Design of Low-Power Coarse-Grained Reconfigurable Architectures

Yoonjin Kim Rabi N. Mahapatra



A CHAPMAN & HALL BOOK

Design of Low-Power Coarse-Grained Reconfigurable Architectures

Design of Low-Power Coarse-Grained Reconfigurable Architectures

Yoonjin Kim Rabi N. Mahapatra



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

© 2011 by Taylor and Francis Group, LLC CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works

Printed in the United States of America on acid-free paper 10 9 8 7 6 5 4 3 2 1

International Standard Book Number-13: 978-1-4398-2511-2 (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

Li	st of	Figure	es	xi
Li	st of	Tables	5	xv
Pı	refac	e		xvii
1	Intr	oducti	ion	1
	1.1	Coarse	e-Grained Reconfigurable Architecture	1
	1.2	Object	tive and Approach	3
	1.3	Overv	iew of the Book's Contents	3
2	Tre	nds in	CGRA	7
	2.1	Introd	uction	7
	2.2	Archit	ecture	7
		2.2.1	MorphoSys	8
			2.2.1.1 RC Array	8
			2.2.1.2 TinyRISC Processor	12
			2.2.1.3 Frame Buffer and DMA	12
			2.2.1.4 Context Memory	13
		2.2.2	REMARC	14
			2.2.2.1 Nano Processor Array Structure	15
		2.2.3	PACT-XPP	17
			2.2.3.1 Configuration Manager	18
			2.2.3.2 Micro-Controller	18
			2.2.3.3 Processing Array	19
		2.2.4	ADRES	20
		2.2.5	RaPiD	23
			2.2.5.1 Functional Units	23
			2.2.5.2 Configurable Interconnect	24
		2.2.6	PipeRench	25
	2.3	Design	Space Exploration	27
		2.3.1	PACT-XPP - PARO	27
		2.3.2	KressArray Xplorer	29
		2.3.3	ADRES - DRESC	29
	2.4	Code	Compilation and Mapping	31
		2.4.1	MorphoSys - Hierarchical Loop Synthesis	31

		2.4	.1.1 Function Inlining	33
		2.4	.1.2 Transformation to Context Codes	33
		2.4	.1.3 Hierarchical Loop Synthesis	33
		2.4.2 AI	ORES - Modulo Scheduling	35
	2.5	Physical I	mplementation	37
		2.5.1 Mo	orphoSys	37
		2.5.2 PA	CT-XPP	37
		2.5.3 AI	DRES	39
		2.5.4 Pir	peRench	11
	2.6	Summary		11
3	\mathbf{CG}	RA for Hi	gh Performance and Flexibility 4	15
	3.1	Performan	ce versus Flexibility for Embedded Systems 4	15
	3.2	Performan	ce Evaluation Examples	16
		3.2.1 Mo	orphoSys	17
		3.2	.1.1 Motion Estimation for MPEG	17
		3.2	.1.2 Discrete Cosine Transform (DCT) for MPEG	18
		3.2	.1.3 Automatic Target Recognition 4	18
		3.2	.1.4 International Data Encryption Algorithm 4	19
		3.2.2 AI	DRES	51
		3.2	.2.1 H.264/AVC Decoder	51
		3.2	.2.2 Image Processing Algorithm	51
		3.2.3 PA	CT-XPP	53
		3.2.4 Pip	peRench	56
		3.2.5 RI)P	57
	3.3	Summary	····· ٤	59
4	Bas	e CGRA	Implementation 6	61
	4.1	Introducti	on	31
	4.2	Reconfigu	rable Array Architecture Coupling with Processor . 6	31
	4.3	Base Reco	nfigurable Array Architecture	33
		4.3.1 Pro	cocessing Element	34
		4.3.2 PE	Array	34
		4.3.3 Fra	ame Buffer	35
		4.3.4 Co	nfiguration Cache 6	36
		4.3.5 Ex	ecution Controller	36
	4.4	Breakdow	n of Area and Delay Cost	37
		4.4.1 A	rea and Delay	37
	4.5	Summary	· · · · · · · · · · · · · · · · · · ·	37
5	Pov	ver Consu	mption in CGRA 7	'1
	5.1	Introducti	on	71
	5.2	Breakdow	n of Power Consumption in CGRA	71
		5.2.1 Ba	se CGRA	71
		5.2.2 AI	DRES	72

