operations which work on finite sets of data that we shall subsequently study.

1.3 Groups, Rings, and Fields

Next we develop the concepts of groups and fields, which are central to the study of ciphers. We browse through a series of definitions in order to build up the concept.

Definition 1.3.1 If X is a set, a map $\circ: X \times X \rightarrow X$, which transforms an element (x_1, x_2) to the element $x_1 \circ x_2$ is called an operation.

Example 4 The sum of the residue classes $a + m\mathbb{Z}$ and $b + m\mathbb{Z}$ is $(a + b) + m\mathbb{Z}$.

Example 5 The product of the residue classes $a + m\mathbb{Z}$ and $b + m\mathbb{Z}$ is $(a * b) + m\mathbb{Z}$.

Definition 1.3.2 An operation \circ on X is associative if $(a \circ b) \circ c = a \circ (b \circ c)$, for all $a, b, c \in X$.

Definition 1.3.3 A pair (H, \circ) consisting of a set H and an associative operation \circ on H is called a semigroup. The semigroup is called abelian or commutative if the operation is also commutative.

Example 6 The pairs $(\mathbb{Z}, +)$, $(\mathbb{Z}, *)$, $(\mathbb{Z}/m\mathbb{Z}, +)$, $(\mathbb{Z}/m\mathbb{Z}, *)$ are abelian semigroups.

Definition 1.3.4 An identity element of the semigroup (H, \circ) is an element $e \in H$, which satisfies $e \circ a = a \circ e = a$, $\forall a \in H$.

Definition 1.3.5 If the semigroup contains an identity element it is called a monoid.

It can be easily seen that the semigroup can have at most one identity element. This is so, if there are two identities, e and e' , we have $e \circ e' = e = e'$.

Definition 1.3.6 If $e \in H$ is an identity element of the semigroup (H, \circ) (i.e. it is a monoid), then $b \in H$ is called an inverse of $a \in H$ if $a \circ b = b \circ a = e$.

If a has an inverse it is called invertible in the semigroup H . In a monoid, each element can have at most one inverse. To see this, assume that an element $a \in H$ has two inverses, b and b' and let e be the identity element. Thus from the definition, $a \circ b = b \circ a = a \circ b' = b' \circ a = e \Rightarrow b' = b' \circ e = b' \circ (a \circ b) = (b' \circ a) \circ b = e \circ b = b$.

Example 7 $(\mathbb{Z}, +)$ has an identity element 0, and inverse $-a$. $(\mathbb{Z}, *)$ has an identity element 1, while the only invertible elements are +1 and -1. $(\mathbb{Z}/m\mathbb{Z}, +)$ has as identity element, $m\mathbb{Z}$. The inverse is $-a + m\mathbb{Z}$. This monoid is often referred to as \mathbb{Z}_m . $(\mathbb{Z}/m\mathbb{Z}, *)$ has as identity element $1 + m\mathbb{Z}$. The invertible elements of \mathbb{Z} are $t \in \mathbb{Z}$, st. $\gcd(t, m) = 1$, i.e., if and only if t and m are mutually co-prime. The invertible elements of this monoid is a set denoted by \mathbb{Z}_m^* .

Definition 1.3.7 A group is a monoid in which every element is invertible. The group is commutative or abelian if the monoid is commutative.

Example 8 $(\mathbb{Z}, +)$ and $(\mathbb{Z}/m\mathbb{Z}, +)$ are abelian groups. However, $(\mathbb{Z}, *)$ is not an abelian group. $((\mathbb{Z}/m\mathbb{Z}) \setminus \{0\}, *)$ is an abelian group if and only if m is prime.

Definition 1.3.8 A ring is a triplet $(R, +, *)$ such that $(R, +)$ is an abelian group, $(R, *)$ is a monoid and the operation $*$ distributes over the operation $+$, i.e. $\forall x, y, z \in R, x * (y + z) = (x * y) + (x * z)$. The ring is called commutative if the monoid $(R, *)$ is commutative.

Definition 1.3.9 A field is a commutative ring $(R, +, *)$ in which every element in the monoid $(R, *)$ is invertible.

Example 9 The set of integers is not a field. The set of real and complex numbers form a field. The residue class modulo a prime number, except the element 0, is a field.

Thus summarizing the above concepts, the definition of groups, rings, and fields are rewritten below:

Definition 1.3.10 A group denoted by $\{\mathbb{G}, \cdot\}$, is a set of elements \mathbb{G} with a binary operation \cdot , such that for each ordered pair (a, b) of elements in \mathbb{G} , the following axioms are obeyed [122]/[369]:

- Closure: If $a, b \in \mathbb{G}$, then $a \cdot b \in \mathbb{G}$.
- Associative: $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all $a, b, c \in \mathbb{G}$.
- Identity element: There is a unique element $e \in \mathbb{G}$ such that $a \cdot e = e \cdot a = a$ for all $a \in \mathbb{G}$.
- Inverse element: For each $a \in \mathbb{G}$, there is an element $a' \in \mathbb{G}$ such that $a \cdot a' = a' \cdot a = e$

If the group also satisfies $a \cdot b = b \cdot a$ for all $a, b \in \mathbb{G}$ then it is known as a *commutative* or an *abelian group*.

Definition 1.3.11 A ring denoted by $\{\mathbb{R}, +, *\}$ or simply \mathbb{R} is a set of elements with two binary operations called addition and multiplication, such that for all $a, b, c \in \mathbb{R}$ the following are satisfied:

- \mathbb{R} is an abelian group under addition.
- The closure property of \mathbb{R} is satisfied under multiplication.
- The associativity property of \mathbb{R} is satisfied under multiplication.
- There exists a multiplicative identity element denoted by 1 such that for every $a \in \mathbb{F}$, $a * 1 = 1 * a = 1$.
- Distributive Law : For all $a, b, c \in \mathbb{R}$, $a * (b + c) = a * b + a * c$ and $(a + b) * c = a * c + b * c$.

The set of integers, rational numbers, real numbers, and complex numbers are all rings. A ring is said to be commutative if the commutative property under multiplication holds. That is, for all $a, b \in \mathbb{R}$, $a * b = b * a$.

Definition 1.3.12 A field denoted by $\{\mathbb{F}, +, *\}$ or simply \mathbb{F} is a commutative ring which satisfies the following properties

- Multiplicative inverse : For every element $a \in \mathbb{F}$ except 0, there exists a unique element a^{-1} such that $a \cdot (a^{-1}) = (a^{-1}) \cdot a = 1$. a^{-1} is called the multiplicative inverse of the element a .
- No zero divisors : If $a, b \in \mathbb{F}$ and $a \cdot b = 0$, then either $a = 0$ or $b = 0$.

As we have seen the set of rational numbers, real numbers and complex number are examples of fields, while the set of integers is not. This is because the multiplicative inverse property does not hold in the case of integers.

Definition 1.3.13 The characteristic of a field \mathbb{F} is the minimal value of the integer k , such that for any element $a \in \mathbb{F}$, $a + \dots + a$ (k times) $= k \cdot a = 0$, where $0 \in \mathbb{F}$, is the additive identity of the field \mathbb{F} . Since the inverse a^{-1} exists, an alternative way of defining is by the equation, $k \cdot 1 = 0$, where 1 is the multiplicative identity of the field \mathbb{F} .

The characteristic of a field is a prime number. If k is not prime, one can factor $k = k_1 k_2$, $1 < k_1, k_2 < k$. Thus, for $1 \in \mathbb{F}$, $k \cdot 1 = (k_1 \cdot 1) \cdot (k_2 \cdot 1) = 0$. Since in a field there are no zero-divisors, either $k_1 \cdot 1 = 0$, or $k_2 \cdot 1 = 0$. Since k is the smallest positive integer such that $k \cdot 1 = 0$, k has to be prime.

The residue class $\mathbb{Z}/p\mathbb{Z}$ is of extreme importance to cryptography. Each element in the set, a has a multiplicative inverse if and only if $\gcd(a, p) = 1$. This happens if and only if p is prime. Thus,

Theorem 2 The residue class $\mathbb{Z}/p\mathbb{Z}$ is a field if and only if p is prime.

1.4 Greatest Common Divisors and Multiplicative Inverse

Computing multiplicative inverse is a central topic of number theory and finite fields. This operation is at the heart of cryptography, and is central to several important public

Algorithm 1.1: Euclidean Algorithm for computing gcd

Input: $a, b, a > b$
Output: $d = \gcd(a, b)$

```

1  $r_0 = a$ 
2  $r_1 = b$ 
3  $m = 1$ 
4 while  $r_m \neq 0$  do
5    $q_m = \left\lfloor \frac{r_{m-1}}{r_m} \right\rfloor$ 
6    $r_{m+1} = r_{m-1} - q_m r_m$ 
7    $m = m + 1$ 
8 end
9 return  $r_{m-1}$ 
```

key algorithms. While there are several techniques for computing multiplicative inverses, Euclidean algorithm is one of the most well known techniques. The original Euclidean algorithm computes the greatest common divisor of two integers (elements).

1.4.1 Euclidean Algorithm

In this section, we present a discussion on greatest common divisor (gcd) of two integers. For simplicity, we restrict the discussion to positive integers. A non-zero integer has a finite number of divisors because the divisors must lie between 0 and n . An integer d is called a common divisor of a and b if it divides both a and b ; that is $d | a$ and $d | b$.

Definition 1.4.1 *If a and b are integers that are not both zero, then the greatest common divisor d of a and b is the largest of the common divisors of a and b . We denote this as: $d = \gcd(a, b)$.*

Because every number divides 0, there is strictly no greatest common divisor of 0 and 0; for convenience we set $\gcd(0, 0) = 0$. Further, it may be easy to observe that $\gcd(a, a) = a$ and $\gcd(a, 0) = a$. The following facts are useful for computing the gcd of two integers:

Theorem 3 *If a , b , and k are integers, then*

$$\gcd(a, b) = \gcd(a + kb, b)$$

This follows from the fact that the set of common divisors of a and b is the same set as the common divisors of $a + kb$ and b . Thus we have the following useful corollary:

Corollary 1 *If a and b are positive integers, then $\gcd(a, b) = \gcd(a \bmod b, b) = \gcd(b, a \bmod b)$.*

The proof of correctness of the above algorithm is easy to verify. It follows because of the following sequence of equations:

$$\begin{aligned}
\gcd(a, b) &= \gcd(r_0, r_1) \\
&= \gcd(q_1 r_1 + r_2, r_1) \\
&= \gcd(r_2, r_1) \\
&= \gcd(r_1, r_2) \\
&= \dots \\
&= \gcd(r_{m-1}, r_m) \\
&= r_m
\end{aligned}$$

Thus the Euclidean algorithm can be used to check the gcd of two positive integers. It can also be used for checking for the existence of inverse of an element, a modulo another element n . An extended version of this algorithm can also be used for the computation of the inverse, $a^{-1} \bmod n$ and is explained in the next subsection.

1.4.2 Extended Euclidean Algorithm

Let us start this section, with an example to compute the inverse of 28 mod 75. If we apply the Euclidean Algorithm (EA) to compute the $\gcd(28, 75)$, the following steps are performed:

$$\begin{aligned}
75 &= 2 \times 28 + 19 \\
28 &= 1 \times 19 + 9 \\
19 &= 2 \times 9 + 1 \\
9 &= 9 \times 1
\end{aligned}$$

The algorithm terminates at this point and we obtain that the gcd is 1. As stated previously, this implies the existence of the inverse of 28 mod 75. The inverse can be easily obtained by observing the sequence of numbers generated while applying the EA above. In order to compute the inverse, we first express the gcd as a linear combination of the numbers, 28 and 75. This can be easily done as:

$$\begin{aligned}
19 &= 75 - 2 \times 28 \\
9 &= 28 - 19 = 28 - (75 - 2 \times 28) = -75 + 3 \times 28 \\
1 &= 19 - 2 \times 9 = (75 - 2 \times 28) - 2 \times (-75 + 3 \times 28) = 3 \times 75 - 8 \times 28
\end{aligned}$$

It may be observed that the linear expression: $1 = 3 \times 75 - 8 \times 28$ is unique. Thus the inverse of 28 mod 75 can be easily obtained by taking modulo 75 to the above expression, and we observe that $-8 \times 28 \equiv 1 \bmod 75$. This shows that $28^{-1} \bmod 75 \equiv -8 = 67$.

Thus if the inverse of an integer $a \bmod n$ exists (ie. $\gcd(a, n) = 1$), one applies the Euclidean Algorithm on a and n and generates a sequence of remainders. These remainders can be expressed as unique linear combinations of the integers a and n . The Extended Euclidean Algorithm (EEA) is a systematic method for generating the coefficients of these linear combinations.

The coefficients are generated by EEA as two series: s and t , and the final $\gcd(a, b) = sa + tn$. The series elements, $s_0, s_1, \dots, s_m = s$, and $t_0, t_1, \dots, t_m = t$, are generated along with the remainders. The above series are obtained through the following recurrences:

Algorithm 1.2: Extended Euclidean Algorithm for computing inverse

Input: $a, n, a < n, \gcd(a, n) = 1$
Output: $r = \gcd(a, n), s \equiv a^{-1} \pmod{n}$

```

1   $r_0 = n$ 
2   $r_1 = a$ 
3   $s_0 = 1$ 
4   $s_1 = 0$ 
5   $m = 1$ 
6  while  $r_m > 0$  do
7     $q_m = \lfloor \frac{r_{m-1}}{r_m} \rfloor$ 
8     $r_{m+1} = r_{m-1} - q_m r_m$ 
9     $s_{m+1} = s_{m-1} - q_m s_m$ 
10    $m = m + 1$ 
11 end
12  $r = r_{m-1}, s = s_{m-1}$ 
13 return  $r, s$ 
```

$$s_j = \begin{cases} 1 & \text{if } j = 0 \\ 0 & \text{if } j = 1 \\ s_{j-2} - q_{j-1}s_{j-1} & \text{if } j \geq 2 \end{cases} \quad (1.1)$$

$$t_j = \begin{cases} 0 & \text{if } j = 0 \\ 1 & \text{if } j = 1 \\ t_{j-2} - q_{j-1}t_{j-1} & \text{if } j \geq 2 \end{cases} \quad (1.2)$$

For $0 \leq j \leq m$, we have that $r_j = s_j r_0 + t_j r_1$, where the r_j s are defined in the EA, and the s_j s and t_j s are as defined in the above recurrences. Note that the final remainder, $r = \gcd(a, n) = 1$ is expressed as $sa + tn$, and hence, $1 \equiv sa \pmod{n}$. Thus $s \equiv a^{-1} \pmod{n}$. Thus while computing the inverse the computations of the t -series is not required and can be omitted.

The operations are summarized in algorithm 1.2.

In the next section, we present another application of the Euclidean Algorithm (EA), called the Chinese Remainder Theorem (CRT). This theorem is often useful in development of some implementations of the RSA algorithm.

1.4.3 Chinese Remainder Theorem

Consider there are t integers m_1, m_2, \dots, m_t , which are relatively prime in pairs, i.e. $\gcd(m_i, m_j) = 1$ for any distinct i and j where $1 \leq i, j \leq t$.

Let a_1, a_2, \dots, a_t be any t integers. The Chinese Remainder Theorem says, then there is a *unique* number x , which satisfies the property:

$$x \equiv a_i \pmod{m_i}, i = 1, 2, \dots, t$$

The uniqueness of x is defined in the following sense: Let M be the product $m_1 m_2 \dots m_t$, and let y satisfy the above system of congruences. Then $y \equiv x \pmod{M}$.

The uniqueness can be proved in the following inductive manner on the variable t . For $t = 1$, it suffices to take $x = a_1$. The uniqueness is straight forward.

Let us assume that the result holds for $t = k$. We prove the result (that such an x exists and is unique in the sense asserted) for $t = k + 1$. Consider the system of congruences:

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ &\vdots \\ x &\equiv a_k \pmod{m_k} \\ x &\equiv a_{k+1} \pmod{m_{k+1}} \end{aligned}$$

The moduli are relatively prime in pairs. From the inductive hypothesis, there is a number x' satisfying the first k congruences.

It is easy to check that the product of the first k integers, $m_1 m_2 \dots m_k$ are relatively prime to the integer m_{k+1} . Hence from Extended Euclidean Algorithm, there are unique integers, u and v , such that:

$$um_1 m_2 \dots m_k + v m_{k+1} = 1$$

Multiplying both sides by $(a_{k+1} - x')$, we obtain;

$$u(a_{k+1} - x')m_1 m_2 \dots m_k + v(a_{k+1} - x')m_{k+1} = (a_{k+1} - x')$$

This can be written as:

$$x' + u'' m_1 m_2 \dots m_k = a_{k+1} + v'' m_{k+1}$$

Here we have, $u'' = u(a_{k+1} - x')$ and $v'' = -v(a_{k+1} - x')$. Thus, denoting $x'' = x' + u'' m_1 m_2 \dots m_k$, we have:

$$x'' \equiv a_{k+1} \pmod{m_{k+1}}$$

On the other hand,

$$x'' \equiv x' \equiv a_i \pmod{m_i}, i = 1, \dots, k$$

This proves the existence of a solution.

The uniqueness can be observed as follows: From the induction hypothesis, x' is a unique solution to the first k congruences mod $m = m_1 \dots m_k$. Consider for $k + 1$ congruences, there are two distinct solutions x and x_1 . However since these values also are a solution for the first k congruences, we have by the induction hypothesis: $x \equiv x_1 \pmod{m}$. Further since both satisfy the last congruence: $x \equiv x_1 \pmod{m_{k+1}}$. Note that m and m_{k+1} are relatively prime. Thus, $x \equiv x_1 \pmod{mm_{k+1}}$. This proves that $x \equiv x' \pmod{M}$, where $M = mm_{k+1} = m_1 \dots m_k m_{k+1}$. Hence, $x'' \equiv x' \pmod{M}$. This completes the proof.

The above theorem, though it gives a hint on how to obtain the solution, is not constructive. Next we present a solution to the above problem, called as the Chinese Remainder Algorithm (CRA). The above algorithm occurred for the first time in the mathematical classic of Sun Zi, a mathematician in ancient China.

The Chinese Remainder Theorem (CRT) is a classic example of expressing the whole in parts. The complete information is expressed as subparts, and the problem is to recover the original information from the parts. As can be understood, there can be several applications like secret sharing schemes, efficient implementations as the parts are often smaller numbers.

The CRT problem is to find the original x from the parts, a_i , where:

$$x \equiv a_i \pmod{m_i}, i = 0, 1, \dots, n-1 \quad (1.3)$$

where m_i are pairwise relatively primes, and a_i are integers. In order to solve Equations 1.3 we compute the values, s_0, s_1, \dots, s_{n-1} satisfying:

$$s_i \equiv 1 \pmod{m_i}, \quad s_i \equiv 0 \pmod{m_j} \text{ for } i \neq j \quad (1.4)$$

Then $x = \sum_{i=0}^{n-1} a_i s_i \pmod{m}$ is the smallest non-negative integer congruent modulo m , where $m = m_0 m_1 \dots m_{n-1}$, which is a solution to Equation 1.3.

Let $M_i = m/m_i$. Then the above congruence is equivalent to solve:

$$M_i y_i \equiv 1 \pmod{m_i} \quad (1.5)$$

It may be noted that y_i is the multiplicative inverse of M_i modulo m_i . Further it may be observed that the inverse exists, as M_i and m_i are relatively prime. The inverse can be computed using the Extended Euclidean Algorithm.

It can be easily seen then the congruence of Equation 1.4 can be solved by setting $s_i = M_i y_i$, as then $s_i \equiv 1 \pmod{m_i}$, and $s_i \equiv 0 \pmod{m_j}$, for $i \neq j$.

We discuss the above in the following theorem, called the Chinese Remainder Theorem.

