New design methodologies for low complexity fir filters and reconfigurable constant multipliers

Chen, Jiajia

2010

Chen, J. (2010). New design methodologies for low complexity fir filters and reconfigurable constant multipliers. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46976>

<https://doi.org/10.32657/10356/46976>

Nanyang Technological University

Downloaded on 26 Apr 2023 17:26:29 SGT

48
2x
p/Blue

NEW DESIGN METHODOLOGIES FOR LOW COMPLEXITY FIR FILTERS AND RECONFIGURABLE CONSTANT MULTIPLIERS

CHEN JIAJIA

School of Electrical & Electronic Engineering

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

2010

Abstract

Multipliers, being the area and power hungry units, are deciding factors to the overall area, speed, and power consumption of digital circuits. To meet the stringent design requirements, multiplications have been avoided, reduced or replaced if possible by powerful design methodologies in high-level synthesis for many compute intensive digital signal processing problems. Multiple Constant Multiplication (MCM) and Reconfigurable Constant Multiplier (RCM) problems are two typical problems that have attracted a lot of attention in recent years. In this thesis, the first part presents a new paradigm of design methodology to reduce the complexity of MCM block of finite impulse response (FIR) digital filters. Motivated by the advantages of reconfigurable architectures, this thesis also presents a new high level design methodology for RCM block. This new technique leads to two new architectures for the implementation of circular convolution. Last but not least, an efficient switching power estimation method for the MCM block of FIR filter is proposed in view of the inefficiency of using gate-level and circuit-level techniques for its power estimation.

Acknowledgements

I wish to thank my supervisor, Associate Professor Chang Chip Hong, for his intellectual guidance, constant attention, unceasing encouragement and strict demands through this research program. In particular, I wish to thank him for teaching me some of the philosophical aspects of research and some intangible skills which I consider are the most invaluable knowledge I have acquired in this research program.

I would like to thank Dr Xu Fei, Mr Mathias Faust and Mr Qian Hanhua for their friendship, help, discussions and suggestions pertaining to this research work. I would like to thank Mr Zhai Yao, Mr Bang Viet and Mr Pang Yuanchen for their tedious but indispensable work to speed up the research schedule.

To the staff and other students in Integrated Systems Research Laboratory and Center for High Performance Embedded Systems, I wish to convey my appreciation to all of them for their kind and friendly assistance.

I would like to express my gratitude to my parents and my wife for their consistent support and encouragement throughout the period of my study.

Finally, I would like to acknowledge the School of Electrical & Electronic Engineering, Nanyang Technological University for providing the research facility and the research scholarship.

Summary

Multipliers, being the area and power hungry units, are deciding factors to the overall area, speed, and power consumption of digital circuits. To meet the stringent design requirements, multiplications have been avoided, reduced or replaced if possible by powerful design methodologies in high-level synthesis for many compute intensive digital signal processing problems. Multiple Constant Multiplication (MCM) and Reconfigurable Constant Multiplier (RCM) problems are two typical problems that have attracted a lot of attention in recent years. In this research, the characteristics and limitations of existing design methodologies are studied in order to develop new optimization algorithms to automate the generation of area-time efficient solutions in an application specific integrated circuit (ASIC) design flow.

The first part of the thesis presents a new paradigm of design methodology to reduce the complexity of MCM block of finite impulse response (FIR) digital filters. A new Multi-root Binary Partition Graph (MBPG) is proposed to partition a set of coefficients into symbols so that common subexpression identification and elimination become congruent to parsing of information in data compression. A minimum number of different pairs or groups of symbols can then be used to code a set of coefficients based on their probability and conditional probability of occurrence. This information concept enables the notion of entropy to be applied as a quantitative measure to evaluate the coding density of different compositions of symbols towards a set of coefficients. The minimal MBPG synthesized by this information theoretic approach results in direct correspondences between the vertices and adders, and edges and physical interconnections. Unlike Common Subexpression Elimination (CSE) algorithms based on other data structures, the symbol-level information carried in each vertex and the graph isomorphism in MBPG also promise further fine-grain optimization in a reduced search space. Simulation results on benchmark FIR filters and randomly generated coefficients indicates that the proposed MBPG algorithm is capable of producing solutions with 40.38% area reduction and 25.77% throughput rate improvement over the baseline implementation with all the

coefficients recorded in Canonical Signed Digit (CSD) format with no attempt to eliminate any common subexpressions.