	5.3	Necessity of Power-Conscious CGRA Design	74
	5.4	Summary	76
6	Low	-Power Reconfiguration Technique	77
U	6.1	Introduction	77
	6.2	Motivation	77
	0	6.2.1 Loop Pipelining	77
		6.2.2 Spatial Mapping and Temporal Mapping	82
	6.3	Individual Approaches to Reduce Power in Configuration Cache	83
	0.0	6.3.1 Spatial Mapping with Context Reuse	83
		6.3.2 Temporal Mapping with Context Pipelining	84
		6.3.3 Limitation of Individual Approaches	85
	64	Integrated Approach to Reduce Power in Configuration Cache	86
	0.1	6.4.1 Reusable Context Pipelining	86
		6.4.2 Limitation of Reusable Context Pipelining	88
		6.4.3 Hybrid Configuration Cache Structure	91
	65	Application Mapping Flow	91
	0.0	6.5.1 Temporal Mapping Algorithm	92
		6.5.1.1 Covering	92
		6.5.1.2 Time Assignment	93
		6.5.1.3 Place Assignment	93
		6.5.2 Context Rearrangement	94
	66	Experiments	96
	0.0	6.6.1 Experimental Setup	96
		662 Results	96
		6.6.2.1 Necessary Context Registers for Evaluated	50
		Kernels	96
		6.6.2.2 Configuration Cache Size	97
		6 6 2 3 Performance Evaluation	98
		6 6 2 4 Power Evaluation	98
	67	Summary	99
	0		00
7	Dyn	namic Context Compression for Low-Power CGRA	101
	7.1	Introduction	101
	7.2	Preliminary	101
		7.2.1 Context Architecture	101
	7.3	Motivation	102
		7.3.1 Power Consumption by Configuration Cache	102
		7.3.2 Valid Bit-Width of Context Words	102
		7.3.3 Dynamic Context Compression for Low-Power CGRA	103
	7.4	Design Flow of Dynamically Compressible Context	
		Architecture	104
		7.4.1 Context Architecture Initialization	106
		7.4.2 Field Grouping	106
		7.4.3 Field Sequence Graph Generation	106

vii

		7.4.4	Generation of Field Control Signal
			7.4.4.1 Control Signals for ALU-Dependent Fields . 107
			7.4.4.2 Control Signals for ALU-Independent Fields 108
		7.4.5	Field Positioning
			7.4.5.1 Field Positioning on Uncompressed Context
			Word \ldots 110
			7.4.5.2 Field Positioning on Compressed Context
			Word
		7.4.6	Compressible Context Architecture
		7.4.7	Context Evaluation
	7.5	Exper	iments
		7.5.1	Experimental Setup
		7.5.2	Results
			7.5.2.1 Area Cost Evaluation
			7.5.2.2 Performance Evaluation 120
			7.5.2.3 Context Compression Batio and Power
			Evaluation 121
	7.6	Summ	arv 121
		Samm	ary
8	Dyr	namic	Context Management for Low-Power CGRA 123
	8.1	Introd	uction
	8.2	Motiv	ation
		8.2.1	Power Consumption by Configuration Cache 123
		8.2.2	Redundancy of Context Words
			8.2.2.1 NOP Context Words
			8.2.2.2 Consecutively Same Part in Context Words . 124
			8.2.2.3 Redundancy Ratio
	8.3	Dynar	nic Context Management
		8.3.1	Context Partitioning
		8.3.2	Context Management at Transfer Time
		8.3.3	Context Management at Run Time
	8.4	Exper	iments \ldots 132
		8.4.1	Experimental Setup 132
		8.4.2	Results
			8.4.2.1 Area Cost Evaluation
			8.4.2.2 Power Evaluation
			8.4.2.3 Performance Evaluation
	8.5	Summ	ary
a	Cos	t-Effor	tive Array Fabric 137
0	0 1	Introd	uction 127
	9.1	Prelim	inary 127
	9.4	0.9.1	Resource Sharing 120
		9.2.1	Resource Dipolining 129
	0.2	9.2.2 Cost I	Itesource I ipenning 130
	9.0	COSt-1	meetive neconingurable Array rabric