Theorem 4 Suppose m_0, m_1, \dots, m_{n-1} are pairwise relatively prime positive integers, and suppose a_0, a_1, \dots, a_{n-1} are integers. Then the system of congruences $x \equiv a_i \pmod{m_i}$, $0 \leq i < n$ has a unique solution modulo $m = m_0 \times m_1 \dots \times m_{n-1}$, $x = \sum_{i=0}^{n-1} a_i M_i y_i \pmod{m}$, where $M_i = m/m_i$, and $y_i = M_i^{-1} \pmod{m_i}$, for $0 \leq i < n-1$.

1.5 Subgroups, Subrings, and Extensions

Groups and rings give a nice theory of subgroups and subrings, which may be extended to fields as well. The theory of subrings (subfields) gives a very useful concept of equivalent rings (subrings), which are technically called isomorphic subrings (subfields), which are often useful for efficient implementations.

We first define subgroups and then subrings:

Definition 1.5.1 Let \mathbb{R} be a group under the operation $+$. Let \mathbb{S} be a subset of \mathbb{R} . \mathbb{S} is called a subgroup of \mathbb{R} if and only if:

- $a \in \mathbb{S}$ and $b \in \mathbb{S} \Rightarrow a + b \text{ belong to } \mathbb{S}$.
- $a \in \mathbb{S} \Rightarrow -a \in \mathbb{S}$, that is the additive inverse of \mathbb{S} also belongs to \mathbb{S} .

A subring is an extension of the concept of subgroups, reminding that a ring is defined wrt. two operations, say $+$ and $*$.

Definition 1.5.2 Let \mathbb{R} be a ring under the operations $+$ and $*$. Let \mathbb{S} be a subset of \mathbb{R} . \mathbb{S} is called a subring of \mathbb{R} if and only if:

- \mathbb{S} is a subgroup of \mathbb{R} .
- \mathbb{S} is closed under the operation $*$, called as multiplication.

TABLE 1.3: Addition on the set \mathbb{S}

+	0	3	6	9
0	0	3	6	9
3	3	6	9	0
6	6	9	0	3
9	9	0	3	6

TABLE 1.4: Additive inverse operation on the set \mathbb{S}

-	0	3	6	9
0	0	9	6	3

TABLE 1.5: Multiplication on the set \mathbb{S}

*	0	3	6	9
0	0	0	0	0
3	0	9	6	3
6	0	6	0	6
9	0	3	6	9

Every ring has two trivial subrings: itself and the set $\{0\}$.

Example 10 Consider the set $\mathbb{R} = \mathbb{Z}/12\mathbb{Z}$. Consider the subset $\mathbb{S} = \{0, 3, 6, 9\}$. The following Tables (Tables 1.3, 1.4, 1.5 confirm that it is a subring).

It may be noted that the set \mathbb{S} does not form a group wrt. multiplication. Further although the set $\mathbb{R} = \mathbb{Z}/12\mathbb{Z}$ possesses a multiplicative identity, \mathbb{S} does not. It may be interesting to note that it may also be possible for \mathbb{S} to have a multiplicative identity, but not \mathbb{R} . The two sets can also have same or different identities wrt. multiplication. However as per our definition for rings, we impose the further condition that \mathbb{S} has to be a ring for it to qualify as a subring. Thus the subring also has to have a *multiplicative identity*. When \mathbb{S} is the subring of \mathbb{R} , the later is called as the ring extension of the former. When both \mathbb{S} and \mathbb{R} are fields, the later is called as the *field extension* of the former. Equivalently, one also says that $\mathbb{S} \subseteq \mathbb{R}$ is a field extension or \mathbb{R} is a field extension over \mathbb{S} .

1.6 Groups, Rings, and Field Isomorphisms

Often a group, ring or a field can be expressed in several equivalent forms. For two groups, G_1 and G_2 , a surjective function $f : G_1 \rightarrow G_2$ is said to be a homomorphism if and only if: $f(x \circ y) = f(x) \dagger f(y)$. Note that the group operations on the left need not be the same as on the right. Thus the homomorphism is a function which not only transforms the elements, but also transforms the operators. For simplicity, we state $f(x \cdot y) = f(x) \cdot f(y)$, though the operators on the left and right are defined over different groups.

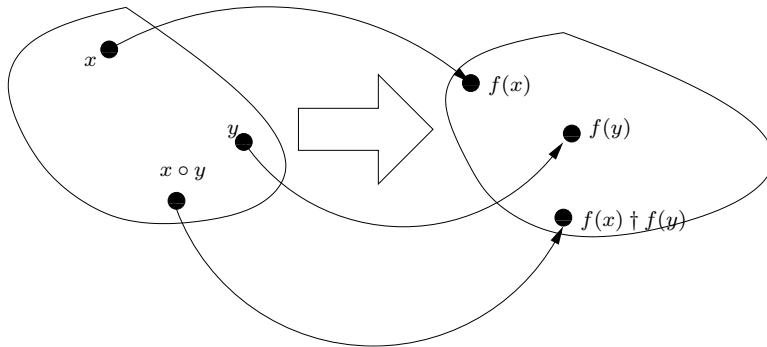


FIGURE 1.1: Homomorphism

The following theorems state two important properties for homomorphisms on groups.

Theorem 5 *If \$f : G_1 \rightarrow G_2\$ is a group homomorphism then \$f(e_1) = e_2\$, where \$e_1\$ is the identity of \$G_1\$ and \$e_2\$ is the identity of \$G_2\$.*

Let \$x\$ be an element of \$G_1\$. Thus \$f(x)\$ is an element of \$G_2\$. Thus, \$f(x) = f(x \cdot e_1) = f(x) \cdot f(e_1) \Rightarrow f(x) \cdot e_2 = f(x) \cdot f(e_1)\$. Thus, \$f(x) = f(x) \cdot f(e_1)\$. It may be noted that the cancellation is allowed, owing to the existence of the multiplicative inverse, \$[f(x)]^{-1}\$ and associativity of the groups.

Theorem 6 *If \$f : G_1 \rightarrow G_2\$ is a group homomorphism then for every \$x \in G_1\$, \$f(x^{-1}) = (f(x))^{-1}\$.*

We have \$f(x \cdot x^{-1}) = f(e_1) = e_2\$. Also, \$f(x \cdot x^{-1}) = f(x) \cdot f(x^{-1})\$. Hence, we have \$f(x^{-1}) = (f(x))^{-1}\$.

Example 11 *Let \$G_1\$ be the group of all real numbers under multiplication, and \$G_2\$ be the group of all real numbers under addition. The function defined as \$f : G_1 \rightarrow G_2\$, where \$f(x) = \log_e(x) = \ln(x)\$, is a group homomorphism.*

An injective (one-to-one) homomorphism is called an **isomorphism**.

The idea of homomorphism and hence isomorphism can be extended to rings and fields in a similar fashion. In these extensions the only difference is from the fact that a ring and a field are defined wrt. two operations, denoted by \$+\$ and \$\circ\$.

Let \$R_1\$ and \$R_2\$ be rings and consider a surjective function, \$f : R_1 \rightarrow R_2\$. It is called a ring isomorphism if and only if:

1. \$f(a + b) = f(a) + f(b)\$ for every \$a\$ and \$b\$ in \$R_1\$
2. \$f(a \circ b) = f(a) \circ f(b)\$ for every \$a\$ and \$b\$ in \$R_1\$

An obvious extension of the previous two theorems to the rings \$R_1\$ and \$R_2\$, is \$f(0) = 0\$, and \$f(-x) = -f(x)\$, for every \$x \in R_1\$. If \$R_1\$ has a multiplicative identity denoted by 1, and \$R_2\$ has a multiplicative identity denoted by \$1'\$, we have \$f(1) = 1'\$.

Further, if \$x\$ is a unit in the ring \$R_1\$, then \$f(x)\$ is a unit in the ring \$R_2\$, and \$f(x^{-1}) = [f(x)]^{-1}\$. These properties also holds for fields. The property of isomorphisms have been found to be useful for developing efficient implementations for finite field-based algorithms. The fact of isomorphism is utilized to transform a given field into another isomorphic field, perform operations in this field, and then transform back the solutions. The advantage in such implementations occurs from the fact that the operations in the newer field are more efficient to implement than the initial field.

1.7 Polynomials and Fields

Elements of a field are often represented in the form of polynomials. Let \mathbb{R} be a commutative ring, with unit element 1. A polynomial in the variable $X \in \mathbb{R}$, is an expression:

$$f(X) = a_n X^n + a_{n-1} X^{n-1} + \dots + a_1 X + a_0,$$

where X is the variable and the coefficients a_0, \dots, a_n of the polynomial are elements of \mathbb{R} . The set of all polynomials over \mathbb{R} in the variable X is denoted by $\mathbb{R}[X]$.

If the *leading coefficient* of the polynomial f , denoted by a_n is nonzero, then the degree of the polynomial is said to be n . A *monomial* is a polynomial whose all co-efficients except the leading one are zero.

If the value of a polynomial vanishes for a particular value of the variable, $r \in \mathbb{R}$: ie. $f(r) = 0$, then r is called a *root* or *zero* of f .

Consider two polynomials, $f(x) = \sum_{i=0}^n a_i X^i$, and $g(x) = \sum_{i=0}^m b_i X^i$, defined over R , and suppose $n \geq m$. Then the sum of the polynomials is defined by $(f+g)(X) = \sum_{i=0}^m (a_i + b_i) X^i + \sum_{i=(m+1)}^n a_i X^i$. The number of operations needed is $O(n+1)$.

The product of the polynomials f and g is $(fg)(X) = \sum_{k=0}^{n+m} c_k X^k$, where $c_k = \sum_{i=0}^k a_i b_{k-i}$, $0 \leq k \leq n+m$. The coefficients, a_i and b_i which are not defined are set to 0. The multiplication requires $O(nm)$ computations, considering the products and additions.

Example 12 The set $\mathbb{Z}/3\mathbb{Z}$ contains the elements, 0, 1 and 2. Consider the polynomials, $f(X) = X^2 + X + 1$, and $g(X) = X^3 + 2X^2 + X \in (\mathbb{Z}/3\mathbb{Z})[X]$. It can be checked that the first polynomial has a zero at 1, while the later has at 2.

The sum of the polynomials, is $(f+g)(X) = X^3 + (1+2)X^2 + (1+1)X + 1 = X^3 + 2X + 1$. The product of the polynomials is denoted by $fg(X) = X^5 + (1+2)X^4 + (1+2+1)X^3 + (2+1)X^2 + X = X^5 + X^3 + X$.

The set of polynomials, $\mathbb{R}[X]$ forms a commutative ring with the operations, addition and multiplication. If \mathbb{K} is a field, then the ring $\mathbb{K}[X]$ of polynomials over \mathbb{K} contains no zero divisors, that there does not exist two non-zero polynomials, $a(X)$ and $b(X)$ st. $a(X)b(X) = 0$, the zero polynomial.

The following theorem is stated without proof but can be followed from any classic text of number theory:

Theorem 7 Let $f(X), g(X) \in \mathbb{K}[X]$, $g(X) \neq 0$. Then there are uniquely determined polynomials $q(X), r(X) \in \mathbb{K}[X]$, with $f(X) = q(X)g(X) + r(X)$ and $r(X) = 0$ or $\deg r(X) < \deg g(X)$. The polynomials $q(X)$ and $r(X)$ are referred to as the quotient and remainder polynomials.

Example 13 Let, $g(X) = X^2 + X + 1$, and $f(X) = X^3 + 2X^2 + X \in (\mathbb{Z}/3\mathbb{Z})[X]$. The polynomials, $q(X) = X + 1$ and $r(X) = 2X + 2$ are unique and satisfies the property, $\deg r(X) < \deg g(X)$.

An important observation based on the above result, which is often called the *division algorithm* on polynomials, is that if $b \in \mathbb{K}$ is the root of a polynomial $f(X) \in \mathbb{K}[X]$, then $(X-b)$ divides the polynomial $f(X)$. It can be followed quite easily, as we have by polynomial division of $f(X)$ by $(X-b)$, polynomials $q(X)$ and $r(X)$, st. $\deg(r(X)) < \deg(X-b) = 1$ and $f(X) = (X-b)q(X) + r(X)$. Thus we have $r(X)$ a constant, and we denote it by $r \in \mathbb{K}$.

Substituting, $X = b$, since b is a root of $f(X)$ we have $f(b) = r = 0$. This shows that the remainder constant is zero, and thus $(X - b)$ divides the polynomial $f(X)$.

The above result shows that *modular arithmetic* can be defined over the ring of polynomials, in the same way as it can be done over integers.

Definition 1.7.1 Let $a(x), b(x)$, and $f(x)$ be polynomials in $\mathbb{R}[X]$. Then $a(x)$ and $b(x)$ are **congruent modulo** $m(x)$ if $a(x) - b(x) = m(x)k(x)$ for some polynomial $k(x) \in \mathbb{R}[X]$. We write this as:

$$a(x) \equiv b(x) \pmod{m(x)}$$

Likewise for two polynomials $f(X)$ and $g(X) \in \mathbb{K}[X]$, we can define the greatest common divisor of the two polynomials. However, the greatest common divisor (gcd) of two non-zero polynomials implicitly refers to the monic polynomial. This makes the gcd of two polynomials unique. From *Theorem 7*, we have $f(X) \equiv r(X) \pmod{g(X)}$. This gives rise to the following result, which can be used to determine the gcd of two polynomials and, later as we shall see, to compute the inverse of a polynomial defined in a field.

Theorem 8 Let $f(X), g(X) \in \mathbb{K}[X]$, both of which are not zero. The (monic) gcd of $f(X)$ and $g(X)$ is denoted by $d(X)$. Then, there are polynomials $u(X)$ and $v(X) \in \mathbb{K}[X]$, st. $d(X) = u(X)f(X) + v(X)g(X)$. Further, if $f(X)$ and $g(X)$ are non-zero and not both constants, then $u(X)$ and $v(X)$ can be chosen st. $\text{degree}(u(X)) < \text{degree}(g(X))$ and $\text{degree}(v(X)) < \text{degree}(f(X))$.

This theorem forms the basis of the famous Euclidean algorithm for computing the greatest common divisors for two polynomials. An extension of the algorithm, referred to as the Extended Euclidean algorithm is used for computing the inverse of a polynomial defined in a field.

1.8 Construction of Galois Field

Fields with a finite number of elements are called *Finite Fields*, often called *Galois Fields* and abbreviated as **GF**. We know that when p is prime, the residue class $\mathbb{Z}/p\mathbb{Z}$ is a field. This field is often abbreviated as $GF(p)$, and commonly referred to as the *prime field*. The characteristic of this field is p . When $p = 2$, this field $GF(2)$ is called the binary field and is popular for fast and efficient implementations.

The field $GF(p)$ is often extended to a larger field, which has p^n elements for any positive integer n . The field is represented by the symbol $GF(p^n)$. The construction of the extension field is detailed underneath and is due to *Kronecker*.

Let p be a prime number and let f be a polynomial in $\mathbb{Z}/p\mathbb{Z}$, of degree n , where n is a positive integer. The polynomial is irreducible, implying that it cannot be factored into polynomials, g and h which are polynomials in $(\mathbb{Z}/p\mathbb{Z})[X]$ of positive degree, i.e., neither of the polynomials is a constant. As an analogy, the irreducible polynomials can be imagined to correspond to prime numbers in the domain of integers.

Example 14 The polynomial $f(X) = X^2 + X + 1$ is irreducible in $(\mathbb{Z}/2\mathbb{Z})[X]$. This can be checked easily, as if the polynomial is reducible (not irreducible), one can factor $f(X) = g(X)h(X)$, and both $g(X)$ and $h(X)$ are degree one polynomials in $(\mathbb{Z}/2\mathbb{Z})[X]$. This follows from the fact, that since $g(X), h(X) \neq 0$, $\text{degree}(f(X)) = \text{degree}(h(X)) + \text{degree}(g(X))$. Thus, $f(X)$ must have a zero in $(\mathbb{Z}/2\mathbb{Z})[X]$, i.e., either 0 or 1 is a zero or root of $f(X)$. But both $f(0)$ and $f(1)$ are non-zero. Hence, the polynomial $f(X)$ is irreducible.

The finite field is constructed much similar to what we do in the context of modular arithmetic. We define residue classes modulo $f(X)$, ie. we generate the set of polynomials modulo $f(X)$ and place them in separate classes. Thus the set consists of all polynomials of $\text{degree} < \text{degree}(f(X))$. Each of these polynomials has representative elements of all the polynomials in the corresponding residue class. The residue class represented by the polynomial $h(X)$ is denoted as:

$$g(X) + f(\mathbb{Z}/p\mathbb{Z})[X] = \{g(X) + h(X)f(X) : h(X) \in (\mathbb{Z}/p\mathbb{Z})[X]\}$$

In other words, the polynomials $g(X)$ and the elements of the residue class are *congruent* modulo $f(X)$. It is easy to see that the representative elements, denoted by $(\mathbb{Z}/p\mathbb{Z})[X]/\langle f(X) \rangle$ form a ring under the standard operations of addition and multiplications. However, they form a field if and only if the polynomial $f(X)$ is irreducible.

Below we state a theorem which states that the above fact.

Theorem 9 *For a non-constant polynomials $f(X) \in (\mathbb{Z}/p\mathbb{Z})[X]$, the ring $(\mathbb{Z}/p\mathbb{Z})[X]/\langle f(X) \rangle$ is a field if and only if $f(X)$ is irreducible in $(\mathbb{Z}/p\mathbb{Z})[X]$.*

The proof is quite straightforward. If, $f(X)$ is reducible over $(\mathbb{Z}/p\mathbb{Z})[X]$, we have $g(X), h(X)$, st. $f(X) = g(X)h(X)$ and $1 \leq \text{degree}(g(X)), \text{degree}(h(X)) < \text{degree}(f(X))$.

Then both $g(X)$ and $h(X)$ are non-zero elements in $(\mathbb{Z}/p\mathbb{Z})[X]$ whose product is zero modulo $f(X)$. Thus, the ring $(\mathbb{Z}/p\mathbb{Z})[X]$ contains non-zero zero divisors.

If $f(X)$ is irreducible over $(\mathbb{Z}/p\mathbb{Z})[X]$ and $g(X)$ is a non-zero polynomial, st. $\text{degree}(g(X)) < \text{degree}(f(X))$, then $\text{gcd}(f(X), g(X)) = 1$. Thus from Euclidean algorithm \exists polynomials $u(X), v(X) \in (\mathbb{Z}/p\mathbb{Z})[X]$, $u(X)f(X) + v(X)g(X) = 1$, and $\text{degree}(v(X)) < \text{degree}(f(X))$.

Thus, $v(X)g(X) \equiv 1(\text{mod}(f(X)))$, i.e., $g(X)$ has a multiplicative inverse in the ring $(\mathbb{Z}/p\mathbb{Z})[X]/\langle f(X) \rangle$, which hence qualifies as a field.

1.9 Extensions of Fields

The above idea of extending the field $GF(p)$ to $GF(p^n)$ can be generalized. Consider a field K and $f(X)$ irreducible over K . Define, $L = K[X]/\langle f(X) \rangle$ is a field extension. We denote this by stating $K \subseteq L$.

We use $f(X)$ to construct the congruence class modulo $f(X)$. Let θ be the equivalence class of the polynomial X in L . It is also denoted as $[X]$. Clearly, $f(\theta) = 0$.

Example 15 *Consider the field $GF(2^2)$, as an extension field of $GF(2)$. Thus define $GF(2^2)[X] = GF(2)[X]/\langle f(X) \rangle$, where $f(X) = X^2 + X + 1$ is an irreducible polynomial in $GF(2)[X]$. It is clear that the polynomials are of the form $aX + b$, where $a, b \in GF(2)$.*

The four equivalence classes are $0, 1, X, X + 1$, which are obtained by reducing the polynomial of $GF(2)[X]$ by the irreducible polynomial $X^2 + X + 1$.