Motivated by the advantages of reconfigurable architectures, this thesis also presents a new high level design methodology for RCM block. CSE algorithm is first applied to the preset constants represented in minimum signed digit number system to obtain a minimum depth multi-root Directed Acyclic Graph (DAG). This DAG is converted into a primitive Data Flow Graph (DFG) where the mobile adders are identified. After scheduling each mobile adder into a control step within its legitimate time window with the minimum opportunity cost, mutually exclusive adders can be merged to reduce the total adder cost by introducing cheaper multiplexers. The opportunity cost for each scheduling decision is assessed by the probability displacement and disparity measures of the scheduled adders in a particular control step, as well as its predecessors and successors in the DFG. The algorithm is runtime efficient as exhaustive search for the best fusion of independently optimized constant multipliers has been avoided. Simulation results on randomly generated 12-bit constant sets show that the proposed algorithm are on average 25% more area-time efficient than the best reported solutions in the literature.

Last but not least, an efficient switching power estimation method for the MCM block of FIR filter is proposed in view of the inefficiency of using gate-level and circuit-level techniques for its power estimation. The proposed Power Index (PI) model divides the total switching activity into two propagation modes for analysis. The inter-adder propagation models the transitions that take place between different discrete adders through the sum outputs and the intra-adder propagation models the transitions that take place within the same discrete adder through the carry chain of the full adders. Due to the existence of shared adders in the MCM block after the elimination of common subexpressions, the load capacitances of the adders vary. The variation of load capacitance has been taken into account by this model. Dynamic power simulations of the benchmark filters in an ASIC design flow show that this PI measure is more closely correlated with the actual dynamic power consumption than other existing switching power estimation models at operator level.