		9.3.1	Motivati	on	143
			9.3.1.1	Characteristics of Computation-Intensive and	
				Data-Parallel Applications	143
			9.3.1.2	Redundancy in Conventional Array Fabric .	143
		9.3.2	New Cos	st-Effective Data Flow-Oriented Array	
			Structur	e	145
			9.3.2.1	Derivation of Data Flow-Oriented	
				Array Structure	145
			9.3.2.2	Mitigation of Spatial Limitation in the	
				Proposed Array Structure	148
		9.3.3	Data Flo	ow-Oriented Array Design Flow	148
			9.3.3.1	Input Reconfigurable Array Fabric	149
			9.3.3.2	New Array Fabric Specification - Phase I	149
			9.3.3.3	New Array Fabric Specification - Phase II	154
			9.3.3.4	Connectivity Enhancement	156
		9.3.4	Cost-Eff	ective Array Fabric with Resource Sharing and	
			Pipelinir	1º	156
	9.4	Experi	iments .		161
		9.4.1	Experim	ental Setup	161
			9.4.1.1	Evaluated Applications	161
			9.4.1.2	Hardware Design and Power Estimation	161
		9.4.2	Results	· · · · · · · · · · · · · · · · · · ·	162
			9.4.2.1	Area Evaluation	162
			9.4.2.2	Performance Evaluation	162
			9.4.2.3	Power Evaluation	165
	9.5	Summ	ary		165
10	TT •		1.D		105
10	Hiei	rarchic	al Recoi	infigurable Computing Arrays	167
	10.1	Introd	uction .		167
	10.2	Motiva	ation \ldots		167
		10.2.1	Limitatio	on of Existing Processor-RAA Communication	100
		10.0.0	Structur	es	107
	10.9	10.2.2 C	RAA-ba	sed Computing Hierarchy	169
	10.3	Compu	Comment	rarchy in UGRA	109
		10.3.1	Comput	Ing Hierarchy—Size and Speed	170
		10.3.2	Resource	$\begin{array}{c} \text{Biggs} \text{ in ROU and RAA} \\ \text{Solution} \\ \text{Figure 1} \\ \text{Colored RAA} \\ \text{Solution} \\ \text{Colored RAA} \\ $	171
	10.4	10.3.3	Computi	ing Flow Optimization	174
	10.4	Experi	ments .		170
		10.4.1	Experim	ental Setup	170
			10.4.1.1	Architecture Implementation	176
		10.4.0	10.4.1.2	Evaluated Applications	176
		10.4.2	Kesults	Arres Clast Eventson	176
			10.4.2.1	Area Cost Evaluation	176
			10.4.2.2	Performance Evaluation	178
			10.4.2.3	Power Evaluation	178

10.5 Summary \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots	181
11 Integrated Approach to Optimize CGRA	183
11.1 Combination among the Cost-Effective CGRA Design Schemes	183
11.2 Case Study for Integrated Approach	183
11.2.1 An CGRA Design Example Merging Three Design	
Schemes	183
11.2.2 Results	184
11.2.2.1 Area and Performance Evaluation	184
11.2.2.2 Power Evaluation	185
11.3 Potential Combinations and Expected Outcomes	185
11.4 Summary	188
Bibliography	189
Index	197

List of Figures

1.1	Block diagram of general CGRA	2
2.1	Components of MorphoSys implementation (M1 chip)	8
2.2	MorphoSys 8x8 RC array interconnection structure	9
2.3	Express lane connectivity	0
2.4	Reconfigurable cell (RC) structure and context architecture	
	of MorphoSys	1
2.5	TinyRISC pipeline stage	2
2.6	Organization of context memory 1	3
2.7	Block diagram of a microprocessor with REMARC 1	4
2.8	Block diagram of REMARC	5
2.9	Block diagram of REMARC	6
2.10	XPP-based CSoC architecture	7
2.11	CSoC RAM topology	8
2.12	XPP64 architecture overview and structure of one ALU PAE	
	module	9
2.13	ADRES core	1
2.14	FU structure	2
2.15	RaPiD architecture block diagram	4
2.16	PipeRench architecture	6
2.17	PARO design flow	8
2.18	KressArray Xplorer overview	0
2.19	Dynamically reconfigurable embedded system compiler	
	(DRESC) framework	2
2.20	Flow of compilation	4
2.21	Mapping a kernel onto a 2x2 reconfigurable array	6
2.22	Layout of MorphoSys M1 chip	8
2.23	XPP/LEON-based CSoC architecture with multi-layer AMBA	
	interface	9
2.24	Floorplan of the reconfigurable cell in ADRES 4	0
2.25	PipeRench PE floorplan 4	2
2.26	Chip micrograph of PipeRench	3
3.1	Performance and flexibility tradeoffs for different kinds of IP- types A	6
3.2	Performance comparison (in cycles) for motion estimation 4	.8