If θ denotes the equivalence class of the polynomial $X \in GF(2^2)$, the classes can be represented as $0, 1, \theta, \theta + 1$.

Notice, that setting $\theta^2 = \theta + 1$, (ie. $f(\theta) = \theta^2 + \theta + 1 = 0$), reduces any polynomial $f(\theta) \in GF(2)[\theta]$ modulo $(\theta^2 + \theta + 1)$.

If $f(X)$ is not irreducible, then $f(X)$ has a factor $f_1(X) \in K[X]$, st. $1 \leq \deg(f_1(X)) < \deg(f(X))$. Either, $f_1(X)$ is irreducible, or $f_1(X)$ has a factor $f_2(X)$, such that $1 \leq$

$\deg(f_2(X)) < \deg(f_1(X))$. Eventually, we will have an irreducible polynomial $q(X)$, which can be used to define the extension field. Even then, $f(\theta) = 0$, because $q(X)$ is a factor of $f(X)$ in $K[X]$.

The number of elements in a field is defined as the *order*. Thus if the order of the field K is p , the elements of the extension field, $K' = K/\langle f(X) \rangle$, where $f(X)$ is an irreducible polynomial of degree m , can be represented as: $a(X) = a_0 + a_1(X) + \dots + a_{m-1}X^{m-1}$.

Since there are exactly p choices for each of the coefficients, there are p^m values in the field. Thus the order of the extension field is p^m .

Thus summarizing every non-constant polynomial over a field has a root in some extension field.

Theorem 10 *Let $f(X)$ be a non-constant polynomial with coefficients in a field K . Then there is a field L containing K that also contains a root of $f(X)$.*

We can apply the above result repeatedly to obtain further extensions of a given field, and finally arrive at a bigger field. Consider a field K with order p . Let us start with the polynomial $f(X) = X^{p^n} - X$ over a field K . It can be extended to K^1 , where a root θ_1 of f lies. In turn the field K^1 can be extended to the field K^2 , where a root θ_2 lies. Thus continuing we can write:

$$X^{p^n} - X = (X - \theta_1) \dots (X - \theta_{p^n})$$

The set of roots, $\theta_1, \theta_2, \dots, \theta_{p^n}$ itself forms a field and are called a splitting field of $f(X)$. In other words, a splitting field of a polynomial with coefficients in a field is the smallest field extension of that field over which the polynomial splits or decomposes into linear factors.

Next we introduce another important class of polynomials which are called minimal polynomials.

Definition 1.9.1 *Let $K \subseteq L$ be a field extension, and θ an element of L . The minimal polynomial of θ over K is the monic polynomial $m(X) \in K[X]$ of smallest degree st. $m(\theta) = 0$.*

The minimal polynomial divides any polynomial that has θ as a root. It is easy to follow, since dividing $f(X)$ by $m(X)$ gives the quotient $q(X)$ and the remainder $r(X)$, st. $\deg(r(X)) < \deg(m(X))$. Thus we have, $f(X) = q(X)m(X) + r(X)$. Substituting, θ we have $f(\theta) = r(\theta) = 0$, since $f(\theta) = 0$. Since, $\deg(r(X)) < \deg(m(X))$ and $m(X)$ is the minimal polynomial, we have $r(X) = 0$. Thus, the minimal polynomial $m(X)$ of θ divides any polynomial which has θ as its root.

1.10 Cyclic Groups of Group Elements

The concept of cyclic groups is central to the study of cryptographic algorithms. The theory of cyclic groups gives an alternative representation of the group elements, which can be often handy for performing certain operations.

If G is a group wrt. a binary operation indicated by simple juxtaposition. G is said to be cyclic if there exists an element $g \in G$, st. every element of G can be written as g^m for some integer m . The elements in the group are enumerated as $\{g^0, g^1, \dots, g^r, g^{r+1}, \dots\}$. The convention is $g^{-m} = (g^{-1})^m$, and $g^0 = 1$.

The non-zero elements of a field form a commutative group under multiplication. The group is called the multiplicative group of the field F , and is denoted by F^* .

Let a be a finite element of a finite group G , and consider the list of powers of a , $\{a^1, a^2, \dots\}$. As G is finite the list will eventually have duplicates, so there are positive integers $j < k$, st. $a^j = a^k$.

Thus, we have $1 = a^{k-j}$ (multiplying both sides by $(a^{-1})^j$).

So, \exists a positive integer $t = k - j$, st. $a^t = 1$. The smallest such positive integer is called the order of a , and is denoted by $ord(a)$. Thus, whenever we have an integer n , st. $a^n = 1$, we have $ord(a)$ divides n .

The subset $S = \{a, a^2, \dots, a^{ord(a)} = 1\}$ is itself a group and thus qualifies as a subgroup. If G is a finite group, order of the group G is defined as the number of elements in G .

Let S be a subgroup of a finite group G . For each element $a \in G$, the set $aS = \{as | s \in S\}$ has the same number of elements as S . If $a, b \in G$, and $a \neq b$, then aS and bS are either disjoint or equal. Thus the group G can be partitioned into m units, denoted by a_1S, a_2S, \dots, a_mS , so that $G = a_1S \cup \dots \cup a_mS$, and $a_iS \cap a_jS = \emptyset, \forall i \neq j$.

Thus, we have $ord(G) = m \times ord(S)$. Hence, the order of a subgroup divides the order of the group G .

Thus we have the following theorem known as **Lagrange's Theorem**.

Theorem 11 *If S is a subgroup of the finite group G , then the order of S divides the order of G .*

In particular, there is an element α such that every non-zero element can be written in the form of α^k . Such an element α is called the **generator** of the multiplicative group, and is often referred to as the **primitive element**.

Specifically consider the field $GF(p^n)$, where p is a prime number. The primitive element of the field is defined as follows:

Definition 1.10.1 *A generator of the multiplicative group of $GF(p^n)$ is called a **primitive element**.*

The minimal polynomial of the primitive element is given a special name, **primitive polynomials**.

Definition 1.10.2 *A polynomial of degree n over $GF(p)$ is a **primitive polynomial** if it is the minimal polynomial of a primitive element in $GF(p^n)$.*

The concepts of irreducibility, minimality and primitivity of polynomials play a central role in the theory of fields. It can be seen that there are several interesting interrelations and properties of these concepts. We state a few in the following sequel:

Theorem 12 *The minimal polynomial of an element of $GF(p^n)$ is irreducible.*

Further, a minimal polynomial over $GF(p)$ of any primitive element of $GF(p^n)$ is an irreducible polynomial of degree n . Thus, a primitive polynomial is irreducible, but not the vice versa. A primitive polynomial must have a non-zero constant term, for otherwise it will be divisible by x . Over the field $GF(2)$, $x + 1$ is a primitive polynomial and all other primitive polynomials have odd number of terms, since any polynomial mod 2 with an even number of terms is divisible by $x + 1$.

If $f(X)$ is an irreducible polynomial of degree n over $GF(p)$, then $f(X)$ divides $g(X) = X^{p^n} - X$. The argument for the above is from the fact, that from the theory of extension of fields, there is a field of order p^n that contains an element θ st. $f(\theta) = 0$. Since, $f(X)$ is irreducible it should be a minimal polynomial as well. Also, the polynomial $g(X) = X^{p^n} - X$ vanishes at $X = \theta$. Thus, $g(X) = f(X)q(X) + r(X)$, where $degree(r(X)) < degree(f(X))$, then $r(\theta) = 0$. Since, $f(X)$ is a minimal polynomial of θ , $r(X) = 0$.

Thus, we have an alternative definition of primitive polynomials, which are nothing but minimal polynomials of the primitive polynomials.

Definition 1.10.3 An irreducible polynomial of degree n , $f(X)$ over $GF(p)$ for prime p , is a primitive polynomial if the smallest positive integer m such that $f(X)$ divides $x^m - 1$ is $m = p^n - 1$.

Over $GF(p^n)$ there are exactly $\phi(p^n - 1)/n$ primitive polynomials, where ϕ is Euler's Totient function. The roots of a primitive polynomial all have order $p^n - 1$. Thus the roots of a primitive polynomial can be used to generate and represent the elements of a field.

We conclude this section, with the comment that all fields of the order p^n are essentially the same. Hence, we can define isomorphism between the elements of the field. We explain a specific case in the context of binary fields in a following section.

1.11 Efficient Galois Fields

In cryptography, *finite fields* play an important role. A finite field is also known as *Galois field* and is denoted by $GF(p^m)$. Here, p is a prime called the *characteristic* of the field, while m is a positive integer. The *order* of the finite field, that is, the number of elements in the field is p^m . When $m = 1$, the resulting field is called a *prime field* and contains the *residue classes* modulo p [122].

In cryptography, two of the most studied fields are finite fields of characteristic two and prime fields. Finite fields of characteristic two, denoted by $GF(2^m)$, is also known as *binary extension finite fields* or simply *binary finite fields*. They have several advantages when compared to prime fields. Most important is the fact that modern computer systems are built on the binary number system. With m bits all possible elements of $GF(2^m)$ can be represented. This is not possible with prime fields (with $p \neq 2$). For example a $GF(2^2)$ field would require 2 bits for representation and use all possible numbers generated by the 2 bits. A $GF(3)$ field would also require 2 bits for representing the three elements in the field. This leaves one of the four possible numbers generated by 2 bits unused leading to an inefficient representation. Another advantage of binary extension fields is the simple hardware required for computation of some of the commonly used arithmetic operations such as addition and squaring. Addition in binary extension fields can be easily performed by a simple *XOR*. There is no carry generated. Squaring in this field is a linear operation and can also be done using *XOR* circuits. These circuits are much simpler than the addition and squaring circuits of a $GF(p)$ field.

1.11.1 Binary Finite Fields

A polynomial of the form $a(x) = a_m x^m + a_{m-1} x^{m-1} + \cdots + a_1 x + a_0$ is said to be a *polynomial over $GF(2)$* if the coefficients $a_m, a_{m-1}, \dots, a_1, a_0$ are in $GF(2)$. Further, the polynomial is said to be *irreducible* over $GF(2)$ if $a(x)$ is divisible only by c or by $c \cdot a(x)$ where $c \in GF(2)$ [286]. An irreducible polynomial of degree m with coefficients in $GF(2)$ can be used to construct the extension field $GF(2^m)$. All elements of the extension field can be represented by polynomials of degree $m - 1$ over $GF(2)$.

Binary finite fields are generally represented using two types of bases. These are the *polynomial* and *normal base* representations.

Definition 1.11.1 Let $p(x)$ be an irreducible polynomial over $GF(2^m)$ and let α be the root of $p(x)$. Then the set

$$\{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\}$$

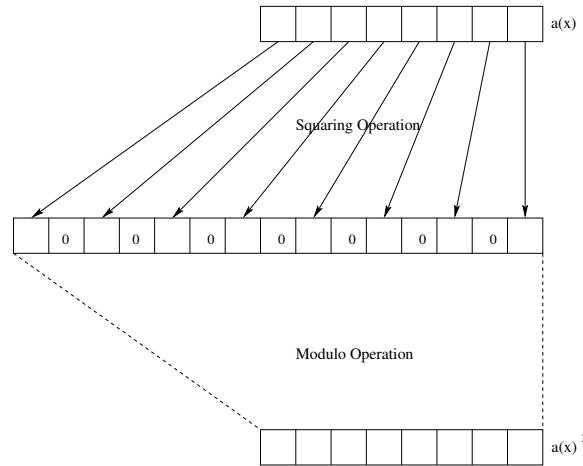


FIGURE 1.2: Squaring Circuit

is called the polynomial base.

Definition 1.11.2 Let $p(x)$ be an irreducible polynomial over $GF(2^m)$, and let α be the root of $p(x)$, then the set

$$\{\alpha, \alpha^2, \alpha^{2^2}, \dots, \alpha^{2^{(m-1)}}\}$$

is called the normal base if the m elements are linearly independent.

The normal bases representation is useful for arithmetic circuits, as squaring an element is accomplished by cyclic shifts. More generally, for any field $GF(p^m)$, the basis vector is $\{b^{p^0}, b^{p^1}, \dots, b^{p^{m-1}}\}$, where b is chosen such that they are linearly independent.

Any element in the field $GF(2^m)$ can be represented in terms of its bases as shown below.

$$a(x) = a_{m-1}\alpha^{m-1} + \dots + a_1\alpha + a_0$$

Alternatively, the element $a(x)$ can be represented as a binary string $(a_{m-1}, \dots, a_1, a_0)$ making it suited for representation on computer systems. For example, the polynomial $x^4 + x^3 + x + 1$ in the field $GF(2^8)$ is represented as $(00011011)_2$.

Various arithmetic operations such as addition, subtraction, multiplication, squaring and inversion are carried out on binary fields. *Addition* and *subtraction* operations are identical and are performed by *XOR* operations.

Let $a(x), b(x) \in GF(2^m)$ be denoted by

$$a(x) = \sum_{i=0}^{m-1} a_i x^i \quad b(x) = \sum_{i=0}^{m-1} b_i x^i$$

then the *addition* (or *subtraction*) of $a(x)$ and $b(x)$ is given by

$$a(x) + b(x) = \sum_{i=0}^{m-1} (a_i + b_i)x^i \tag{1.6}$$

here the $+$ between a_i and b_i denotes a *XOR* operation.

The *squaring* operation on binary finite fields is as easy as addition. The square of the polynomial $a(x) \in GF(2^m)$ is given by

$$a(x)^2 = \sum_{i=0}^{m-1} a_i x^{2i} \bmod p(x) \quad (1.7)$$

The squaring essentially spreads out the input bits by inserting zeroes in between two bits as shown in **Fig. 1.2**.

Multiplication is not as trivial as addition or squaring. The product of the two polynomials $a(x)$ and $b(x)$ is given by

$$a(x) \cdot b(x) = \left(\sum_{i=0}^{n-1} b(x)a_i x^i \right) \bmod p(x) \quad (1.8)$$

Most multiplication algorithms are $O(n^2)$.

Inversion is the most complex of all field operations. Even the best technique to implement inversion is several times more complex than multiplication. Hence, algorithms which use finite field arithmetic generally try to reduce the number of inversions at the cost of increasing the number of multiplications.

The multiplication and squaring operation require a *modular operation* to be done. The modular operation is the remainder produced when divided by the field's irreducible polynomial. If a certain class of irreducible polynomials is used, the modular operation can be easily done. Consider the irreducible trinomial $x^m + x^n + 1$, having a root α and

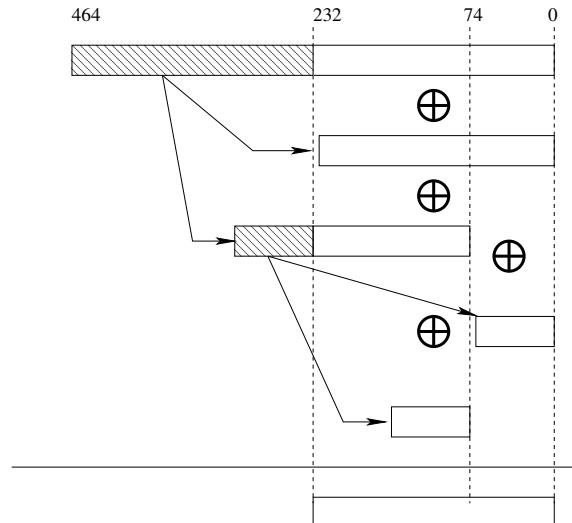


FIGURE 1.3: Modular Reduction with Trinomial $x^{233} + x^{74} + 1$

$1 < n < m/2$. Therefore, $\alpha^m + \alpha^n + 1 = 0$. Therefore,

$$\begin{aligned}\alpha^m &= 1 + \alpha^n \\ \alpha^{m+1} &= \alpha + \alpha^{n+1} \\ &\vdots \\ \alpha^{2m-3} &= \alpha^{m-3} + \alpha^{m+n-3} \\ \alpha^{2m-2} &= \alpha^{m-2} + \alpha^{m+n-2}\end{aligned}\tag{1.9}$$

For example, consider the irreducible trinomial $x^{233} + x^{74} + 1$. The multiplication or squaring of the polynomial results in a polynomial of degree at most 464. This can be reduced as shown in **Fig. 1.3**. The higher-order terms 233 to 464 are reduced by using Equation 1.9.

1.12 Mapping between Binary and Composite Fields

In this section, we define two extension fields of $GF(2)$, one of them called as the composite fields.

Definition 1.12.1 *The pair of the fields $GF(2^n)$ and $GF(2^n)^m$ are called a composite field, if there exists irreducible polynomials, $Q(Y)$ of degree n and $P(X)$ of degree m , which are used to extend $GF(2)$ to $GF(2^n)$, and $GF((2^n)^m)$ from $GF(2^n)$.*

Composite fields are denoted by $GF(2^n)^m$. A composite field is isomorphic to the field, $GF(2^k)$, where $k = m \times n$. However, it is interesting to note that the underlying field operations in both the fields have different complexity, and varies with the exact values of n, m and the polynomials used to construct the fields.

Below we provide an example.

Example 16 Consider the fields $GF(2^4)$, elements of which are the following 16 polynomials with binary coefficients:

$$\begin{array}{cccc}0 & z^2 & z^3 & z^3 + z^2 \\ 1 & z^2 + 1 & z^3 + 1 & z^3 + z^2 + 1 \\ z & z^2 + z & z^3 + z & z^3 + z^2 + z \\ z + 1 & z^2 + z + 1 & z^3 + z + 1 & z^3 + z^2 + z + 1\end{array}$$

There are 3 irreducible polynomials of degree 4, which can be used to construct the fields: $f_1(z) = z^4 + z + 1$, $f_2(z) = z^4 + z^3 + 1$, $f_3(z) = z^4 + z^3 + z^2 + z + 1$.

The resulting fields, F_1, F_2, F_3 all have the same elements, ie. the above 16 polynomials.

However, the operations are different: like the same operation, $z \cdot z^3$ would result in $z + 1, z^3 + 1, z^3 + z^2 + z + 1$ in the three fields F_1, F_2 , and F_3 , respectively.

The fields are isomorphic and one can establish between the fields, say F_1 and F_2 a mapping, by computing $c \in F_2$, st. $f_1(c) \equiv 0 \pmod{f_2}$. The map $z \rightarrow c$ is thus used to construct an isomorphism $\mathbb{T} : F_1 \rightarrow F_2$.

The choices of c are $z^2 + z$, $z^2 + z + 1$, $z^3 + z^2$, and $z^3 + z^2 + 1$. One can verify that $c = z^2 + z \Rightarrow f_1(c) = (z^2 + z)^4 + (z^2 + z) + 1 = z^8 + z^4 + z^2 + z + 1 \equiv 0 \pmod{f_2}$. The modulo f_2 can be performed by substituting, z^4 by $z^3 + 1$, ie. $f_2(z) = 0$.

The homomorphism property of the mapping wrt. to the multiplication operation can easily be observed. Mathematically we check that, $T(e_1 \times e_2 \text{mod } f_1) \equiv (T(e_1) \times T(e_2)) \text{mod } f_2$

Let us consider two elements, $e_1 = z^2 + z$ and $e_2 = z^3 + z$. The product of the elements in the field F_1 are: $(z^2 + z)(z^3 + z) \text{mod } (z^4 + z + 1) = z^3 + 1$. The above reduction, uses the fact that in the field E_1 $f_1(z) = 0 \Rightarrow z^4 = z + 1$.

The same operation can also be performed in the field, F_2 . That is $T(e_1) = (z^2 + z)^2 + (z^2 + z) \text{mod } (z^4 + z^3 + 1) = z^3 + z + 1$. This uses the fact that the mapping, \mathbb{T} maps, $z \in F_1$ to $z^2 + z \in F_2$. Reduction modulo f_2 is performed by setting $f_2(z) = 0 \Rightarrow z^4 = z^3 + 1$.