Contents

ACKNOWLEDGEMENTS..... I

SUMMARY II

CONTENTSIV

LIST OF FIGURES.....VII

LIST OF TABLESIX

LIST OF ABBREVIATIONSX

CHAPTER 1 INTRODUCTION..... 1

1.1 Motivation.....1

1.2 Objectives4

1.3 Major Contributions of the Thesis5

1.4 Organization of the Thesis8

CHAPTER 2 BACKGROUND AND LITERATURE REVIEW 12

2.1 Overview of digital FIR filter, direct and transposed direct form..... 12

2.2 Number Representations for Filter Coefficients 15

2.2.1 Binary Number Representation 16

2.2.2 Signed Digit Representation..... 18

2.2.3 Canonical Signed Digit and Minimal Signed Digit Representations..... 18

2.3 MCM Problem and Entropy Theories21

2.3.1 Graph-Dependence (GD) Algorithms22

2.3.2 Common Subexpressions Elimination Algorithms (CSE).....26

2.3.3 Information Theory32

2.4 Reconfigurable Constant Multiplier and Scheduling Algorithms34

2.4.1 Algorithms Optimized for FPGA Solutions36

2.4.2 Algorithms Optimized for ASIC Solutions38

2.4.3 High Level Synthesis Scheduling Algorithms.....41

2.5 Summary.....43

CHAPTER 3 THEORETIC APPROACH TO FIR FILTER DESIGN	44
3.1 Introduction.....	44
3.2 Multi-root Binary Partition Graph.	46
3.3 Information Theoretic Approach to MBPG Minimization	49
3.4 Shift Inclusive Computation Reordering for Adder Complexity Reduction	58
3.5 Experimental Results.....	64
3.6 Summary.....	72
 CHAPTER 4 HIGH-LEVEL SYNTHESIS ALGORITHM FOR DESIGN OF RECONFIGURABLE CONSTANT MULTIPLIER	 74
4.1 Introduction.....	74
4.2 Hardware Reduction by Reconfigurable Constant Multiplications	76
4.3 Proposed High Level Algorithm for RCM Design	80
4.3.1 Design Problem Formulation	80
4.3.2 Common Subexpression Elimination.....	83
4.3.3 Proposed Mobile Operator Scheduling Scheme.....	85
4.3.4 Adder Merging and Multiplexer Splitting.....	89
4.3.5 Design Example	92
4.4 Experimental Results and Discussion.....	96
4.5 Application of RCM in Circular Convolution	109
4.5.1 Present Architecture for Circular Convolution	109
4.5.2 Proposed Architectures for Circular Convolution.....	111
4.5.3 Experimental Results and Discussions.....	116
4.6 Summary.....	118
 CHAPTER 5 NEW POWER INDEX MODEL FOR SWITCHING POWER ANALYSIS	 120
5.1 Introduction.....	120
5.2 Problems with Existing Dynamic Power Estimation Models.....	121
5.3 CMOS Switching Power and Proposed PI Model.....	124
5.3.1 Formulation of Switching Power	124
5.3.2 Switching Activity of Ripple Carry Adder.....	126
5.3.3 Proposed Power Index (PI) Model	127

5.4	An Illustrative Example	131
5.5	Results and Discussion.....	134
5.6	Summary.....	138
CHAPTER 6 CONCLUSION AND FUTURE WORKS		140
6.1	Conclusion	140
6.2	Further Work.....	143
AUTHOR'S PUBLICATION		145
BIBLIOGRAPHY		147

List of Figures

Figure 2.1	Direct form FIR filter structure	14
Figure 2.2	Transpose direct form of FIR filter	15
Figure 2.3	Primitive operator graph for the coefficient set $\{283, 139, 815\}$	22
Figure 2.4	Multiple Constant Multiplication Block in FIR Filter	24
Figure 2.5	Example of common subexpression elimination	27
Figure 2.6	Time-multiplexed implementation of transposed direct form FIR filter	35
Figure 2.7	Basic cell structure used in FPGA mapping algorithms	37
Figure 2.8	(a) DAG fusion of constants 45 and 19 by [TUM07] (b) Alternative DAG for 45.....	40
Figure 3.1	(a) Transposed direct form FIR filter. (b) Adder tree decomposition.....	47
Figure 3.2	(a) Multi-root Binary Partition Graph. (b) Parent and children vertex	47
Figure 3.3	Flow chart of the proposed information theoretic method.....	53
Figure 3.4	Algorithm for vertex decomposition.....	56, 57
Figure 3.5	Generation of minimal MBPG for $H = \{1132, 815, 556\}$	57
Figure 3.6	Full adder cost for addition/subtraction using RCA	59
Figure 3.7	Critical path delay computation	60
Figure 3.8	MBPG after computation reordering (a) $\Delta b_{12}=6$ (b) $\Delta b_{13}=0$, and (c) $\Delta b_{23}=2$. ..	63
Figure 3.9	Normalized AT complexity comparison for Filters D to J. For each filter, the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM	67
Figure 3.10	Normalized AT complexity comparison for FIR1. For each N , the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM	70
Figure 3.11	Normalized AT complexity comparison for FIR2. For each N , the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM	70
Figure 4.1	Reconfigurable constant multiplier (RCM)	75
Figure 4.2	RCM blocks in matrix multiplication	77
Figure 4.3	Recursive Goertzel filter	78