3	3.3	DCT/IDCT performance comparison (cycles)
3	3.4	Performance comparison of MorphoSys for SLD (ATR)
3	3.5	Performance comparison for IDEA mapping on MorphoSys .
3	8.6	Scheduling results of kernels
3	3.7	Simulation results of the mapped $H.264/AVC$ decoder (at
		100MHz)
3	8.8	Different ADRES architectures and their characteristics
3	3.9	Comparison of the image processing benchmarks for different
		ADRES architectures and the TI c64x
3	8.10	Speedup of H.264
3	3.11	Speedup for nine kernels: raw speedup of a 128-bit fabric with
		8-bit PEs and eight registers per PE
3	3.12	Outline of a CGC-based data-path
3	3.13	Applications' characteristics
3	8.14	Execution times and application speedups
Δ	11	Basic types of reconfigurable array coupling
4	12	Block diagram of base CGRA
4	13	Processing element structure of base BAA
4	1.4	Interconnection structure of PE array
4	1.5	Distributed configuration cache structure
4	1.6	Area cost breakdown for CGRA
4	1.7	Cost analysis for a PE
5).l	Power cost breakdown for CGRA running 2D-FDCT
5).2 ()	Power cost breakdown for only ADRES core with IDC1
5	0.3	Overview of entire ADRES processor power consumption with
		IDCT
6	6.1	A 4x4 reconfigurable array
6	5.2	C-code of Eq. 6.1
6	5.3	Execution model for CGRA
6	6.4	Comparison between temporal mapping and spatial mapping
6	5.5	Configuration cache structure for context reuse
6	6.6	Cache structure for context pipelining
6	3.7	Proposed configuration cache structure
6	6.8	Reusable context pipelining for Eq. 6.1
6	5.9	Reusable context pipelining with temporal cache
6	3 10	Reusable context pipelining according to the execution time for

	1 0	
6.8	Reusable context pipelining for Eq. 6.1	89
6.9	Reusable context pipelining with temporal cache	90
6.10	Reusable context pipelining according to the execution time for	
	one iteration $(i > 1)$	90
6.11	Hybrid configuration cache structure	92
6.12	Application mapping flow for base architecture and proposed	
	architecture	93
6.13	Temporal mapping steps	94
6.14	Context rearrangement	95
	-	

7.1	Valid bit-width of context words	103
7.2	Entire design flow	104
7.3	Context architecture initialization	105
7.4	Field grouping	107
7.5	Field sequence graph	108
7.6	Control signals for 'MUX_B' and 'PRED'	109
7.7	Updated FSG from flag merging	110
7.8	Default field positioning	111
7.9	Field concurrency graph	112
7.10	Examples of 'Find_Interval'	115
7.11	Multiplexer port-mapping graph	117
7.12	Compressible context architecture	118
8.1	Consecutively same part in context words	125
8.2	Redundancy ratio of context words	126
8.3	An example of PE and context architecture	127
8.4	Context partitioning	128
8.5	Comparison between general CE and proposed CE	128
8.6	Context management when context words are transferred	129
8.7	Context management at run time	131
9.1	Snapshots of three mappings	139
9.2	Eight multipliers shared by sixteen PEs	140
9.3	The connection between a PE and shared multipliers	141
9.4	Critical paths	141
9.5	Loop pipelining with pipelined multipliers	142
9.6	Subtask classification	144
9.7	Data flow on square reconfigurable array	144
9.8	Data flow-oriented array structure derived from three types of	
	data flow	145
9.9	An example of data flow-oriented array	146
9.10	Snapshots showing the maximum utilization of PEs	147
9.11	Overall design flow	149
9.12	Basic concept of local triangulation method	151
9.13	Local triangulation method	152
9.14	Interconnection derivation in Phase I	153
9.15	New array fabric example by Phase I	155
9.16	Global triangulation method when $n = 2$ (L2)	157
9.17	Global triangulation method when $n = 2$ (L2)	158
9.18	New array fabric example by Phase II	158
9.19	New array fabric example by connectivity enhancement	159
9.20	New array fabric with resource sharing and pipelining	160
9.21	Mapping example on new array fabric	161
10.1	Analogy between memory and RAA-computing hierarchy	169