Likewise, $\mathbb{T}(e_2) = (z^2 + z)^3 + (z^2 + z) = z + 1$.

Multiplying the two mapped elements results in $(z+1)(z^3+z+1) \equiv z^2 \text{mod } (z^4+z^3=1)$. This can be seen as same as $\mathbb{T}(z^3+1)$.

We present next an algorithm based on this same idea to construct an isomorphism between the composite fields.

1.12.1 Constructing Isomorphisms between Composite Fields

Let the polynomial used to construct, $GF(2^n)$ be denoted by $Q(Y)$ and the polynomial used to construct the polynomial $GF(2^n)^m$ by $P(X)$. Assuming both the polynomials as primitive, the roots of $Q(Y)$ and $P(X)$, denoted as ω and α can be used to generate the fields. The field elements of $GF(2^n)$ are thus represented by $\{0, 1, \omega, \omega^2, \dots, \omega^{2^n-2}\}$, while the elements of the composite field are represented by $\{0, 1, \alpha, \alpha^2, \dots, \alpha^{2^{nm}-2}\}$.

Arithmetic in the field $GF(2^k)$, $k = nm$ can be performed by modulo the polynomial $R(z) = z^k + r_{k-1}z^{k-1} + \dots + 1$, $r_i \in GF(2)$. If γ is a root of the polynomial $R(z)$, $B_2 = (\gamma^{k-1}, \gamma^{k-2}, \dots, \gamma, 1)$ is the standard basis with which the elements in $GF(2^k)$ are represented. Each element of the field can thus be visualized as a binary vector of dimension, k , and each element can be expressed as a linear combination of the basis elements.

The elements of the composite field $GF(2^n)^m$ are likewise represented as m -bit vectors, where each element is an element of $GF(2^n)$. The operations are performed modulo the two field generator polynomials $Q(Y) = Y^n + q_{n-1}Y^{n-1} + \dots + q_1Y + 1$ and $P(X) = X^m + p_{m-1}X^{m-1} + \dots + p_1X + p_0$.

The primitive elements of the field $GF(2^n)^m$ is denoted by α , while that of $GF(2^k)$ is denoted by γ .

We first present the basic idea for the conversion between the two fields in the following section.

1.12.1.1 Mapping from $GF(2^k)$ to $GF(2^n)^m$, where $k = nm$

A simple method to obtain such a conversion is to find the primitive element of both the fields, $GF(2^k)$ and $GF(2^n)^m$. The primitive elements are denoted by γ and α respectively. One checks that $R(\gamma)$ and $R(\alpha)$ are both zero, and thus we establish the mapping $GF(2^k) \rightarrow GF(2^n)^m : \gamma \rightarrow \alpha$. If the roots do not satisfy the polynomial R , we find the next primitive element and repeat the test.

The subsequent mappings are easy to obtain, by raising the respective primitive elements to their power and establishing the mappings. Thus, the mappings are obtained as: $GF(2^k) \rightarrow GF(2^n)^m : \gamma^i \rightarrow \alpha^i$, for $0 \leq i \leq 2^k - 2$, where $R(\alpha) \equiv 0 \text{mod } Q(Y), P(X)$. It may be noted that $R(\gamma) \equiv 0 \text{mod } R(Z)$, as γ is primitive.

The above is stated in algorithm 1.3.

Example 17 Consider $GF(2^4)$ with primitive polynomial $R(Z) = Z^4 + Z + 1$, $GF(2^2)$ with primitive polynomials $Q(Y) = Y^2 + Y + 1$, and $GF(2^2)^2$ with primitive polynomials $P(X) = X^2 + X + \{2\}$, where $\{2\} \in GF(2^2)$.

Algorithm 1.3: Determining Composite Field Mapping Using Primitive Roots

Input: $n, m, Q(Y), P(X), R(Z)$
Output: Mapping: $GF(2^k) \rightarrow GF(2^n)^m$, where $k = nm$

```

1 Find Primitive Element of  $GF(2^k)$ : denoted as  $\gamma$ 
2 for ( $\alpha = 1; \alpha < 2^{nm} - 1;$ ) do
3   if IsPrimitive( $\alpha$ ) &  $R(\alpha)$  then
4     break;
5   end
6 end
7 for ( $i = 0; i < 2^{nm} - 1; i++$ ) do
8    $a_1 = \alpha^i \bmod P(X), Q(Y)$ 
9    $b_1 = \gamma^i \bmod R(Z)$ 
10  Map:  $a_1 \rightarrow b_1$ 
11 end

```

$GF(2^4) \rightarrow GF(2^2)^2$	$GF(2^4) \rightarrow GF(2^2)^2$
$\{02\} \rightarrow \{04\}$	$\{04\} \rightarrow \{06\}$
$\{08\} \rightarrow \{0e\}$	$\{03\} \rightarrow \{05\}$
$\{06\} \rightarrow \{02\}$	$\{0c\} \rightarrow \{08\}$
$\{0b\} \rightarrow \{0b\}$	$\{05\} \rightarrow \{07\}$
$\{0a\} \rightarrow \{0a\}$	$\{07\} \rightarrow \{03\}$
$\{0e\} \rightarrow \{0c\}$	$\{0f\} \rightarrow \{0d\}$
$\{0d\} \rightarrow \{09\}$	$\{09\} \rightarrow \{0f\}$
$\{01\} \rightarrow \{01\}$	$\{00\} \rightarrow \{00\}$

TABLE 1.6: An Example Isomorphic Mapping between the field $GF(2^4)$ and $GF(2^2)^2$

The first primitive element $\gamma \in GF(2^4)$ is 2. It can be checked that raising higher powers of 2, modulo $Z^4 + Z + 1$ all the non-zero elements of $GF(2^4)$ can be generated. Likewise, the first primitive element of $GF(2^2)^2$, such that $R(Z) \equiv 0$ modulo $Q(Y)$ and $P(X)$ is 4. Hence, we establish the mapping $\{02\} \rightarrow \{04\}$.

The complete mapping obtained by raising the above elements to their higher powers is written in Table 1.6. It may be noted that for completeness, we specify that 0 is mapped to 0 in the table.

The above algorithm can be made more efficient, by using suitable tests for primitivity and also storages. One such algorithm is presented in the following subsection.

1.12.1.2 An Efficient Conversion Algorithm

Next we present an algorithm to determine the mapping between a binary to a composite field representation. That is the algorithm takes two fields, $GF(2^k)$ and $GF(2^n)^m$, with $k = nm$ and returns a binary matrix of dimension $k \times k$, denoted by T , which performs an isomorphic mapping from the field $GF(2^k)$ to $GF(2^n)^m$. The inverse matrix, T^{-1} can be used to perform the mapping in the reverse direction.

Thus in order to construct the mapping you need to develop the relationship between the k elements of the two fields, $GF(2^k)$ and $GF(2^n)^m$, where $k = mn$.

The unity element in $GF(2^k)$ is mapped to the unity element in the composite field. The primitive element, γ is mapped to the element α^t , the base element γ^2 is mapped to

α^{2t} . Thus continuing similarly we have:

$$\mathbb{T}\gamma^i = \alpha^{it}, i = 0, 1, \dots, k-1$$

It may be noted that the choice of t cannot be arbitrary; it has to be done such that the homomorphism is established. wrt. addition and multiplications. For this we use the property discussed before:

$$R(\alpha^t) = 0, \text{ mod } Q(Y), P(X) \quad (1.10)$$

There will be exactly k primitive elements which will satisfy the condition, namely α^t and α^{t2^j} , $j = 1, 2, \dots, k-1$, where the exponents are computed modulo $2^k - 1$.

We summarize the above in algorithm 1.4.

Algorithm 1.4: Determining Composite Field Mappings, Isomorphic Mapping

Input: $n, m, Q(Y), P(X), R(Z)$
Output: Transformation \mathbb{T}

```

1  $\alpha$  is the primitive element in  $GF(2^n)^m$  for which  $P(\alpha) = 0$ .
2  $t = 1$ 
3 Initialize a list  $S[1 : 2^k - 1]$  with  $2^k - 1$  addresses and one bit amount of information
   is stored in each address location.
4 Initialize a  $k \times k$  matrix  $\mathbb{T}$  with each column being indicated by  $\mathbb{T}[i]$ , where  $1 \leq i \leq k$ .
5 Set,  $\mathbb{T}[k] = (0, 0, \dots, 1)$ .
6 while ( $R(\alpha^t) \neq 0$ ) do
7   for ( $j = 0; j \leq k-1; j++$ ) do
8      $S[t2^j \bmod (2^k - 1)] = 0$ 
9   end
10   $t = t + 1$ 
11  while ( $S[t] == 0$  or  $\gcd(t, 2^k - 1) > 1$ ) do
12     $t = t + 1$ 
13  end
14 end
15 for ( $j = 2; j \leq k; j++$ ) do
16    $T[j] = \text{binary}(\alpha^{(j-1)t})$ 
17 end
```

The algorithm can be explained as follows: Line 5 of the algorithm ensures that the identity element in the field $GF(2^k)$ is mapped to the identity element in $GF(2^n)^m$. Both the identity elements are represented by the polynomial 1. Line 6 checks for the equality of $R(\alpha^t)$ to zero, which if true indicates that t is found. If α^t is not the element to which β is mapped, then α^{t2^j} are also not primitive elements, where $1 < j < k-1$. Hence we set the corresponding elements in the array, S to 0, and proceed by incrementing t by 1.

In line 10 and 11, we continue the search for the appropriate t by checking whether the corresponding entry in the S array has been set to 0 (indicating it was found unsuitable during a previous run of the while loop of line 6), or if not previously marked by checking the primitivity of α^t by computing the gcd of t and $2^k - 1$. If the gcd is found to be greater than 1, it indicates that α^t is not primitive, hence, we increment t .

When the correct value of t is obtained, the matrix T is populated columnwise from the right by the binary representations of α^{jt} , where $2 \leq j \leq k$.

1.13 Conclusions

In this chapter, we developed several mathematical concepts, which form the foundations of modern cryptography. The chapter presented discussions on modular arithmetic and defined the concepts of mathematical groups, rings, and fields. Useful operations like the Euclidean algorithm, its extensions to evaluate the greatest common divisor, and the multiplicative inverse were elaborated. We also discussed the Chinese Remainder Theorem, which is a useful tool to develop efficient designs for RSA-like algorithms and also to perform attacks on them. The chapter subsequently developed the important concept of subfields and shows how to construct extension fields from a field. As modern cryptographic algorithm relies heavily on Galois (finite) fields, the chapter presents a special attention to them, efficient representation of elements in the Galois fields, and their various properties, like formation of cyclic group, etc. The chapter also elaborated with examples on how to define isomorphic mapping between several equivalent fields, often technically referred to as the composite fields. All these concepts and techniques built on them have useful impact in efficient (hardware) designs and attacks. We show several such applications in the following chapters.

Bibliography

- [1] *39th International Symposium on Computer Architecture (ISCA 2012), June 9-13, 2012, Portland, OR, USA.* IEEE, 2012.
- [2] Evolutionary Computation. Wikipedia, 2012. http://en.wikipedia.org/wiki/Evolutionary_computation.
- [3] N Mazzocca A. Cilardo, L Coppolino and L Romano. Elliptic Curve Cryptography Engineering. *Proceedings of the IEEE*, 94(2):395–406, Feb 2006.
- [4] A. Rudra, P. K. Dubey, C. S. Jutla et al. Efficient Implementation of Rijndael Encryption with Composite Field Arithmetic. In *CHES*, pages 171–184, CHES 2001:Paris, France, May 14-16 2001. Springer.
- [5] J. Aarestad, P. Ortiz, D. Acharyya, and J. Plusquellic. HELP: A Hardware-Embedded Delay PUF. *IEEE Design and Test of Computers*, 30(2):17–25, 2013.
- [6] Masayuki Abe, editor. *Topics in Cryptology - CT-RSA 2007, The Cryptographers' Track at the RSA Conference 2007, San Francisco, CA, USA, February 5-9, 2007, Proceedings*, volume 4377 of *Lecture Notes in Computer Science*. Springer, 2006.
- [7] M. Abramovici and P. L. Levin. Protecting integrated circuits from silicon Trojan horses. *Military Embedded Systems*, 2009. <http://www.mil-embedded.com/articles/id/?3748>.
- [8] Onur Aciicmez. Yet another MicroArchitectural Attack: : exploiting I-Cache. In Peng Ning and Vijay Atluri, editors, *CSAW*, pages 11–18. ACM, 2007.
- [9] Onur Aciicmez, Billy Bob Brumley, and Philipp Grabher. New Results on Instruction Cache Attacks. In Mangard and Standaert [244], pages 110–124.
- [10] Onur Aciicmez and Çetin Kaya Koç. Trace-Driven Cache Attacks on AES (Short Paper). In Peng Ning, Sihan Qing, and Ninghui Li, editors, *ICICS*, volume 4307 of *Lecture Notes in Computer Science*, pages 112–121. Springer, 2006.
- [11] Onur Aciicmez, Çetin Kaya Koç, and Jean-Pierre Seifert. On the Power of Simple Branch Prediction Analysis. *IACR Cryptology ePrint Archive*, 2006:351, 2006.
- [12] Onur Aciicmez, Çetin Kaya Koç, and Jean-Pierre Seifert. Predicting secret keys via branch prediction. In Abe [6], pages 225–242.
- [13] Onur Aciicmez, Shay Gueron, and Jean-Pierre Seifert. New Branch Prediction Vulnerabilities in OpenSSL and Necessary Software Countermeasures. In Steven D. Galbraith, editor, *IMA Int. Conf.*, volume 4887 of *Lecture Notes in Computer Science*, pages 185–203. Springer, 2007.

- [14] Onur Acıicmez and Werner Schindler. A Vulnerability in RSA Implementations Due to Instruction Cache Analysis and Its Demonstration on OpenSSL. In Tal Malkin, editor, *CT-RSA*, volume 4964 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2008.
- [15] Onur Acıicmez, Werner Schindler, and Çetin Kaya Koç. Cache Based Remote Timing Attack on the AES. In Abe [6], pages 271–286.
- [16] Onur Acıicmez, Jean-Pierre Seifert, and Çetin Kaya Koç. Micro-Architectural Cryptanalysis. *IEEE Security & Privacy*, 5(4):62–64, 2007.
- [17] S. Adee. The hunt for the kill switch. *IEEE Spectrum*, 45(5):34–39, May 2008.
- [18] Aes (Rijndael) ip-cores. http://www.erst.ch/download/aes_standard_cores.pdf, 2011.
- [19] Full datasheet aes-ccm core family for actel fpga. http://www.actel.com/ipdocs/HelionCore_AES-CCM_8bit_Actel_DS.pdf, 2011.
- [20] M. Agarwal, B. Zhang Paul, M., and S. Mitra. Circuit failure prediction and its application to transistor aging. In *VTS'07: Proceedings of the IEEE VLSI Test Symposium*, pages 277–286, 2007.
- [21] Michel Agoyan, Jean-Max Dutertre, Amir-Pasha Mirbaha, David Naccache, Anne-Lise Ribotta, and Assia Tria. How to Flip a Bit? pages 235–239. IOLTS, Jul 2010.
- [22] Michel Agoyan, Jean-Max Dutertre, David Naccache, Bruno Robisson, and Assia Tria. When Clocks Fail: On Critical Paths and Clock Faults. pages 182–193. CARDIS, 2010.
- [23] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar. Trojan detection using IC Fingerprinting. In *Proc. IEEE Symposium on Security and Privacy*, pages 296–310, Washington, DC, USA, 2007.
- [24] Gregory C. Ahlquist, Brent E. Nelson, and Michael Rice. Optimal Finite Field Multipliers for FPGAs. In *FPL '99: Proceedings of the 9th International Workshop on Field-Programmable Logic and Applications*, pages 51–60, London, UK, 1999. Springer-Verlag.
- [25] Monjur Alam, Sonai Ray, Debdeep Mukhopadhyay, Santosh Ghosh, Dipanwita Roy Chowdhury, and Indranil Sengupta. An area optimized reconfigurable encryptor for aes-rijndael. In *DATE*, pages 1116–1121, 2007.
- [26] Subidh Ali and Debdeep Mukhopadhyay. A Differential Fault Analysis on AES Key Schedule Using Single Fault. In Breveglieri et al. [63], pages 35–42.
- [27] Subidh Ali and Debdeep Mukhopadhyay. An Improved Differential Fault Analysis on AES-256. In Abderrahmane Nitaj and David Pointcheval, editors, *AFRICACRYPT*, volume 6737 of *Lecture Notes in Computer Science*, pages 332–347. Springer, 2011.
- [28] Subidh Ali, Debdeep Mukhopadhyay, and Michael Tunstall. Differential Fault Analysis of AES using a Single Multiple-Byte Fault. Cryptology ePrint Archive, Report 2010/636, 2010. <http://eprint.iacr.org/>.
- [29] Y. M. Alkabani and F. Koushanfar. Active hardware metering for intellectual property protection and security. In *SS'07: Proceedings of USENIX Security Symposium*, pages 20:1–20:16, 2007.

- [30] Y. M. Alkabani, F. Koushanfar, and M. Potkonjak. Remote activation of ICs for piracy prevention and digital right management. In *ICCAD '07: Proceedings of the International Conference on CAD*, pages 674–677, 2007.
- [31] M. E. Amyeen, S. Venkataraman, A. Ojha, and S. Lee. Evaluation of the quality of N-detect scan ATPG patterns on a processor. In *ITC'04: Proceedings of the International Test Conference*, pages 669–678, 2004.
- [32] D. Anastasakis, R. Damiano, Hi-Keung T. Ma, and T. Stanion. A practical and efficient method for compare-point matching. In *DAC'02: Proceedings of the Design Automation Conference*, pages 305–310, 2002.
- [33] B. Ansari and M.A. Hasan. High-performance architecture of elliptic curve scalar multiplication. *Computers, IEEE Transactions on*, 57(11):1443 –1453, November 2008.
- [34] Frederik Armknecht, Roel Maes, Ahmad-Reza Sadeghi, Francois-Xavier Standaert, and Christian Wachsmann. A Formal Foundation for the Security Features of Physical Functions. *IEEE Security and Privacy*, 2011(1):16, 2011.
- [35] D. Aucsmith. Tamper resistant software: An implementation. In *IH'96: Proceedings of the International Workshop on Information Hiding*, pages 317–333, 1996.
- [36] Australian Government DoD-DSTO. Towards countering the rise of the silicon trojan. <http://dspace.dsto.defence.gov.au/dspace/bitstream/1947/9736/1/DSTO-TR-2220\%20PR.pdf>, 2008.
- [37] R. Azarderakhsh and A. Reyhani-Masoleh. Efficient fpga implementations of point multiplication on binary edwards and generalized hessian curves using gaussian normal basis. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, PP(99):1, 2011.
- [38] M. Banga and M. S. Hsiao. A region based approach for the identification of hardware Trojans. In *Proc. IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, pages 40–47, Washington, DC, USA, 2008.
- [39] M. Banga and M. S. Hsiao. A novel sustained vector technique for the detection of hardware Trojans. In *VLSID'09: Proceedings of the International Conference on VLSI Design*, pages 327–332, January 2009.
- [40] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S.P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO '01: Proceedings of the International Cryptology Conference on Advances in Cryptology*, pages 1–18, 2001.
- [41] Alessandro Barenghi, Cédric Hocquet, David Bol, François-Xavier Standaert, Francesco Regazzoni, and Israel Koren. Exploring the Feasibility of Low Cost Fault Injection Attacks on Sub-Threshold Devices through An Example of A 65nm AES Implementation. pages 48–60. in Proc. Workshop RFID Security Privacy, 2011.
- [42] Alberto Battistello and Christophe Giraud. Fault Analysis of Infective AES Computations. In Wieland Fischer and Jörn-Marc Schmidt, editors, *Fault Diagnosis and Tolerance in Cryptography – FDTC 2013*, pages 101–107. IEEE Computer Society, 2013.