Figure 4.4	Polyphase decimation filters ($M = 3$).....	79
Figure 4.5	Example of RCM desgin (a) DAGs of constant multipliers (b) Reduced DFG (c) Mapping to RCM	81
Figure 4.6	Disparities of operators scheduled in different control steps.....	88
Figure 4.7	RCM design (a) before (b) after multiplexer splitting	90
Figure 4.8	Flow chart of the proposed algorithm.....	92
Figure 4.9	DAG of constant multipliers with $C = \{362, 392, 473\}$	93
Figure 4.10	RCM of the design example (a) DFG (b) Circuit architecture	96
Figure 4.11	AT complexity comparison of proposed RCM, ReMB methods and [TUM07] with RCA (upper) and CLA (lower) adders for data sets A-D.	102
Figure 4.12	AT complexity comparison of proposed RCM and [TUM07] for type A Goertzel recursive DCT with RCA (upper) and CLA (lower) adders	104
Figure 4.13	Comparison of average AT products on randomly generated constant sets, (a) $N = 10$ (b) $N = 20$	106
Figure 4.14	Conventional architecture of circular convolution	111
Figure 4.15	Proposed Type A architecture for circular convolution.....	113
Figure 4.16	Proposed Type B architecture for circular convolution.....	115
Figure 5.1	A CMOS inverter	125
Figure 5.2	Inter-RCA and intra-RCA propagation	127
Figure 5.3	Adder graph annotation and switching activity matrices	129
Figure 5.4	Illustrative example of PI calculation on a 7-tap FIR filter	131
Figure 5.5	Adder graph annotation for illustrative example.....	131
Figure 5.6	Normalized ratios of GPC, GPS and PI to dynamic power for (a) FIR1 (b) FIR2 (c) FIR3 (d) FIR4 (e) FIR5	136, 137, 138

List of Tables

Table 2.1 Summary of some most representative and commonly cited MCM optimization algorithms 32

Table 3.1 Test filter specifications..... 65

Table 3.2 FA cost comparison 65

Table 3.3 FA critical path delay comparison 66

Table 3.4 FA cost comparison of FIR1 and FIR2..... 68

Table 3.5 FA delay comparison of FIR1 and FIR2..... 69

Table 4.1 Traces of computation of the proposed scheduling algorithm..... 95

Table 4.2 Area cost estimation for data set A 97

Table 4.3 Area cost estimation for data set B 98

Table 4.4 Area cost estimation for data set C 98

Table 4.5 Area cost estimation for data set D 98

Table 4.6 Synthesized areas of data sets A-D (in μm^2) 100

Table 4.7 Synthesized delays of data sets A-D (in ns)..... 100

Table 4.8 AT products of data sets A-D (in $\mu\text{m}^2\times\text{ns}$) 101

Table 4.9 Synthesized Area (A)(μm^2), Delay (T)(ns) and AT product ($\mu\text{m}^2\times\text{ns}$) of $2\times\alpha$ multipliers in 8-point/16-point type A Goertzel recursive DCT structures 103

Table 4.10 AT products of FIR filter coefficient sets (in $\mu\text{m}^2\times\text{ns}$)..... 105

Table 4.11 Comparison of estimated area costs based on FIR filter coefficient sets ... 117

Table 4.12 Comparison of estimated area costs based on reconfigurable constant multiple data sets A-D 117

Table 5.1 GPC, GPS, PI and simulated power of FIR1 to FIR5..... 135

List of Abbreviations

ADC	Analog to Digital Converter
ALAP	As Late As Possible
ASAP	As Soon As Possible
ASIC	Application Specific Integrated Circuit
AT	Area-Time
BCSE	Binary Common Subexpression Elimination
CLA	Carry Look-ahead Adder
CLB	Configurable Logic Block
CMOS	Complementary Metal Oxide Semiconductor
CRA	Contention Resolution Algorithm
CSD	Canonical Signed Digit
CSE	Common Subexpression Elimination
DAG	Directed Acyclic Graph
D-AMPS	Digital Advanced Mobile Phone Systems
DCT	Discrete Cosine Transform
DFG	Data Flow Graph
DFT	Discrete Fourier Transform
DSP	Digital Signal Processing
FA	Full Adder
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Arrays
GD	Graph Dependence
GPC	Glitch-Path Count
GPS	Glitch-Path Score
HWP	Hamming Weight Pyramid
LPFIR	Linear Phase Finite Impulse Response
IIR	Infinite Impulse response
ILP	Integer Linear Programming
ITM	Iterative Matching