10.2 Computing hierarchy of CGRA	170
10.3 CGRA configuration with RCC and RAA	171
10.4 Two cases of functional resource assignment	172
10.5 Critical resource sharing and pipelining in L1 and L2 PE array	173
10.6 Interconnection structure among RCC, shared critical resources	
and L2 PE array \ldots	173
10.7 Four cases of computing flow according to the input/output	
size of application	175
10.8 Performance comparison	179
10.9 Power comparison	180
11.1 Combination flow of the proposed design schemes	184
11.2 A combination example combining three design schemes	184
11.3 Potential combination of multiple design schemes	187

List of Tables

6.1	Architecture specification of base and proposed architecture .	97
6.2	Necessary context registers for evaluated kernels	97
6.3	Size of configuration cache and context registers	98
6.4	Power reduction ratio by reusable context pipelining \ldots .	99
7.1	Notations for port-mapping algorithm	113
7.2	Area overhead by dynamic context compression	120
7.3	Power reduction ratio by dynamic context compression	120
8.1	Area overhead by dynamic context management	133
8.2	Power reduction ratio by dynamic context management	134
9.1	Area reduction ratio by RSPA and NAF	163
9.2	Applications characteristics and performance evaluation	164
9.3	Power reduction ratio by RSP+NAF	165
10.1	Comparison of the basic coupling types	168
10.2	Comparison of the architecture implementations	176
10.3	Applications characteristics	177
10.4	Area cost comparison	177
11.1	Area reduction ratio by integrated RAA	185
11.2	Entire power comparison	186

Preface

Application-specific optimization of embedded systems becomes inevitable to satisfy the market demand for designers to meet tighter constraints on cost, performance and power. On the other hand, the flexibility of a system is also important to accommodate the short time-to-market requirements for embedded systems. To compromise these incompatible demands, coarse-grained reconfigurable architecture (CGRA) has emerged as a suitable solution. A typical CGRA requires many processing elements (PEs) and a configuration cache for reconfiguration of its PE array. However, such a structure consumes significant area and power. Therefore, designing cost-effective CGRA has been a serious concern for reliability of CGRA-based embedded systems.

As an effort to provide such cost-effective design, the first half of this book focuses on reducing power in the configuration cache. For power saving in the configuration cache, a *low-power reconfiguration technique* is presented based on reusable context pipelining achieved by merging the concept of context reuse into context pipelining. In addition, we propose *dynamic context compression* capable of supporting only required bits of the context words set to enable and the redundant bits set to disable. Finally, we provide *dynamic context management* capable of reducing power consumption in configuration cache by controlling a read/write operation of the redundant context words.

In the second part of this book, we focus on designing a cost-effective PE array to reduce area and power. For area and power saving in a PE array, we devise a *cost-effective array fabric* that addresses novel rearrangement of processing elements and their interconnection designs to reduce area and power consumption. In addition, *hierarchical reconfigurable computing arrays* are proposed consisting of two reconfigurable computing blocks with two types of communication structure together. The two computing blocks have shared critical resources and such a sharing structure provides efficient communication interface between them with reducing overall area. Based on the proposed design approaches, a CGRA combining the multiple design schemes is shown to verify the synergy effect of the integrated approach.

Audience for This Book

This book is intended for computer professionals, graduate students, and advanced undergraduates who need to understand issues involved in designing and constructing embedded systems. The reader is assumed to have had introductory courses in digital system, VLSI design, computer architecture, or equivalent work experience.

Chapter 1

Introduction

1.1 Coarse-Grained Reconfigurable Architecture

With the growing demand for high quality multimedia, especially over portable media, there has been continuous development on more sophisticated algorithms for audio, video, and graphics processing. These algorithms have the characteristics of data-intensive computation of high complexity. For such applications, we can consider two extreme approaches to implementation: software running on a general purpose processor and hardware in the form of Application-Specific Integrated Circuit (ASIC). In the case of general purpose processor, it is flexible enough to support various applications but may not provide sufficient performance to cope with the complexity of the applications. In the case of ASIC, we can optimize best in terms of power and performance but only for a specific application. With a coarse-grained reconfigurable architecture (CGRA), we can take advantage of the above two approaches. This architecture has higher performance level than general purpose processor and wider applicability than ASIC.