- [43] M. Bednara, M. Daldrup, J. von zur Gathen, J. Shokrollahi, and J. Teich. Reconfigurable Implementation of Elliptic Curve Crypto Algorithms. In *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM*, pages 157–164, 2002.
- [44] Daniel J. Bernstein. Cache-timing Attacks on AES. Technical report, 2005.
- [45] Guido Bertoni, Luca Breveglieri, Israel Koren, Paolo Maistri, and Vincenzo Piuri. Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard. *IEEE Trans. Computers*, 52(4):492–505, 2003.
- [46] Guido Bertoni, Vittorio Zaccaria, Luca Breveglieri, Matteo Monchiero, and Gianluca Palermo. AES Power Attack Based on Induced Cache Miss and Countermeasure. In *ITCC (1)*, pages 586–591. IEEE Computer Society, 2005.
- [47] Régis Bevan and Erik Knudsen. Ways to Enhance Differential Power Analysis. In *Proceedings of Information Security and Cryptology (ICISC 2002), LNCS Volume 2587*, pages 327–342. Springer-Verlag, 2002.
- [48] Eli Biham. A Fast New DES Implementation in Software. In *FSE* [49], pages 260–272.
- [49] Eli Biham, editor. *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*. Springer, 1997.
- [50] Eli Biham and Adi Shamir. Differential Fault Analysis of Secret Key Cryptosystems. *Proceedings of Eurocrypt, Lecture Notes in Computer Science*, 1233:37–51, 1997.
- [51] Alex Biryukov and David Wagner. Slide attacks. In *FSE*, pages 245–259, 1999.
- [52] Johannes Blömer, Jorge Guajardo, and Volker Krummel. Provably secure masking of aes. In *Proceedings of the 11th international conference on Selected Areas in Cryptography*, SAC'04, pages 69–83, Berlin, Heidelberg, 2005. Springer-Verlag.
- [53] Johannes Blömer and Volker Krummel. Analysis of Countermeasures Against Access Driven Cache Attacks on AES. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2007.
- [54] Johannes Blömer and Jean-Pierre Seifert. Fault Based Cryptanalysis of the Advanced Encryption Standard (AES). In *Financial Cryptography*, pages 162–181, 2003.
- [55] Kaijje Wu Bo Yang and R. Karri. Secure scan: A design-for-test architecture for crypto-chips. In *DAC'05: Proceedings of 42nd Design Automation Conference*, pages 135–140, 2005.
- [56] Andrey Bogdanov, Thomas Eisenbarth, Christof Paar, and Malte Wienecke. Differential Cache-Collision Timing Attacks on AES with Applications to Embedded CPUs. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 235–251. Springer, 2010.
- [57] B. Bollig and I. Wegener. Improving the Variable Ordering of OBDDs is NP-Complete. *IEEE Transactions on Computers*, 45:993–1002, September 1996.
- [58] Joseph Bonneau and Ilya Mironov. Cache-Collision Timing Attacks Against AES. In Louis Goubin and Mitsuru Matsui, editors, *CHES*, volume 4249 of *Lecture Notes in Computer Science*, pages 201–215. Springer, 2006.

- [59] Giacomo Boracchi and Luca Breveglieri. A Study on the Efficiency of Differential Power Analysis on AES S-Box. *Technical Report*, January 15, 2007.
- [60] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *DAC'03: Proceedings of the Design Automation Conference*, pages 338–342, 2003.
- [61] Christoph Bösch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient Helper Data Key Extractor on FPGAs. In *Cryptographic Hardware and Embedded Systems (CHES)*, volume 5154 of *Lecture Notes in Computer Science*, pages 181–197. 2008.
- [62] X. Boyen. Reusable cryptographic fuzzy extractors. In *Proc. of the 10th ACM conference on Computer and Communications*, pages 82–91, 2004.
- [63] Luca Breveglieri, Sylvain Guilley, Israel Koren, David Naccache, and Junko Takahashi, editors. *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2011, Tokyo, Japan, September 29, 2011*. IEEE, 2011.
- [64] Ernie Brickell, Gary Graunke, Michael Neve, and Jean-Pierre Seifert. Software Mitigations to Hedge AES Against Cache-based Software Side Channel Vulnerabilities. *Cryptology ePrint Archive*, Report 2006/052, 2006.
- [65] Billy Bob Brumley and Nicola Tuveri. Remote Timing Attacks are Still Practical. In Vijay Atluri and Claudia Díaz, editors, *ESORICS*, volume 6879 of *Lecture Notes in Computer Science*, pages 355–371. Springer, 2011.
- [66] David Brumley and Dan Boneh. Remote Timing Attacks are Practical. *Computer Networks*, 48(5):701–716, 2005.
- [67] R.E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 35:677–691, August 1986.
- [68] Maciej Brzozowski and Vyacheslav N. Yarmolik. Obfuscation as intellectual rights protection in VHDL language. In *Proceedings of the 6th International Conference on Computer Information Systems and Industrial Management Applications*, pages 337–340, Washington, DC, 2007. IEEE Computer Society.
- [69] C. Rebeiro, S. S. Roy, D. S. Reddy and D. Mukhopadhyay. Revisiting the Itoh Tsujii Inversion Algorithm for FPGA Platforms. *IEEE Transactions on VLSI Systems.*, 19(8):1508–1512, 2011.
- [70] Methodology for protection and licensing of HDL IP. <http://www.us.design-reuse.com/news/?id=12745&print=yes>, 2011.
- [71] David Canright. A very compact s-box for aes. In *CHES*, pages 441–455, 2005.
- [72] Anne Canteaut, Cédric Lauradoux, and André Seznec. Understanding Cache Attacks. Research Report RR-5881, INRIA, 2006.
- [73] E. Castillo, U. Meyer-Baese, A. García, L. Parrilla, and A. Lloris. IPP@HDL: efficient intellectual property protection scheme for IP cores. *IEEE Transactions on VLSI*, 15:578–591, May 2007.
- [74] Ç. K. Koç and B. Sunar. An Efficient Optimal Normal Basis Type II Multiplier. *IEEE Trans. Comput.*, 50(1):83–87, 2001.

- [75] Çetin K. Koç and Tolga Acar. Montgomery Multiplication in $GF(2^k)$. *DES Codes Cryptography*, 14(1):57–69, 1998.
- [76] Çetin Kaya Koç. *Cryptographic Engineering*. Springer, 2009.
- [77] R. S. Chakraborty and S. Bhunia. HARPOON: a SoC design methodology for hardware protection through netlist level obfuscation. *IEEE Transactions on CAD*, 28(10):1493–1502, October 2009.
- [78] R. S. Chakraborty and S. Bhunia. Hardware protection and authentication through netlist level obfuscation. In *ICCAD'08: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 674–677, 2009.
- [79] R. S. Chakraborty and S. Bhunia. Security against hardware Trojan through a novel application of design obfuscation. In *ICCAD '09: Proceedings of the International Conference on CAD*, pages 113–116, 2009.
- [80] R. S. Chakraborty and S. Bhunia. RTL hardware IP protection using key-based control and data flow obfuscation. In *VLSID'10: Proceedings of the International Conference on VLSI Design*, pages 405–410, 2010.
- [81] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia. MERO: a statistical approach for hardware Trojan detection using logic testing. *Lecture Notes on Computer Science*, 5737:396–410, September 2009.
- [82] R.S. Chakraborty and S. Bhunia. Security through obscurity: An approach for protecting Register Transfer Level hardware IP. In *HOST'08: Proceedings of the International Workshop on Hardware Oriented Security and Trust*, pages 96–99, 2009.
- [83] R.S. Chakraborty and S. Bhunia. RTL hardware IP protection using key-based control and data flow obfuscation. In *VLSID '10: Proceedings of the International Conference on VLSI Design*, pages 405–410, 2010.
- [84] H. Chang and M.J. Atallah. Protecting software code by guards. In *DRM '01: Revised Papers from the ACM CCS-8 Workshop on Security and Privacy in Digital Rights Management*, pages 160–175, 2002.
- [85] E. Charbon and I. Torunoglu. Watermarking techniques for electronic circuit design. In *IWDW'02: Proceedings of the International Conference on Digital Watermarking*, pages 147–169, 2003.
- [86] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.
- [87] W. N. Chelton and M. Benaissa. Fast Elliptic Curve Cryptography on FPGA. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 16(2):198–205, February 2008.
- [88] Chien-Ning Chen and Sung-Ming Yen. Differential fault analysis on AES key schedule and some countermeasures. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, *ACISP 2003*, volume 2727 of *LNCS*, pages 118–129. Springer, 2003.
- [89] Deming Chen, Jason Cong, and Peichen Pan. FPGA Design Automation: A Survey. *Found. Trends Electron. Des. Autom.*, 1(3):139–169, 2006.

- [90] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. R ührmair. The Bistable Ring PUF: A new architecture for strong physical unclonable functions. In *Proc. of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 134 –141, 2011.
- [91] Z. Chen, X. Guo, R. Nagesh, A. Reddy, M. Gora, and A. Maiti. Hardware Trojan designs on BASYS FPGA board, 2012. <http://isis.poly.edu/~vikram/vt.pdf>.
- [92] Z. Cherif, J. Danger, S. Guilley, and L. Bossuet. An easy-to-design PUF based on a single oscillator: The Loop PUF. In *Proc. of 15th Euromicro Conference on Digital System Design (DSD)*, pages 156–162, 2012.
- [93] T. Chou and K. Roy. Accurate power estimation of CMOS sequential circuits. *IEEE Transactions on VLSI*, 4(3):369–380, September 1996.
- [94] Christophe Clavier, Jean-Sébastien Coron, and Nora Dabbous. Differential power analysis in the presence of hardware countermeasures. In Çetin Kaya Koç and Christof Paar, editors, *CHES*, volume 1965 of *Lecture Notes in Computer Science*, pages 252–263. Springer, 2000.
- [95] Obfuscation by code morphing. http://en.wikipedia.org/wiki/Obfuscated_code#Obfuscation_by_code_morphing, 2011.
- [96] C. Collberg, C. Thomborson, and D. Low. Manufacturing cheap, resilient, and stealthy opaque constructs. In *POPL ’98: Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 184–196, 1998.
- [97] C.S. Collberg and C. Thomborson. Watermarking, tamper-proofing, and obfuscation: tools for software protection. *IEEE Transactions on Software Engineering*, 28:735–746, August 2002.
- [98] Bart Coppens, Ingrid Verbauwhede, Koen De Bosschere, and Bjorn De Sutter. Practical Mitigations for Timing-Based Side-Channel Attacks on Modern x86 Processors. In *IEEE Symposium on Security and Privacy*, pages 45–60. IEEE Computer Society, 2009.
- [99] Jean-Sébastien Coron and Ilya Kizhvatov. An Efficient Method for Random Delay Generation in Embedded Software. In Christophe Clavier and Kris Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 2009.
- [100] Jean-Sébastien Coron and Ilya Kizhvatov. Analysis and Improvement of the Random Delay Countermeasure of CHES 2009. In Mangard and Standaert [244], pages 95–109.
- [101] Scott A. Crosby, Dan S. Wallach, and Rudolf H. Riedi. Opportunities and Limits of Remote Timing Attacks. *ACM Trans. Inf. Syst. Secur.*, 12(3), 2009.
- [102] F. Crowe, A. Daly, and W. Marnane. Optimised Montgomery Domain Inversion on FPGA. In *Circuit Theory and Design, 2005. Proceedings of the 2005 European Conference on*, volume 1, August/September 2005.
- [103] J. Daemen and V. Rijmen. *The Design of Rijndael*. Springer-Verlag, 2002.
- [104] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.

- [105] Ds2432 1kb protected 1-wire eeprom with sha-1 engine. <http://www.maxim-ic.com/datasheet/index.mvp/id/2914>, 2012.
- [106] Ds5002fp secure microprocessor chip. <http://www.maxim-ic.com/datasheet/index.mvp/id/2949>, 2012.
- [107] DARPA. TRUST in Integrated Circuits (TIC) - Proposer Information Pamphlet. <http://www.darpa.mil/MT0/solicitations/baa07-24/index.html>, 2007.
- [108] Guerick Meurice de Dormale, Philippe Bulens, and Jean-Jacques Quisquater. An Improved Montgomery Modular Inversion Targeted for Efficient Implementation on FPGA. In O. Diessel and J.A. Williams, editors, *International Conference on Field-Programmable Technology - FPT 2004*, pages 441–444, 2004.
- [109] Defense Science Board. Task force on high performance microchip supply. <http://www.acq.osd.mil/dsb/reports/200502HPMSReportFinal.pdf>, 2005.
- [110] John Demme, Robert Martin, Adam Waksman, and Simha Sethumadhavan. Side-Channel Vulnerability Factor: A Metric for Measuring Information Leakage. In *ISCA* [1], pages 106–117.
- [111] W. Diffie and M. Hellman. New Directions in Cryptography. In *IEEE Transactions on Information Theory* (22), pages 644–654. IEEE, 1976.
- [112] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In C. Cachin and J.L. Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. 2004.
- [113] Leonid Domnitser, Aamer Jaleel, Jason Loew, Nael B. Abu-Ghazaleh, and Dmitry Ponomarev. Non-monopolizable caches: Low-complexity Mitigation of Cache Side-Channel Attacks. *TACO*, 8(4):35, 2012.
- [114] D. Du, S. Narasimhan, R. S. Chakraborty, and S. Bhunia. Self-referencing: a scalable side-channel approach for hardware Trojan detection. In *Proc. of the International Workshop on Cryptographic Hardware and Embedded Systems (CHES'11)*, pages 173–187, Berlin, Heidelberg, 2010.
- [115] Zoya Dyka and Peter Langendoerfer. Area Efficient Hardware Implementation of Elliptic Curve Cryptography by Iteratively Applying Karatsuba’s Method. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 70–75, Washington, DC, USA, 2005. IEEE Computer Society.
- [116] Z. Chen *et al.* Hardware Trojan Designs on BASYS FPGA Board. CSAW Embedded Systems Challenge, 2008. <http://isis.poly.edu/~vikram/vt.pdf>.
- [117] Federal Information Processing Standards Publication 197. Announcing the Advanced Encryption Standard (AES), 2001.
- [118] Federal Information Processing Standards Publication 46-2. Announcing the Standard for Data Encryption Standard (DES), 1993.
- [119] H. Feistel. Cryptography and Computer Privacy. *Scientific American*, 228(5):15–23, May 1973.
- [120] M. Feldhofer, J. Wolkerstorfer, and V. Rijmen. Aes implementation on a grain of sand. *Information Security, IEE Proceedings*, 152(1):13–20, 2005.

- [121] Jacques J. A. Fournier and Michael Tunstall. Cache Based Power Analysis Attacks on AES. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *ACISP*, volume 4058 of *Lecture Notes in Computer Science*, pages 17–28. Springer, 2006.
- [122] John B. Fraleigh. *First Course in Abstract Algebra*. Addison-Wesley, Boston, MA, USA, 2002.
- [123] W. F. Friedman. The index of coincidence and its application in cryptography. In *Riverbank Publication, Riverbank Labs*. Reprinted by Aegian Park Press, 1920.
- [124] Hideo Fujiwara and Marie Engelene J. Obien. Secure and testable scan design using extended de bruijn graphs. In *ASPDAC 10: Proceedings of the 2010 Asia and South Pacific Design Automation Conference*, pages 413–418, 2010.
- [125] Katsuya Fujiwara, Hideo Fujiwara, Marie Engelene J. Obien, and Hideo Tamamoto. Sreep: Shift register equivalents enumeration and synthesis program for secure scan design. *Design and Diagnostics of Electronic Circuits and Systems*, 0:193–196, 2010.
- [126] G. Piret and J. J. Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad. In *CHES 2003*, pages 77–88. LNCS 2779, 2003.
- [127] Jean-François Gallais, Ilya Kizhvatov, and Michael Tunstall. Improved Trace-Driven Cache-Collision Attacks against Embedded AES Implementations. In Yongwha Chung and Moti Yung, editors, *WISA*, volume 6513 of *Lecture Notes in Computer Science*, pages 243–257. Springer, 2010.
- [128] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *Proc. of ACM Conference on Computer and Communications Security*, pages 148–160, 2002.
- [129] Blaise Gassend, Daihyun Lim, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Identification and authentication of integrated circuits: Research Articles. *Concurrency and Computation: Practice & Experience*, 16(11):1077–1098, 2004.
- [130] M. J. Geuzebroek, J. Th. van der Linden, and A. J. van de Goor. Test point insertion that facilitates ATPG in reducing test time and data volume. In *ITC'02: Proceedings of the International Test Conference*, pages 138–147, 2002.
- [131] Benedikt Gierlichs, Jörn-Marc Schmidt, and Michael Tunstall. Infective Computation and Dummy Rounds: Fault Protection for Block Ciphers without Check-before-Output. In Alejandro Hevia and Gregory Neven, editors, *Progress in Cryptology – LATINCRYPT 2012*, volume 7533 of *Lecture Notes in Computer Science*, pages 305–321. Springer, 2012.
- [132] Christophe Giraud. DFA on AES. In *IACR e-print archive 2003/008*, page 008. <http://eprint.iacr.org/2003/008>, 2003.
- [133] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.
- [134] O. Goldreich. *Foundations of Cryptography*, volume 2. Cambridge University Press, 2005.
- [135] Oded Goldreich and Rafail Ostrovsky. Software Protection and Simulation on Oblivious RAMs. *J. ACM*, 43(3):431–473, 1996.

- [136] C. Grabbe, M. Bednara, J. Shokrollahi, J. Teich, and J. von zur Gathen. FPGA Designs of Parallel High Performance $GF(2^{233})$ Multipliers. In *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS-03)*, volume II, pages 268–271, Bangkok, Thailand, May 2003.
- [137] Johann Großschädl and Guy-Armand Kamendje. Instruction Set Extension for Fast Elliptic Curve Cryptography over Binary Finite Fields $GF(2^m)$. In *ASAP*, pages 455–. IEEE Computer Society, 2003.
- [138] Johann Großschädl and Erkay Savas. Instruction Set Extensions for Fast Arithmetic in Finite Fields $GF(p)$ and $GF(2^m)$. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 133–147. Springer, 2004.
- [139] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls. FPGA intrinsic PUFs and their use for IP protection. In *Proc. of Cryptographic Hardware and Embedded Systems Workshop (CHES)*, volume 4727 of *LNCS*, pages 63–80, 2007.
- [140] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls. Physical unclonable functions and public-key crypto for FPGA IP protection. In *Field Programmable Logic and Applications*, pages 189–195, August 2007.
- [141] Jorge Guajardo and Christof Paar. Itoh-Tsujii Inversion in Standard Basis and Its Application in Cryptography and Codes. *Des. Codes Cryptography*, 25(2):207–216, 2002.
- [142] David Gullasch, Endre Bangerter, and Stephan Krenn. Cache Games - Bringing Access-Based Cache Attacks on AES to Practice. In *IEEE Symposium on Security and Privacy*, pages 490–505. IEEE Computer Society, 2011.
- [143] Xiaofei Guo and R. Karri. Invariance-based Concurrent Error Detection for Advanced Encryption Standard. In *DAC*, pages 573–578, Jun 2012.
- [144] Nils Gura, Sheueling Chang Shantz, Hans Eberle, Sumit Gupta, Vipul Gupta, Daniel Finchelstein, Edouard Goupy, and Douglas Stebila. An End-to-End Systems Approach to Elliptic Curve Cryptography. In *CHES '02: Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, pages 349–365, London, UK, 2003. Springer-Verlag.
- [145] Torben Hagerup and C. Rüb. A guided tour of Chernoff bounds. In *Information Processing Letters*, Volume 33, Issue 6, pages 305–308. Elsevier North-Holland, Inc., 1990.
- [146] Shikha Bisht Harshal Tupsamudre and Debdeep Mukhopadhyay. Destroying fault invariant with randomization - a countermeasure for aes against differential fault attacks. In *CHES*, 2014.
- [147] Ryan Helinski, Dhruva Acharyya, and Jim Plusquellic. A physical unclonable function defined using power distribution system equivalent resistance variations. In *Proc. of 46th Annual Design Automation Conference (DAC)*, pages 676–681, 2009.
- [148] David Hely, Maurin Augagneur, Yves Clauzel, and Jeremy Dubeuf. A physical unclonable function based on setup time violation. In *Proc. of IEEE 30th International Conference on Computer Design (ICCD)*, pages 135–138, 2012.