List of Abbreviations

LD	Logic Depth
LO	Logic Operators
LUT	Look-Up Table
LS	Least Square
LSD	Least Significant Digit
MAC	Multiply Accumulate
MAG	Minimum Adder Graph
MBPG	Multi-root Binary Partition Graph
MCM	Multiple Constant Multiplications
MRAG	Modified n -Dimensional Reduced Adder Graph
MSB	Most Significant Bit
MSD	Minimal Signed Digit
NRSCSE	Nonrecursive Signed Common Subexpression Elimination
PDC	Personal Digital Cellular
PI	Power Index
PM	Parks-McClellan
RAG- n	n -Dimensional Reduced Adder Graph
RCA	Ripple Carry Adder
RCM	Reconfigurable Constant Multiplier
ReMB	Reconfigurable Multiplier Block
RF	Radio Frequency
SDR	Software Defined Radio
SISO	Single Input Single Output
SIMO	Single Input Multiple Output
SoC	System-On-Chip
SOP	Sum Of Product
SPT	Signed-Power-of-Two
TDFG	Time-multiplexed Data Flow Graph
VLSI	Very Large Scale Integration

Chapter 1

Introduction

1.1 Motivation

The electronic industry has witnessed a phenomenal growth over the last two decades, mainly due to the rapid advance of very-large-scale integration (VLSI) technology [FAL05, FEN09]. The number of applications of integrated circuits in high-performance computing, digital signal processing, telecommunications, and consumer electronics has been rising steadily, and at a very fast pace. The economy of scale accompanied with the advancement of radio frequency integrated circuit technology has made ubiquitous computing a reality with infrastructure support of wireless sensor and mobile network. In order to accommodate the demand for high data transmission rates of real-time multimedia applications on wireless mobile communication systems and devices, signal conditioning functions processed in analog hardware are still a common sight. The performance of analog circuits are, however, highly sensitive to various nonidealities, such as dc offset, phase distortions, thermal effect, supply voltage drifts and device non-linearities. Comparing with analog circuit, digital Integrated Circuit (IC) technology is more robust with respect to temperature and process variations [HAR07]. The major drawbacks of digital filters are the cost of production, speed limit and switching power. As technology matures, Application-Specific Integrated Circuit (ASIC) has become more cost effective for some generic classes of problems in digital signal processing. The emerging Software Defined Radio (SDR) ensues the ASIC implementation of the Radio Frequency (RF) and analog front-end signal processing functions to eliminate the sophisticated compensation methods needed to provide the agility to power management and to exploit the robustness of post-processing in digital domain. Filtering and waveform shaping functions are the main performance stumbling blocks immediately after the analog-to-digital watershed in RF systems. Significant design challenges have been posed on the design of the channelizers and frequency selective filters of SDR due

to the additional constraints on area and power requirements when the bandlimited real-time applications are targeted for migration from the fixed station to the mobile terminals [TUT02, HEN02]. To achieve an all-digital or a digital-dominant system architecture, the associated kernel functions like convolution, correlation and linear transformations must be processed at very high data rates, in order to keep up with the IF signal processing at 700~1000 MHz used in [BAR09].

One of the most frequently used application-specific digital signal processors in wireless communications is digital filter. Finite Impulse Response (FIR) filter is an important class of linear time-invariant systems. It can be seen as a computational process or algorithm by which a sampled digital signal or a sequence of numbers is transformed into a second sequence of numbers as the output signal [KAI66]. The operations used to obtain the output signal from the input signal can be completely characterized by their unit-sample (or impulse) response and analyzed by the convolution of two discrete sequences. The arithmetic computations involved in this process are mainly multiplications and additions. Depending on the attenuation characteristics of the filter response, the number and complexity of these operators may grow to an extent that the stringent throughput rate, area and power constraints are difficult to be met by the tight design turnaround time. One approach to overcome this design challenge is to capitalize on the ASIC design flow through low cost advanced CMOS semiconductor fabrication process technology. Design turnaround time and non-recurrent engineering cost can be substantially reduced by exploiting the design automation tool with cell based design methodologies. By leveraging on the downsizing of cell geometry in advanced submicron technology, dedicated architecture can be employed to design high density digital integrated circuit with high throughput rate and low power consumption. Therefore, many researchers have expended their effort in developing VLSI efficient algorithms in recent years to narrow the productivity gap accrued with the desired optimality and design complexity in domain-specific applications. Specific optimization methods dedicated to a well-defined class of application-specific problems can lead to substantial improvements in overall design parameters, such as delay, area and power consumption over what is achievable with standard EDA tools targeted for generic applications. One such generic class of