As the market pressure of embedded systems compels the designer to meet tighter constraints on cost, performance, and power, the application specific optimization of a system becomes inevitable. On the other hand, the flexibility of a system is also important to accommodate rapidly changing consumer needs. To compromise these incompatible demands, domain-specific design is focused on as a suitable solution for recent embedded systems. Coarse-grained reconfigurable architecture is the very domain-specific design in that it can boost the performance by adopting specific hardware engines while it can be reconfigured to adapt to ever-changing characteristics of the applications.

Typically, a CGRA consists of a main processor, a Reconfigurable Array Architecture (RAA), and their interface as Figure 1.1. The RAA has identical processing elements (PEs) containing functional units and a few storage units such as ALU, multiplier, shifter and register file. The data buffer provides operand data to PE array through a high-bandwidth data bus. The configuration cache (or context memory) stores the context words used for configuring the PE array elements. The context register between a PE and a cache element (CE) in configuration cache is used to keep the cache access path from being the critical path of the CGRA.



FIGURE 1.1: Block diagram of general CGRA.

Unlike FPGA (most typical of a fine-grained reconfigurable architecture), which are built with bit-level configurable logic blocks (CLBs), CGRA is built with PEs, which are word-level configurable functional blocks. By raising the granularity of operations from a bit to a word, CGRA can improve on the speed and the performance as well as the resource utilization for computeintensive applications. Another consequence of this raised granularity is that whereas FPGA can be used for implementing any digital circuits, CGRA is targeted only for a limited set of applications, although different CGRAs may target different application domains. Still, CGRA retains the idea of "reprogrammable hardware" in the reprogrammable interconnects as well as in the configurable functional blocks (i.e., PEs). Moreover, since the amount of the configuration bit-stream is greatly reduced through the raised granularity, the configuration can be actually changed even at the runtime very fast. Most of the CGRAs feature single-cycle configuration change, fetching the configuration data from a distributed local cache. This unique combination of efficiency and flexibility, which is the main advantage of CGRA, explains an evaluation result [9] that under certain conditions CGRAs are actually more cost-effective for wireless communication applications than alternatives such as FPGA implementations as well as DSP architectures. It is worth mentioning that the improved efficiency of CGRAs in terms of the performance and flexibility is a result of the architecture specialization for compute-intensive applications.

In spite of the above advantages, the deployment of CGRA is prohibitive due to its significant area and power consumption. This is due to the fact that CGRA is composed of several memory components and the array of many processing elements including ALU, multiplier and divider, etc. Especially, processing element (PE) array occupies most of the area and consumes most of the power in the system to support flexibility and high performance. Therefore, reducing area and power consumption in the PE array has been a serious concern for the adoption of CGRA.

1.2 Objective and Approach

This book explores the problem of reducing area and power in CGRA based on architecture optimization. To provide cost-effective CGRA design, the following questions are considered.

- How to reduce area and power consumption in CGRA? For power saving in CGRA, we should obtain area and power breakdown data of CGRA to identify area and power-dominant components. Then the components may be optimized for area and power by removing redundancies of CGRA wasting area and power. Such redundancies may depend on the characteristics of computation model or applications.
- How to design cost-effective CGRA with non-sacrificing or enhancing performance? Ultimately, the goals of designing cost-effective CGRA is that proposed approaches do not cause performance degradation with saving area and power. It means that the proposed cost-effective CGRA keeps original functionality of CGRA intact and does not increase critical path delay. In addition, the performance may be enhanced by optimizing the performance bottleneck with keeping the area and power-efficient approaches.

In this book, these central questions are addressed for area/power-critical components of CGRA and we suggest new frameworks to achieve these goals. The validation of the proposed approaches is demonstrated through the use of real application benchmarks and gate level simulations.

1.3 Overview of the Book's Contents

This volume is divided into 11 chapters as follows:

• Chapter 1. Introduction

This chapter introduces general characteristics of Coarse-Grained Reconfigurable Architecture (CGRA). In addition we present the contribution and the organization of this book.