- [149] David Hely, Marie-Lise Flottes, Frederic Bancel, Bruno Rouzeyre, Nicolas Berard, and Michel Renovell. Scan design and secure chip. In *IOLTS '04: Proceedings of the International On-Line Testing Symposium, 10th IEEE*, page 219, Washington, DC, USA, 2004. IEEE Computer Society.
- [150] Alireza Hodjat and Ingrid Verbauwhede. Area-throughput trade-offs for fully pipelined 30 to 70 gbits/s aes processors. *IEEE Trans. Comput.*, 55(4):366–372, April 2006.
- [151] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh. Quantitative and Statistical Performance Evaluation of Arbiter Physical Unclonable Functions on FPGAs. In *Proceedings of International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, pages 298–303, 2010.
- [152] Takashi Horiyama, Masaki Nakanishi, Hirotugu Kajihara, and Shinji Kimura. Folding of Logic Functions and its Application to Look Up Table Compaction. *ICCAD*, 00:694–697, 2002.
- [153] T.W. Hou, H.Y. Chen, and M.H. Tsai. Three control flow obfuscation methods for Java software. *IEE Proceedings on Software*, 153(2):80–86, April 2006.
- [154] Wei-Ming Hu. Lattice scheduling and covert channels. In *Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium on*, pages 52–61, may 1992.
- [155] Y. L. Huang, F.S. Ho, H.Y. Tsai, and H.M. Kao. A control flow obfuscation method to discourage malicious tampering of software codes. In *ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, pages 362–362, 2006.
- [156] T. Huffmire, B. Brotherton, W. Gang, T. Sherwood, R. Kastner, T. Levin, T. Nguyen, and C. Irvine. Moats and Drawbridges: An isolation primitive for reconfigurable hardware based systems. In *SP '07: Proceedings of the IEEE Symposium on Security and Privacy*, pages 281–295, 2007.
- [157] T. Huffmire, C. Irvine, T. D. sand T. Levin Nguyen, R. Kastner, and T. Sherwood. *Handbook of FPGA Design Security*. Springer, Dordrecht, 2010.
- [158] Michael Hutton, Jay Schleicher, David M. Lewis, Bruce Pedersen, Richard Yuan, Sinan Kaptanoglu, Gregg Baeckler, Boris Ratchev, Ketan Padalia, Mark Bourgeault, Andy Lee, Henry Kim, and Rahul Saini. Improving FPGA Performance and Area Using an Adaptive Logic Module. In *FPL*, pages 135–144, 2004.
- [159] Xilinx Inc. Virtex-II Platform FPGA User Guide (v 2.2). <http://www.xilinx.com/support/documentation/virtex-ii.htm>, 2012.
- [160] Toshiya Itoh and Shigeo Tsujii. A Fast Algorithm For Computing Multiplicative Inverses in $GF(2^m)$ Using Normal Bases. *Inf. Comput.*, 78(3):171–177, 1988.
- [161] C. Patel J. Plusquellic J. Lee, M. Tehranipoor. Securing scan design using lock and key technique. In *DFT 05: Proceedings of 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pages 51–62, 2005.
- [162] M.H. Jakubowski, C.W. Saw, and R. Venkatesan. Tamper-tolerant software: Modeling and implementation. In *IWSEC '09: Proceedings of the International Workshop on Security: Advances in Information and Computer Security*, pages 125–139, 2009.

- [163] K. Järvinen and J. Skytta. On parallelization of high-speed processors for elliptic curve cryptography. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 16(9):1162 –1175, sept. 2008.
- [164] Kimmo Järvinen. On repeated squarings in binary fields. In Michael Jacobson, Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *Selected Areas in Cryptography*, volume 5867 of *Lecture Notes in Computer Science*, pages 331–349. Springer Berlin / Heidelberg, 2009.
- [165] D. Jayasinghe, J. Fernando, R. Herath, and R. Ragel. Remote Cache Timing Attack on Advanced Encryption Standard and Countermeasures. In *Information and Automation for Sustainability (ICIAFs), 2010 5th International Conference on*, pages 177 –182, dec. 2010.
- [166] Marie-Lise Flottes Jean Da Rolt, Giorgio Di Natale and Bruno Rouzeyre. New security threats against chips containing scan chain structures. In *HOST 11: Proceedings of IEEE Symposium on Hardware-Oriented Security and Trust*, pages 105–110, 2011.
- [167] Y. Jin and Y. Makris. Hardware Trojan detection using path delay fingerprint. In *Proc. IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, pages 51–57, Washington, DC, USA, 2008.
- [168] H.G. Joepgen and S. Krauss. Software by means of the protprog method. *Elektronik*, 42:52–56, August 1993.
- [169] D.S. Johnson. The NP-completeness column. *ACM Transactions on Algorithms*, 1:160–176, July 2005.
- [170] M. Joye, P. Manet, and JB. Rigaud. Strengthening Hardware AES Implementations against Fault Attack. *IET Information Security*, 1:106–110, 2007.
- [171] D. Kahn. The codebreakers: The story of secret writing. New York: Macmillan Publishing Co, 1967.
- [172] A.B. Kahng, J. Lach, W.H. Mangione-Smith, S. Mantik, I.L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe. Constraint-based watermarking techniques for design IP protection. *IEEE Transactions on CAD*, 20(10):1236 – 1252, October 2001.
- [173] Burton S. Kaliski. The Montgomery Inverse and its Applications. *IEEE Transactions on Computers*, 44(8):1064–1065, 1995.
- [174] R. Kapoor. Security vs. test quality: Are they mutually exclusive? In *ITC '04: Proceedings of the International Test Conference*, page 1413, Washington, DC, USA, 2004. IEEE Computer Society.
- [175] D. Karakoyunlu and B. Sunar. Differential template attacks on PUF enabled cryptographic devices. In *Proceedings of IEEE International Workshop on Information Forensics and Security (WIFS)*, 2010.
- [176] Anatoly A. Karatsuba and Y. Ofman. Multiplication of Multidigit Numbers on Automata. *Soviet Physics Doklady*, 7:595–596, 1963.
- [177] Mark Karpovsky, Konrad J. Kulikowski, and Alexander Taubin. Differential Fault Analysis Attack Resistant Architectures for the Advanced Encryption Standard. In *CARDIS*, pages 177–192, Aug 2004.

- [178] R. Karri, J. Rajendran, Rosenfeld K., and M. Tehranipoor. Trustworthy hardware: identifying and classifying hardware trojans. *IEEE Computer*, 43(10):39–46, Oct. 2010.
- [179] Ramesh Karri, G. Kuznetsov, and M. Goessel. Parity-Based Concurrent Error Detection of Substitution-Permutation Network Block Ciphers. In *CHES*, pages 113–124, Sept 2003.
- [180] Ramesh Karri, Kaijie Wu, P. Mishra, and Y. Kim. Concurrent Error Detection Schemes of Fault Based Side-Channel Cryptanalysis of Symmetric Block Ciphers. *IEEE Trans. on Computer-Aided Design*, 21(12):1509–1517, Dec 2002.
- [181] Ramesh Karri, Kaijie Wu, Piyush Mishra, and Yongkook Kim. Concurrent Error Detection of Fault-Based Side-Channel Cryptanalysis of 128-Bit Symmetric Block Ciphers. In *DAC*, pages 579–585, 2001.
- [182] Ramesh Karri, Kaijie Wu, Piyush Mishra, and Yongkook Kim. Fault-based side-channel cryptanalysis tolerant rijndael symmetric block cipher architecture. In *DFT*, pages 427–435, 2001.
- [183] Michael Kasper, Werner Schindler, and Marc Stöttinger. A stochastic method for security evaluation of cryptographic fpga implementations. In *FPT*, pages 146–153, 2010.
- [184] Stefan Katzenbeisser, ÅIJnal Kocabâ§, Vincent Leest, Ahmad-Reza Sadeghi, Geert-Jan Schrijen, and Christian Wachsmann. Recyclable PUFs: logically reconfigurable PUFs. *Journal of Cryptographic Engineering*, 1:177–186, 2011.
- [185] John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. Side Channel Cryptanalysis of Product Ciphers. In Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann, editors, *ESORICS*, volume 1485 of *Lecture Notes in Computer Science*, pages 97–110. Springer, 1998.
- [186] John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. Side Channel Cryptanalysis of Product Ciphers. *J. Comput. Secur.*, 8(2,3):141–158, 2000.
- [187] Mehran Mozaffari Kermani and Arash Reyhani-Masoleh. Parity Prediction of S-Box for AES. In *CCECE*, pages 2357–2360, 2006.
- [188] Mehran Mozaffari Kermani and Arash Reyhani-Masoleh. A Low-cost S-box for the Advanced Encryption Standard using Normal Basis. In *EIT*, pages 52–55, 2009.
- [189] Mehran Mozaffari Kermani and Arash Reyhani-Masoleh. A High-Performance Fault Diagnosis Approach for the AES SubBytes Utilizing Mixed Bases. In Breveglieri et al. [63], pages 80–87.
- [190] Mehran Mozaffari Kermani and Arash Reyhani-Masoleh. A Low-Power High-Performance Concurrent Fault Detection Approach for the Composite Field S-Box and Inverse S-Box. *IEEE Trans. Computers*, 60(9):1327–1340, 2011.
- [191] A. Keshavarzi, K. Roy, C. F. Hawkins, and V. De. Multiple-parameter CMOS ic testing with increased sensitivity for i_{DDQ} . *IEEE Transactions on VLSI*, 11(5):863–870, may 2003.
- [192] Farouk Khelil, Mohamed Hamdi, Sylvain Guille, Jean Luc Danger, and Nidhal Selmane. Fault Analysis Attack on an AES FPGA Implementation. pages 1–5. ESR-Groups, 2008.

- [193] Sagar Khurana, Souvik Kolay, Chester Rebeiro, and Debdeep Mukhopadhyay. Light Weight Cipher Implementations on Embedded Processors. In *Design and Technology of Integrated Systems (DTIS)*. IEEE Computer Society, 2013.
- [194] C. Kim. Improved Differential Fault Analysis on AES Key Schedule. *Information Forensics and Security, IEEE Transactions on*, PP(99):1, 2011.
- [195] Chang Hoon Kim, Soonhak Kwon, and Chun Pyo Hong. FPGA Implementation of High Performance Elliptic Curve Cryptographic processor over $GF(2^{163})$. *Journal of Systems Architecture - Embedded Systems Design*, 54(10):893–900, 2008.
- [196] Chong Hee Kim. Differential fault analysis against AES-192 and AES-256 with minimal faults. In Luca Breveglieri, Marc Joye, Israel Koren, David Naccache, and Ingrid Verbauwhede, editors, *Fault Diagnosis and Tolerance in Cryptography — FDTC 2010*, pages 3–9. IEEE Computer Society, 2010.
- [197] Chong Hee KIM. Differential fault analysis of aes: Toward reducing number of faults. Cryptology ePrint Archive, Report 2011/178, 2011. <http://eprint.iacr.org/>.
- [198] Chong Hee Kim and Jean-Jacques Quisquater. New Differential Fault Analysis on AES Key Schedule: Two Faults Are Enough. In Gilles Grimaud and François-Xavier Standaert, editors, *CARDIS*, volume 5189 of *Lecture Notes in Computer Science*, pages 48–60. Springer, 2008.
- [199] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou. Designing and implementing malicious hardware. In *LEET’08: Proceedings of the Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pages 5:1–5:8, 2008.
- [200] Alexander Klimov and Adi Shamir. A New Class of Invertible Mappings. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 470–483. Springer, 2002.
- [201] Donald E. Knuth. *The Art of Computer Programming Volumes 1-3 Boxed Set*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1998.
- [202] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *CRYPTO ’96: Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113, London, UK, 1996. Springer-Verlag.
- [203] D. A. Koley. An introduction to Genetic Algorithms for scientists and engineers. In *Action Report*, pages 42–47. Western Governors Association, Jun. 1995.
- [204] Jingfei Kong, Onur Aciicmez, Jean-Pierre Seifert, and Huiyang Zhou. Deconstructing New Cache Designs for Thwarting Software Cache-based Side Channel Attacks. In Trent Jaeger, editor, *CSAW*, pages 25–34. ACM, 2008.
- [205] Jingfei Kong, Onur Aciicmez, Jean-Pierre Seifert, and Huiyang Zhou. Hardware-Software Integrated Approaches to Defend Against Software Cache-Based Side Channel Attacks. In *HPCA*, pages 393–404. IEEE Computer Society, 2009.
- [206] Jingfei Kong, Onur Aciicmez, Jean-Pierre Seifert, and Huiyang Zhou. Architecting Against Software Cache-based Side Channel Attacks. *IEEE Transactions on Computers*, 99(PrePrints), 2012.

- [207] I. Koren and C. Mani Krishna. *Fault-Tolerant Systems*. Morgan-Kaufmann, 2007.
- [208] F. Koushanfar. Provably secure active IC metering techniques for piracy avoidance and digital rights management. *IEEE Transactions on Information Forensics and Security*, 7(1):51–63, February 2012.
- [209] J. R. Koza. A hierarchical approach to learning the Boolean Multiplexer function. In *FOGA ’91: Proceedings of the Workshop on the Foundations of Genetic Algorithms and Classifier Systems*, pages 171–192, 1991.
- [210] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
- [211] Aswin Krishna, Seetharam Narasimhan, Xinmu Wang, and Swarup Bhunia. MECCA: A Robust Low-Overhead PUF Using Embedded Memory Array. In *Cryptographic Hardware and Embedded Systems (CHES)*, volume 6917 of *Lecture Notes in Computer Science*, pages 407–420. 2011.
- [212] R. Kumar, V.C. Patil, and S. Kundu. Design of Unique and Reliable Physically Unclonable Functions Based on Current Starved Inverter Chain. In *Proc. of IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 224–229, 2011.
- [213] S.S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls. Extended abstract: The butterfly PUF protecting IP on every FPGA. In *Proc. of IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pages 67–70, 2008.
- [214] Klaus Kursawe, Ahmad-Reza Sadeghi, Dries Schellekens, Boris Škorić, and Pim Tuyls. Reconfigurable Physical Unclonable Functions – Enabling Technology for Tamper-Resistant Storage . In *Proc. of 2nd IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pages 22–29, 2009.
- [215] J. Lach, W.H. Mangione-Smith, and M. Potkonjak. Robust FPGA intellectual property protection through multiple small watermarks. In *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, DAC ’99, pages 831–836, New York, NY, 1999. ACM.
- [216] Cédric Lauradoux. Collision Attacks on Processors with Cache and Countermeasures. In Christopher Wolf, Stefan Lucks, and Po-Wah Yau, editors, *WEWoRC*, volume 74 of *LNI*, pages 76–85. GI, 2005.
- [217] Jae W. Lee, Daihyun Lim, Blaise Gassend, G. Edward Suh, Marten van Dijk, and Srinivas Devadas. A technique to build a secret key in integrated circuits for identification and authentication application. In *Proceedings of the Symposium on VLSI Circuits*, pages 176–159, 2004.
- [218] Ruby B. Lee, Zhijie Shi, Yiqun Lisa Yin, Ronald L. Rivest, and Matthew J. B. Robshaw. On Permutation Operations in Cipher Design. In *ITCC (2)*, pages 569–577. IEEE Computer Society, 2004.
- [219] Wei Li, Dawu Gu, Yong Wang, Juanru Li, and Zhiqiang Liu. An Extension of Differential Fault Analysis on AES. In *Third International Conference on Network and System Security*, pages 443–446. NSS, 2009.
- [220] D. Lim. Extracting secret keys from integrated circuits. Master’s thesis, Massachusetts Institute of Technology, 2004.

- [221] L. Lin, W. Burleson, and C. Parr. *MOLES*: Malicious off-chip leakage enabled by side-channels. In *ICCAD'09: Proceedings of the International Conference on CAD*, pages 117–122, 2009.
- [222] L. Lin, M. Kasper, T. GĂjneșu, C. Paar, and W. Burleson. Trojan side-channels: Lightweight Hardware Trojans through side-channel engineering. volume 5747 of *Lecture Notes in Computer Science*, pages 382–395, 2009.
- [223] C. Linn and S. Debray. Obfuscation of executable code to improve resistance to static disassembly. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 290–299, 2003.
- [224] Keith Lofstrom, W. Robert Daasch, and Donald Taylor. IC Identification Circuit Using Device Mismatch. In *Proc. of ISSCC*, pages 372–373, 2000.
- [225] Victor Lomné, Thomas Roche, and Adrian Thillard. On the Need of Randomness in Fault Attack Countermeasures - Application to AES. In Guido Bertoni and Benedikt Gierlichs, editors, *Fault Diagnosis and Tolerance in Cryptography – FDTC 2012*, pages 85–94. IEEE Computer Society, 2012.
- [226] Julio López and Ricardo Dahab. Fast multiplication on elliptic curves over $gf(2^m)$ without precomputation. In *Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '99, pages 316–327, London, UK, UK, 1999. Springer-Verlag.
- [227] Julio López and Ricardo Dahab. Improved Algorithms for Elliptic Curve Arithmetic in $GF(2^n)$. In *SAC '98: Proceedings of the Selected Areas in Cryptography*, pages 201–212, London, UK, 1999. Springer-Verlag.
- [228] Jonathan Lutz and Anwarul Hasan. High Performance FPGA based Elliptic Curve Cryptographic Co-Processor. In *ITCC '04: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 2*, page 486, Washington, DC, USA, 2004. IEEE Computer Society.
- [229] B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. Cryptology ePrint Archive, Report 2004/060, 2004. <http://eprint.iacr.org/>.
- [230] P. Lysaght. Dynamic reconfiguration of Xilinx FPGAs: enhanced architectures, design methodologies, & CAD tools. Xilinx, Inc., 2012. <http://www.xilinx.com/univ/FPL06\Invited\Presentation\PLysaght.pdf>.
- [231] Chinese firms favoring soft IP over hard cores. http://www.eetasia.com/ART_8800440032_480100_NT_ac94df1c.HTM, 2011.
- [232] R. Maes, V. Rozic, I. Verbauwheide, P. Koeberl, E. van der Sluis, and V. van der Leest. Experimental evaluation of Physically Unclonable Functions in 65 nm CMOS. In *Proc. of the ESSCIRC*, pages 486–489, 2012.
- [233] Roel Maes, Pim Tuyls, and Ingrid Verbauwheide. Intrinsic PUFs from Flip-flops on Reconfigurable Devices. In *Proc. of 3rd Benelux Workshop on Information and System Security (WISSec)*, page 17, 2008.
- [234] Roel Maes and Ingrid Verbauwheide. Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions. In Ahmad-Reza Sadeghi and David Naccache, editors, *Towards Hardware-Intrinsic Security*, Information Security and Cryptography, pages 3–37. Springer, 2010.

- [235] V. Maingot and R. Leveugle. Influence of Error Detecting or Correcting Codes on the Sensitivity to DPA of an AES S-Box. In *ICSES*, pages 1–5, 2009.
- [236] P. Maistri and R. Leveugle. Double-Data-Rate Computation as a Countermeasure against Fault Analysis. *IEEE Transactions on Computers*, 57(11):1528–1539, Nov 2008.
- [237] Abhranil Maiti, Vikash Gunreddy, and Patrick Schaumont. A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions. *IACR Cryptology ePrint Archive*, 2011:657, 2011.
- [238] A. Maity and P. Schaumont. Improving the quality of a Physical Unclonable Function using configurable Ring Oscillators. In *FPL'09: International Conference on Field Programmable Logic and Applications*, pages 703–707, 2009.
- [239] Mehrdad Majzoobi, Golsa Ghiaasi, Farinaz Koushanfar, and Sani R. Nassif. Ultra-low power current-based PUF. In *International Symposium on Circuits and Systems (ISCAS)*, pages 2071–2074. 2011.
- [240] Mehrdad Majzoobi, Farinaz Koushanfar, and Miodrag Potkonjak. Lightweight secure PUFs. In *Proc. of the 2008 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 670–673, 2008.
- [241] Stefan Mangard. Hardware Countermeasures against DPA ? A Statistical Analysis of Their Effectiveness. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2004.
- [242] Stefan Mangard, Thomas Popp, and Berndt M. Gammel. Side-Channel Leakage of Masked CMOS Gates. In Alfred Menezes, editor, *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*, Lecture Notes in Computer Science (LNCS), pages 351 – 365. Springer, 2005.
- [243] Stefan Mangard and Kai Schramm. Pinpointing the side-channel leakage of masked aes hardware implementations. In *CHES*, pages 76–90, 2006.
- [244] Stefan Mangard and François-Xavier Standaert, editors. *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*. Springer, 2010.
- [245] Robert Martin, John Demme, and Simha Sethumadhavan. TimeWarp: Rethinking Timekeeping and Performance Monitoring Mechanisms to Mitigate Side-Channel Attacks. In *ISCA* [1], pages 118–129.
- [246] B. Mathew and D. G. Saab. Combining multiple DFT schemes with test generation. *IEEE Transactions on CAD*, 18(6):685–696, 1999.
- [247] S. Mathew, F. Sheikh, A. Agarwal, M. Kounavis, S. Hsu, H. Kaul, M. Anders, and R. Krishnamurthy. 53Gbps Native $GF(2^4)^2$ Composite-Field AES-Encrypt/Decrypt Accelerator for Content-Protection in 45nm High-Performance Microprocessors. In *VLSI Circuits (VLSIC), 2010 IEEE Symposium on*, pages 169–170, June.
- [248] Mitsuru Matsui. New Block Encryption Algorithm MISTY. In Biham [49], pages 54–68.

- [249] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.
- [250] Dominik Merli, Dieter Schuster, Frederic Stumpf, and Georg Sigl. Side-channel analysis of PUFs and Fuzzy extractors. In *Proceedings of the 4th international conference on Trust and trustworthy computing, Pittsburgh, PA, TRUST'11*, pages 33–47, 2011.
- [251] Thomas S. Messerges, Ezzat A. Dabbish, and Robert H. Sloan. Examining Smart-Card Security under the Threat of Power Analysis Attacks. *IEEE Trans. Comput.*, 51(5):541–552, 2002.
- [252] P. L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. In *Mathematics of Computation*, volume 48, pages 243–264, January 1987.
- [253] Peter L. Montgomery. Five, Six, and Seven-Term Karatsuba-Like Formulae. *IEEE Transactions on Computers*, 54(3):362–369, 2005.
- [254] W. A. Moore and P. A. Kayfes. US Patent 7213142 - system and method to initialize registers with an EEPROM stored boot sequence. {<http://www.patentstorm.us/patents/7213142/description.html>}, 2007.
- [255] A. Moradi, A. Barenghi, T. Kasper, and C. Paar. On the vulnerability of FPGA bit-stream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs. In *CCS'11: Proceedings of the ACM Conference on Computer and Communications Security*, pages 111–123, 2011.
- [256] Amir Moradi, Mohammad T. Manzuri Shalmani, and Mahmoud Salmasizadeh. A Generalized Method of Differential Fault Attack against AES Cryptosystem. In *CHES*, pages 91–100, 2006.
- [257] C.J. Morford. BitMaT – Bitstream Manipulation Tool for Xilinx FPGAs. Master’s thesis, Virginia Polytechnic Institute and State University, 2005.
- [258] Mehran Mozaffari-Kermani and Arash Reyhani-Masoleh. Parity-Based Fault Detection Architecture of S-box for Advanced Encryption Standard. In *DFT*, pages 572–580, Oct 2006.
- [259] Mehran Mozaffari-Kermani and Arash Reyhani-Masoleh. A Lightweight Concurrent Error Detection Scheme for the AES S-boxes Using Normal Basis. In *Proc. CHES*, pages 113–129, Aug 2008.
- [260] Mehran Mozaffari-Kermani and Arash Reyhani-Masoleh. A Lightweight High-Performance Fault Detection Scheme for the Advanced Encryption Standard Using Composite Field. *IEEE Trans. VLSI Systems*, 19(1):85–91, 2011.
- [261] Dhiman Saha Mukesh Agarwal, Sandip Karmakar and Debdeep Mukhopadhyay. Scan based side channel attacks on stream ciphers and their counter-measures. In *Indocrypt ’08: Proceedings of Progress in Cryptology-Indocrypt*, LNCS 5365, pages 226–238, 2008.
- [262] D. Mukhopadhyay. An improved fault based attack of the Advanced Encryption Standard. In *AFRICACRYPT’09: Progress in Cryptology*, pages 421–434, 2009.
- [263] D. Mukhopadhyay, S. Banerjee, D. RoyChowdhury, and B. B. Bhattacharya. Crys-toscan: A secured scan chain architecture. In *ATS ’05: Proceedings of the 14th Asian Test Symposium on Asian Test Symposium*, pages 348–353, Washington, DC, USA, 2005. IEEE Computer Society.

- [264] Debdeep Mukhopadhyay. An Improved Fault Based Attack of the Advanced Encryption Standard. In *AFRICACRYPT*, pages 421–434, 2009.
- [265] Debdeep Mukhopadhyay. An Improved Fault Based Attack of the Advanced Encryption Standard. In Bart Preneel, editor, *AFRICACRYPT*, volume 5580 of *Lecture Notes in Computer Science*, pages 421–434. Springer, 2009.
- [266] Debdeep Mukhopadhyay and Dipanwita Roy Chowdhury. An efficient end to end design of rijndael cryptosystem in 0.18μ cmos. In *VLSI Design*, pages 405–410, 2005.
- [267] Julian Murphy. Clockless physical unclonable functions. In *Proc. of 5th international conference on Trust and Trustworthy Computing*, TRUST’12, pages 110–121, 2012.
- [268] F. N. Najm. Transition Density: a new measure of activity in digital circuits. *IEEE Transactions on CAD*, 14(2):310–323, February 1993.
- [269] Giogio Di Natale, Marie-Lisa Flottes, and Bruno Rouzeyre. A Novel Parity Bit Scheme for SBox in AES Circuits. In *DDECS*, pages 1–5, Apr 2007.
- [270] Giogio Di Natale, Marie-Lisa Flottes, and Bruno Rouzeyre. On-Line Self-Test of AES Hardware Implementation. *WDSN*, 2007.
- [271] Michael Neve, Jean pierre Seifert, and Zhenghong Wang. Cache Time-Behavior Analysis on AES, 2006.
- [272] Michael Neve and Jean-Pierre Seifert. Advances on Access-Driven Cache Attacks on AES. In Eli Biham and Amr M. Youssef, editors, *Selected Areas in Cryptography*, volume 4356 of *Lecture Notes in Computer Science*, pages 147–162. Springer, 2006.
- [273] Michael Neve, Jean-Pierre Seifert, and Zhenghong Wang. A Refined Look at Bernstein’s AES Side-Channel Analysis. In Ferng-Ching Lin, Der-Tsai Lee, Bao-Shuh Lin, Shiuhyung Shieh, and Sushil Jajodia, editors, *ASIACCS*, page 369. ACM, 2006.
- [274] J. Note and E. Rannaud. From the bitstream to the netlist. In *FPGA ’08: Proceedings of the International ACM/SIGDA Symposium on Field Programmable Gate Arrays*, pages 264–271, 2008.
- [275] Kaisa Nyberg. Differentially uniform mappings for cryptography. In *EUROCRYPT*, pages 55–64, 1993.
- [276] A. Oliveira. Robust techniques for watermarking sequential circuit designs. In *DAC’99: Proceedings of the ACM/IEEE Design Automation Conference*, pages 837–842, 1999.
- [277] A.L. Oliveira. Techniques for the creation of digital watermarks in sequential circuit designs. *IEEE Transactions on CAD*, 20(9):1101 –1117, September 2001.
- [278] OpenCores. <http://www.opencores.org>, 2011.
- [279] Gerardo Orlando and Christof Paar. A High Performance Reconfigurable Elliptic Curve Processor for $GF(2^m)$. In *CHES ’00: Proceedings of the Second International Workshop on Cryptographic Hardware and Embedded Systems*, pages 41–56, London, UK, 2000. Springer-Verlag.
- [280] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache attacks and Countermeasures: the Case of AES. *Cryptology ePrint Archive*, Report 2005/271, 2005.

- [281] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache Attacks and Countermeasures: The Case of AES. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.
- [282] Maria Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen. A Side-Channel Analysis Resistant Description of the AES S-box. In *Proceedings of Fast Software Encryption (FSE 2005)*, LNCS Volume 3557, pages 413–423. Springer-Verlag, 2005.
- [283] E. Ozturk, G. Hammouri, and B. Sunar. Physical unclonable function with tristate buffers. In *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 3194–3197, 2008.
- [284] P. C. Kocher. Timing Attacks on Implementation of Diffie-Hellman, RSA, DSS and Other Systems. In *Proceeding of Crypto*, LNCS 1109, pages 104–113, 1996.
- [285] G. Letourneau P. Dusart and O. Vivolo. Differential Fault Analysis on AES. In *Cryptology ePrint Archive*, pages 293–306, Oct 2003.
- [286] Christof Paar. *Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields*. PhD thesis, Institute for Experimental Mathematics, Universität Essen, Germany, June 1994.
- [287] Christof Paar. A New Architecture for a Parallel Finite Field Multiplier with Low Complexity Based on Composite Fields. *IEEE Transactions on Computers*, 45(7):856–861, 1996.
- [288] D. Page. Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel, 2002.
- [289] D Page. Defending Against Cache-Based Side-Channel Attacks. *Information Security Technical Report*, 8(1):30 – 44, 2003.
- [290] Dan Page. Partitioned Cache Architecture as a Side-Channel Defence Mechanism. *IACR Cryptology ePrint Archive*, 2005:280, 2005.
- [291] A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes* (4th ed.). McGraw–Hill, 2002.
- [292] A. Papoulis and S. U. Pillai. Predictive technology model, 2012. <http://www.eas.asu.edu/~ptm/>.
- [293] Ravikanth S. Pappu. *Physical one-way functions*. PhD thesis, Massachusetts Institute of Technology, March 2001.
- [294] Ravikanth S. Pappu, Ben Recht, Jason Taylor, and Niel Gershenfeld. Physical one-way functions. *Science*, 297:2026–2030, 2002.
- [295] S. Paul, H. Mahmoodi, and S. Bhunia. Low-overhead f_{max} calibration at multiple operating points using delay sensitivity based path selection. *ACM Transactions on Design Automation of Electronic Systems*, 15(2):19:1–19:34, February 2010.
- [296] Paulo S. L. M. Barreto. The AES Block Cipher in C++.
- [297] David Peacham and Byron Thomas. âIJA DFA attack against the AES key scheduleâ. SiVenture White Paper 001, 26 October, 2006.
- [298] Colin Percival. Cache Missing for Fun and Profit. In *Proc. of BSDCan 2005*, 2005.

- [299] Steffen Peter and Peter Langendörfer. An efficient polynomial multiplier in $GF(2^m)$ and its application to ECC designs. In *DATE '07: Proceedings of the conference on Design, automation and test in Europe*, pages 1253–1258, San Jose, CA, USA, 2007. EDA Consortium.
- [300] G. Piret and J.J. Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad. In *CHES*, pages 77–88, Sept 2003.
- [301] Rishabh Poddar, Amit Datta, and Chester Rebeiro. A Cache Trace Attack on CAMELLIA. In Marc Joye, Debdeep Mukhopadhyay, and Michael Tunstall, editors, *InfoSecHiComNet*, volume 7011 of *Lecture Notes in Computer Science*, pages 144–156. Springer, 2011.
- [302] I. Pomeranz and S. M. Reddy. A measure of quality for n-detection test sets. *IEEE Transactions on Computers*, 53(11):1497–1503, 2004.
- [303] Reinhard Posch. Protecting Devices by Active Coating. *Journal of Universal Computer Science*, 4(7):652–668, 1998.
- [304] Norbert Pramstaller, Stefan Mangard, Sandra Dominikus, and Johannes Wolkerstorfer. Efficient aes implementations on asics and fpgas. In Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors, *AES Conference*, volume 3373 of *Lecture Notes in Computer Science*, pages 98–112. Springer, 2004.
- [305] Predictive Technology Model (PTM). [http://www.eas.asu.edu/\\$\sim\\$ptm/](http://www.eas.asu.edu/\simptm/).
- [306] Qiong Pu and Jianhua Huang. A Microcoded Elliptic Curve Processor for $GF(2^m)$ Using FPGA Technology. In *Communications, Circuits and Systems Proceedings, 2006 International Conference on*, volume 4, pages 2771–2775, June 2006.
- [307] R. Rivest, A. Shamir and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM, Previously released as an MIT "Technical Memo" in April 1977*, 21(2):120–126, 1978.
- [308] R. Rad, J. Plusquellic, and M. Tehranipoor. A sensitivity analysis of power signal methods for detecting Hardware Trojans under real process and environmental conditions. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 18(12):1735 –1744, December 2010.
- [309] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic. Power supply signal calibration techniques for improving detection resolution to hardware Trojans. In *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD'08)*, pages 632–639, Piscataway, NJ, USA, 2008.
- [310] D. Rai and J. Lach. Performance of delay-based Trojan detection techniques under parameter variations. In *Proc. IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'09)*, pages 58–65, Washington, DC, USA, 2009.
- [311] J. Rajendran, H. Borad, S. Mantravadi, and R. Karri. SLICED: Slide-based Concurrent Error Detection Technique for Symmetric Block Cipher. In *HOST*, pages 70–75, Jul 2010.

- [312] C. Rebeiro and D. Mukhopadhyay. Boosting Profiled Cache Timing Attacks with Apriori Analysis. *Information Forensics and Security, IEEE Transactions on*, PP(99):1, 2012.
- [313] Chester Rebeiro, Mainack Mondal, and Debdeep Mukhopadhyay. Pinpointing Cache Timing Attacks on AES. In *VLSI Design*, pages 306–311. IEEE Computer Society, 2010.
- [314] Chester Rebeiro and Debdeep Mukhopadhyay. High speed compact elliptic curve cryptoprocessor for fpga platforms. In *INDOCRYPT*, pages 376–388, 2008.
- [315] Chester Rebeiro and Debdeep Mukhopadhyay. Power attack resistant efficient fpga architecture for karatsuba multiplier. In *VLSI Design*, pages 706–711, 2008.
- [316] Chester Rebeiro and Debdeep Mukhopadhyay. Power Attack Resistant Efficient FPGA Architecture for Karatsuba Multiplier. In *VLSID '08: Proceedings of the 21st International Conference on VLSI Design*, pages 706–711, Washington, DC, USA, 2008. IEEE Computer Society.
- [317] Chester Rebeiro and Debdeep Mukhopadhyay. Cryptanalysis of CLEFIA Using Differential Methods with Cache Trace Patterns. In Aggelos Kiayias, editor, *CT-RSA*, volume 6558 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 2011.
- [318] Chester Rebeiro, Debdeep Mukhopadhyay, Junko Takahashi, and Toshinori Fukunaga. Cache Timing Attacks on CLEFIA. In Bimal Roy and Nicolas Sendrier, editors, *INDOCRYPT*, volume 5922 of *Lecture Notes in Computer Science*, pages 104–118. Springer, 2009.
- [319] Chester Rebeiro, Rishabh Poddar, Amit Datta, and Debdeep Mukhopadhyay. An Enhanced Differential Cache Attack on CLEFIA for Large Cache Lines. In Daniel J. Bernstein and Sanjit Chatterjee, editors, *INDOCRYPT*, volume 7107 of *Lecture Notes in Computer Science*, pages 58–75. Springer, 2011.
- [320] Chester Rebeiro, Sujoy Sinha Roy, and Debdeep Mukhopadhyay. Pushing the limits of high-speed $gf(2^m)$ elliptic curve scalar multiplication on fpgas. In *CHES*, pages 494–511, 2012.
- [321] Chester Rebeiro, Sujoy Sinha Roy, Sankara Reddy, and Debdeep Mukhopadhyay. Revisiting the itoh-tsujii inversion algorithm for fpga platforms. *IEEE Trans. VLSI Syst.*, 19(8):1508–1512, 2011.
- [322] Chester Rebeiro, A. David Selvakumar, and A. S. L. Devi. Bitslice Implementation of AES. In David Pointcheval, Yi Mu, and Kefei Chen, editors, *CANS*, volume 4301 of *Lecture Notes in Computer Science*, pages 203–212. Springer, 2006.
- [323] Mathieu Renaud, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Algebraic side-channel attacks on the aes: Why time also matters in dpa. In *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '09, pages 97–111, Berlin, Heidelberg, 2009. Springer-Verlag.
- [324] H.G. Rice. Classes of recursively enumerable sets and their decision problems. *Trans. Am. Math. Soc.*, 74:358–366, 1953.
- [325] V. Rijmen. Efficient Implementation of the Rijndael Sbox. <http://www.esat.kuleuven.ac.be/~rijmen/rijndael>.

- [326] Vincent Rijmen. Efficient Implementation of Rijndael S-Box. <http://www.esat.kuleuven.ac.be/~rijmen/rijndael/>.
- [327] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: Exploring Information Leakage in Third-Party Compute Clouds. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 199–212. ACM, 2009.
- [328] Sabel Mercurio Henríquez Rodríguez and Francisco Rodríguez-Henríquez. An FPGA Arithmetic Logic Unit for Computing Scalar Multiplication using the Half-and-Add Method. In *ReConFig 2005: International Conference on Reconfigurable Computing and FPGAs*, Washington, DC, USA, 2005. IEEE Computer Society.
- [329] Francisco Rodríguez-Henríquez and Çetin Kaya Koç. On Fully Parallel Karatsuba Multipliers for $GF(2^m)$. In *Proc. of the International Conference on Computer Science and Technology (CST)*, pages 405–410.
- [330] Francisco Rodríguez-Henríquez, Guillermo Morales-Luna, Nazar A. Saqib, and Nareli Cruz-Cortés. Parallel Itoh-Tsujii Multiplicative Inversion Algorithm for a Special Class of Trinomials. *Des. Codes Cryptography*, 45(1):19–37, 2007.
- [331] Francisco Rodríguez-Henríquez, N. A. Saqib, A. Díaz-Pérez, and Çetin Kaya Koç. *Cryptographic Algorithms on Reconfigurable Hardware (Signals and Communication Technology)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [332] Francisco Rodríguez-Henríquez, N. A. Saqib, A. Díaz-Pérez, and Cetin Kaya Koç. *Cryptographic Algorithms on Reconfigurable Hardware (Signals and Communication Technology)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [333] Francisco Rodríguez-Henríquez, Nazar A. Saqib, and Nareli Cruz-Cortés. A Fast Implementation of Multiplicative Inversion Over $GF(2^m)$. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume I*, pages 574–579, Washington, DC, USA, 2005. IEEE Computer Society.
- [334] A. Roy, F. Koushanfar, and I.L. Markov. Extended abstract: Circuit CAD tools as a security threat. In *HOST'08: Proceedings of the IEEE International Workshop on Hardware Oriented Security and Trust*, pages 65–66, 2008.
- [335] J. A. Roy, F. Kaushanfar, and I. L. Markov. Extended abstract: circuit CAD tools as a security threat. In *HOST'08: Proceedings of the International Workshop on Hardware-oriented Security and Trust*, pages 61–62, 2008.
- [336] J. A. Roy, F. Koushanfar, and I. L. Markov. EPIC: ending piracy of integrated circuits. In *DATE'08: Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1069–1074, 2008.
- [337] Sujoy Sinha Roy, Chester Rebeiro, and Debdeep Mukhopadhyay. Theoretical modeling of the itoh-tsujii inversion algorithm for enhanced performance on k-lut based fpgas. In *DATE*, pages 1231–1236, 2011.
- [338] Sujoy Sinha Roy, Chester Rebeiro, and Debdeep Mukhopadhyay. Theoretical Modeling of the Itoh-Tsujii Inversion Algorithm for Enhanced Performance on k-LUT based FPGAs. In *Design, Automation, and Test in Europe DATE-2011*, 2011.

- [339] U. Rührmair, S. Devadas, and F. Koushanfar. Security based on Physical Unclonability and Disorder. chapter Book Chapter in Introduction to Hardware Security and Trust. Springer, 2011.
- [340] U. Rührmair, C. Jaeger, C. Hilgers, M. Algasinger, G. Csaba, and M. Stutzmann. Security Applications of Diodes with Unique Current-Voltage Characteristics. In *Financial Cryptography and Data Security*, volume 6052 of *Lecture Notes in Computer Science*, pages 328–335. 2010.
- [341] Dhiman Saha, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury. A Diagonal Fault Attack on the Advanced Encryption Standard. *IACR Cryptology ePrint Archive*, page 581, 2009.
- [342] Dhiman Saha, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury. Pkdpa: An enhanced probabilistic differential power attack methodology. In *INDOCRYPT*, pages 3–21, 2011.
- [343] T. Sakurai. Genetic Algorithms: principles of natural selection applied to computation. *Science (New Series)*, 261(5123):872–878, Aug 1993.
- [344] T. Sakurai and A. R. Newton. Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. *IEEE Journal of Solid-State Circuits*, 25(2):584–594, apr 1990.
- [345] N. A. Saqib, F. Rodríguez-Henríquez, and A. Diaz-Perez. A Parallel Architecture for Fast Computation of Elliptic Curve Scalar Multiplication Over $GF(2^m)$. In *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings*, April 2004.
- [346] sasebo. Side-channel Attack Standard Evaluation Board. <http://staff.aist.go.jp/akashi.satoh/SASEBO/en/index.html>.
- [347] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh. A Compact Rijndael Hardware Architecture with S-Box Optimization. In Colin Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 239–254. Springer, 2001.
- [348] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh. A compact rijndael hardware architecture with s-box optimization. pages 239–254. Springer-Verlag, 2001.
- [349] Akashi Satoh, Takeshi Sugawara, Naofumi Homma, and Takafumi Aoki. High-Performance Concurrent Error Detection Scheme for AES Hardware. In *CHES*, pages 100–112, Aug 2008.
- [350] Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In *CHES*, pages 30–46, 2005.
- [351] B. Schneier. *Applied Cryptography: Protocols, Algorithms and Source Code in C*. John Wiley & Sons, 2001.
- [352] A. Schulman. Examining the Windows AARD detection code. *Dr. Dobb's Journal*, 18, September 1993.
- [353] J. Seberry and J. Pieprzyk. *An Introduction to Computer Security*. Advances in Computer Science Series, 1988.

- [354] Frank Sehnke, Christian Osendorfer, Jan Sölter, Jürgen Schmidhuber, and Ulrich Rührmair. Policy Gradients for Cryptanalysis. In *Proc. of 20th International Conference on Artificial Neural Networks (ICANN)*, volume 6354, pages 168–177, 2010.
- [355] N. Selmane, S. Guilley, and J. L. Danger. Practical Setup Time Violation Attacks on AES. In *EDCC'08: Proceedings of the European Dependable Computing Conference*, pages 91–96, 2008.
- [356] Nidhal Selmane, Sylvain Guilley, and Jean-Luc Danger. Practical Setup Time Violation Attacks on AES. pages 91–96. European Dependable Computing Conference, 2008.
- [357] Gaurav Sengar, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury. Secured flipped scan-chain model for crypto-architecture. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 26(11):2080–2084, 2007.
- [358] R. Sever, A.N. Ismailoglu, Y.C. Tekmen, and M. Askar. A high speed asic implementation of the rijndael algorithm. In *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, volume 2, pages II–541–4 Vol.2, 2004.
- [359] C. E. Shannon. Communication theory of secrecy systems, vol 28, no 4. In *Bell System Technical Journal*, pages 656–715. Bell, 1949.
- [360] Shay Gueron. Intel’s Advanced Encryption Standard (AES) Instructions Set (Rev : 3.0), 2010.
- [361] Y. Shi, N. Togawa, M. Yanagisawa, and T. Ohtsuki. Design-for-secure-test for crypto cores. In *ITC09: Proceedings of International Test Conference.*, page 1, 2009.
- [362] Y. Shi, N. Togawa, M. Yanagisawa, and T. Ohtsuki. Robust secure scan design against scan-based differential cryptanalysis. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, PP(99):1–15, 2011.
- [363] Koichi Shimizu, Daisuke Suzuki, and Tomomi Kasuya. Glitch PUF: Extracting Information from Usually Unwanted Glitches. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E95.A(1):223–233, 2012.
- [364] Daniel P. Siewiorek and Robert S. Swartz. Reliable Computer Systems: Design and Evaluation. *A K Peters/CRC Press; 3 edition*, 1998.
- [365] P. Simons, E. van der Sluis, and V. van der Leest. Buskeeper PUFs, a promising alternative to D Flip-Flop PUFs. In *Proc. of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 7 –12, 2012.
- [366] Eric Simpson and Patrick Schaumont. Offline hardware/software authentication for reconfigurable platforms. In *Proc. of the 8th international conference on Cryptographic Hardware and Embedded Systems (CHES)*, pages 311–323, 2006.
- [367] Sergei P. Skorobogatov and Ross J. Anderson. Optical Fault Induction Attacks. In *proceedings of CHES*, pages 2–12, Aug 2002.
- [368] William Stallings. *Cryptography and Network Security: Principles and Practice*. Pearson Education, 2002.
- [369] William Stallings. *Cryptography and Network Security (4th Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2005.

- [370] Douglas Stinson. *Cryptography: Theory and Practice, Second Edition*, pages 117–154. Chapman & Hall, CRC, London, UK, 2002.
- [371] Y. Su, J. Holleman, and B. Otis. A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations. In *Proc. of IEEE International Solid-State Circuits Conference(ISSCC)*, pages 406–611, 2007.
- [372] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Design Automation Conference*, pages 9–14, 2007.
- [373] G.E. Suh and S. Devadas. Physical unclonable functions for device authentication and secret key generation. In *DAC'07: Proceedings of the ACM/IEEE Design Automation Conference*, pages 9–14, 2007.
- [374] Interra Systems. Concorde – fast synthesis. http://www.interrasystems.com/eda/eda_concorde.php, 2009.
- [375] W. Chelton T. Good and M. Benaissa. Review of stream cipher candidates from a low resource hardware perspective.
- [376] Junko Takahashi and Toshinori Fukunaga. Differential Fault Analysis on AES with 192 and 256-Bit Keys. *Cryptology ePrint Archive*, Report 2010/023, 2010. <http://eprint.iacr.org/>.
- [377] Junko Takahashi, Toshinori Fukunaga, and Kimihiro Yamakoshi. DFA Mechanism on the AES Key Schedule. In Luca Breveglieri, Shay Gueron, Israel Koren, David Naccache, and Jean-Pierre Seifert, editors, *FDTG*, pages 62–74. IEEE Computer Society, 2007.
- [378] M. Tehrani and M. Koushanfar. A survey of hardware trojan taxonomy and detection. *IEEE Design and Test of Computers*, 27(1):10–25, Jan. 2010.
- [379] ThicketTM family of source code obfuscators. <http://www.semdesigns.com>, 2011.
- [380] Kris Tiri, Onur Aciçmez, Michael Neve, and Flemming Andersen. An analytical model for time-driven cache attacks. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 399–413. Springer, 2007.
- [381] I. Torunoglu and E. Charbon. Watermarking-based copyright protection of sequential functions. *IEEE Journal of Solid-State Circuits*, 35(3):434–440, March 2000.
- [382] Elena Trichina. Combinational logic design for aes subbyte transformation on masked data. *IACR Cryptology ePrint Archive*, 2003:236, 2003.
- [383] Trivium. <http://www.ecrypt.eu.org/stream/triviump3.html>.
- [384] Eran Tromer, Dag Arne Osvik, and Adi Shamir. Efficient Cache Attacks on AES, and Countermeasures. *Journal of Cryptology*, 23(2):37–71, 2010.
- [385] Dan Tsafrir, Yoav Etsion, and Dror G. Feitelson. Secretly Monopolizing the CPU without Superuser Privileges. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, SS'07, pages 17:1–17:18, Berkeley, CA, USA, 2007. USENIX Association.

- [386] Yukiyasu Tsunoo, Teruo Saito, Tomoyasu Suzuki, Maki Shigeri, and Hiroshi Miyauchi. Cryptanalysis of DES Implemented on Computers with Cache. In Colin D. Walter, Cetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2779 of *Lecture Notes in Computer Science*, pages 62–76. Springer, 2003.
- [387] Yukiyasu Tsunoo, Etsuko Tsujihara, Kazuhiko Minematsu, and Hiroshi Miyauchi. Cryptanalysis of Block Ciphers Implemented on Computers with Cache. In *International Symposium on Information Theory and Its Applications*, pages 803–806, 2002.
- [388] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Hiroyasu Kubo, and Kazuhiko Minematsu. Improving Cache Attacks by Considering Cipher Structure. *Int. J. Inf. Sec.*, 5(3):166–176, 2006.
- [389] Michael Tunstall and Olivier Benoît. Efficient Use of Random Delays in Embedded Software. In Damien Sauveron, Constantinos Markantonakis, Angelos Bilas, and Jean-Jacques Quisquater, editors, *WISTP*, volume 4462 of *Lecture Notes in Computer Science*, pages 27–38. Springer, 2007.
- [390] Michael Tunstall, Debdeep Mukhopadhyay, and Subidh Ali. Differential Fault Analysis of the Advanced Encryption Standard Using a Single Fault. In *WISTP*, pages 224–233, 2011.
- [391] Pim Tuyls, Geert-Jan Schrijen, Boris Škorić, Jan van Geloven, Nynke Verhaegh, and Rob Wolters. Read-proof hardware from protective coatings. In *Proc. of Cryptographic Hardware and Embedded Systems Workshop*, volume 4249 of *LNCS*, pages 369–383, 2006.
- [392] Ulrich Rührmair, Frank Sehnke, Jan S "olter, Gideon Dror, Srinivas Devadas, and J "urgen Schmidhuber. Modeling attacks on physical unclonable functions. In *Proc. of 17th ACM conference on Computer and communications security(CCS)*, pages 237–249, 2010.
- [393] U.S. Department of Commerce, National Institute of Standards and Technology. Digital signature standard (DSS), 2000.
- [394] Bhanu C. Vattikonda, Sambit Das, and Hovav Shacham. Eliminating Fine Grained Timers in Xen. In Christian Cachin and Thomas Ristenpart, editors, *CCSW*, pages 41–46. ACM, 2011.
- [395] Tobias Vejda, Dan Page, and Johann Großschädl. Instruction Set Extensions for Pairing-Based Cryptography. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *Pairing*, volume 4575 of *Lecture Notes in Computer Science*, pages 208–224. Springer, 2007.
- [396] Ingrid Verbauwhede, Senior Member, Patrick Schaumont, Student Member, and Henry Kuo. Design and performance testing of a 2.29-gb/s rijndael processor. *IEEE Journal of Solid-State Circuits*, 38:569–572, 2003.
- [397] Joachim von zur Gathen and Jamshid Shokrollahi. Efficient FPGA-Based Karatsuba Multipliers for Polynomials over F_2 . In *Selected Areas in Cryptography*, pages 359–369, 2005.
- [398] Serge Vrijaldenhoven. Acoustical Physical Uncloneable Functions. Master's thesis, Technische Universiteit Eindhoven, 2005.

- [399] C. Wang, J. Hill, J. C. Knight, and J. W. Davidson. Protection of software-based survivability mechanisms. In *DSN'01: Proceedings of the International Conference on Dependable Systems and Networks*, pages 193–202, 2001.
- [400] F. Wang. Formal verification of timed systems. *Proceedings of the IEEE*, 92(8):1283–1305, August 2004.
- [401] X. Wang, M. Tehranipoor, and J. Plusquellic. Detecting malicious inclusions in secure hardware: Challenges and solutions. In *HOST'08: Proceedings of the IEEE International Workshop on Hardware-Oriented Security and Trust*, pages 15–19, 2008.
- [402] Zhenghong Wang and Ruby B. Lee. New cache designs for thwarting software cache-based side channel attacks. In Dean M. Tullsen and Brad Calder, editors, *ISCA*, pages 494–505. ACM, 2007.
- [403] Zhenghong Wang and Ruby B. Lee. A Novel Cache Architecture with Enhanced Performance and Security. In *MICRO*, pages 83–93. IEEE Computer Society, 2008.
- [404] André Weimerskirch and Christof Paar. Generalizations of the Karatsuba Algorithm for Efficient Implementations. *Cryptology ePrint Archive*, Report 2006/224, 2006.
- [405] M. A. Weiss. *Data Structures and Algorithm Analysis in C* (2nd ed.). Pearson Education, 1997.
- [406] Michael Weiβ, Benedikt Heinz, and Frederic Stumpf. A cache timing attack on aes in virtualization environments. In Angelos D. Keromytis, editor, *Financial Cryptography*, volume 7397 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 2012.
- [407] Michael J. Wirthlin and Brian McMurtrey. IP delivery for FPGAs using applets and JHDL. In *Proceedings of the 39th annual Design Automation Conference*, DAC '02, pages 2–7, New York, NY, 2002. ACM.
- [408] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty. Towards Trojan-free trusted ICs: problem analysis and detection scheme. In *Proc. Conference on Design, Automation and Test in Europe (DATE'08)*, pages 1362–1365, New York, NY, USA, 2008.
- [409] Johannes Wolkerstorfer. *Hardware Aspects of Elliptic Curve Cryptography*. PhD thesis, Institute for Applied Information Processing and Communications, Graz University of Technology, 2004.
- [410] Johannes Wolkerstorfer, Elisabeth Oswald, and Mario Lamberger. An asic implementation of the aes sboxes. In *Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology*, CT-RSA '02, pages 67–78, London, UK, UK, 2002. Springer-Verlag.
- [411] J. C. Wray. An Analysis of Covert Timing Channels. In *Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on*, pages 2 –7, may 1991.
- [412] Kaijie Wu, Ramesh Karri, G. Kuznetsov, and M. Goessel. Low Cost Concurrent Error Detection for the Advanced Encryption Standard. In *ITC*, pages 1242–1248, Oct 2004.
- [413] M. G. Xakellis and F. N. Najm. Statistical estimation of the switching activity in digital circuits. In *DAC '94: Proceedings of the Design Automation Conference*, pages 728–733, 1994.

- [414] Chenglu Jin Xiaofei Guo, Debdeep Mukhopadhyay and Ramesh Karri. Nrepo:normal basis recomputing with permuted operands. Cryptology ePrint Archive, Report 2014/497, 2014. <http://eprint.iacr.org/>.
- [415] Debdeep Mukhopadhyay Xiaofei Guo and Ramesh Karri. Provably secure concurrent error detection against differential fault analysis. Cryptology ePrint Archive, Report 2012/552, 2012. <http://eprint.iacr.org/>.
- [416] Debdeep Mukhopadhyay Xiaofei Guo and Ramesh Karri. Provably secure concurrent error detection against differential fault analysis. Cryptology ePrint Archive, Report 2012/552, 2012. <http://eprint.iacr.org/>.
- [417] Xilinx. Using Block RAM in Spartan-3 Generation FPGAs. Application Note, XAPP-463, 2005.
- [418] Xilinx. Using Look-Up Tables as Distributed RAM in Spartan-3 Generation FPGAs. Application Note, XAPP-464, 2005.
- [419] Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamoto, Kazuo Ohta, Takao Ochiai, Masahiko Takenaka, and Kouichi Itoh. Uniqueness Enhancement of PUF Responses Based on the Locations of Random Outputting RS Latches. In *Proc. of 13th International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 390–406, October 2011.
- [420] Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamoto, Kazuo Ohta, Masahiko Takenaka, and Kouichi Itoh. Variety enhancement of PUF responses using the locations of random outputting RS latches. In *Journal of Cryptographic Engineering*, pages 1–15, 2012.
- [421] B. Yang, K. Wu, and R. Karri. Secure scan: A design-for-test architecture for crypto chips. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 25(10):2287–2293, Oct. 2006.
- [422] Bo Yang, Kaijie Wu, and Ramesh Karri. Scan based side channel attack on dedicated hardware implementations of data encryption standard. In *ITC '04: Proceedings of the International Test Conference*, pages 339–344, Washington, DC, USA, 2004. IEEE Computer Society.
- [423] Y. Yao, M. Kim, J. Li, I. L. Markov, and F. Koushanfar. ClockPUF: Physical Unclonable Functions based on Clock Networks. In *Design, Automation & Test in Europe (DATE)*, 2013.
- [424] H. Yotsuyanagi and K. Kinoshita. Undetectable fault removal of sequential circuits based on unreachable states. In *VTS'98: Proceedings of the IEEE VLSI Test Symposium*, pages 176–181, 1998.
- [425] Pengyuan Yu and Patrick Schaumont. Secure FPGA circuits using controlled placement and routing. In *Proceedings of International Conference on Hardware Software Codesign (CODES+ISSS)*, pages 45–50. ACM, 2007.
- [426] L. Yuan and G. Qu. Information hiding in finite state machine. In *IH'04: Proceedings of the International Conference on Information Hiding*, IH'04, pages 340–354, 2004.
- [427] Erik Zenner. Cache Timing Analysis of HC-256. In *15th Annual International Workshop, SAC 2008*, 2008.

- [428] XinJie Zhao and Tao Wang. Improved Cache Trace Attack on AES and CLEFIA by Considering Cache Miss and S-box Misalignment. *Cryptology ePrint Archive, Report 2010/056*, 2010.
- [429] Zhijie Jerry Shi and Xiao Yang and Ruby B. Lee. Alternative Application-Specific Processor Architectures for Fast Arbitrary Bit Permutations. *IJES*, 3(4):219–228, 2008.
- [430] X. Zhuang, T. Zhang, H.S. Lee, and S. Pande. Hardware assisted control flow obfuscation for embedded processors. In *CASES '04: Proceedings of the 2004 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems*, pages 292–302, 2004.
- [431] Xiaotong Zhuang, Tao Zhang, Hsien-Hsin S. Lee, and Santosh Pande. Hardware Assisted Control Flow Obfuscation for Embedded Processors. In Mary Jane Irwin, Wei Zhao, Luciano Lavagno, and Scott A. Mahlke, editors, *CASES*, pages 292–302. ACM, 2004.
- [432] Xiaotong Zhuang, Tao Zhang, and Santosh Pande. HIDE: an Infrastructure for Efficiently Protecting Information Leakage on the Address Bus. In Shubu Mukherjee and Kathryn S. McKinley, editors, *ASPLOS*, pages 72–84. ACM, 2004.