applications involves solutions to the multiple constant multiplication (MCM) problem [BUL91, POT96]. It is inspired to observe that many kernel functions associated with digital filtering in SDR and real-time multimedia signal processing can be modeled as a MCM problem. Linear transformations, convolution, correlation functions involve multiple multiplications of a variable by different constants, where the computational intensive and power hungry multipliers can be decomposed into shifters and simpler arithmetic operators. While multiplications by constants appear only in a small fraction of general computation benchmarks, mainly for the purpose of scaling, its frequency of encounters has been abundant in application specific computations, like finite impulse response (FIR) digital filters.

Algorithms for the design of low-complexity FIR filters have been extensively studied [DEM95, HAR96, KAN01, LI93, LIM83, PAS99, SAM89, PEI00, PEI02]. The ever increasing ADC resolutions of RF, data acquisition and video signal processing applications have driven the design of recursive and non-recursive digital filters beyond the traditional area and timing minimization. Besides the primary constraints on throughput rate and implementation cost, power dissipation and reconfigurability/programmability are considered as secondary design criteria for benchmarking these applications. For high-speed applications, the use of transposition theorem results in a transposed direct form filter architecture comprising a bank of delay elements and a pipelined sum-of-products arithmetic network. Dedicated hardware optimized for a fixed coefficient set can then be used to meet the stringent goals of sampling rate and power budget but at reduced flexibility and scalability. For applications that require multi-services flexibility on single expandable platform, fast reconfigurable Field Programmable Gate Arrays (FPGAs) such as the recent Virtex FPGA family [HAW96] is a viable implementation platform. Both ASIC and FPGA designs require automation techniques for combinatorial optimization of formidable complexity with drastically different emphasis from software compilers for general purpose DSPs. Therefore, there is a pressing demand to develop new optimization methods that will facilitate synthesis of efficient hardware solutions to MCM problem. The framework of computer-aided design methodologies to be studied in this research form the basis for

exploitation by high-level synthesis tools for the design of high-speed, low complexity and reconfigurable application-specific digital filters.

In recent decades, reconfigurable and programmable architectures have emerged to become an integral part of a general-purpose computing system [AND2000]. They achieve an impressive performance enhancement of one to two orders of magnitude faster than their processor-based alternatives. The success of the ancillary reconfigurability was demonstrated in a computer consisting of a standard processor and an array of reconfigurable hardware [EST02], where the computations in the reconfigurable fabrics were controlled by the main processor. The hardware can be configured to perform a specific task and reconfigured to execute other tasks after its completion. This new generation of computing structure takes advantage of the flexibility of software and the speed of hardware. From 1980s', a number of designs such as Matrix [MIR96], Garp [HAU97], PACT XPP [BAU04], Morphosys [SIN00], PiCoGA [CAP04] and so on have been proposed for dedicated reconfigurable computing architectures. Reconfiguration is advantageous for multi-function architectures, and previous work has shown that converting multi-function cores to several reconfigurable field programmable gate array (FPGA) cores resulted in core area reduction of around 21% and a performance enhancement of 14% [MAC01]. A FIR filter implementation using a single multiply accumulate (MAC) stage and employing real time reconfiguration to change tap values showed up to 37% improvement in clock speed as compared to a static design [HER01].

Scaling is an important operation in digital signal processing, where the magnitude of the input samples are modulated by multiplying it with a given constant. Such amplitude modulation, implemented as constant multiplication, is also frequently found in other contexts of linear transformations, quantization and digital filters. In most of these applications, the input signals are multiplied by a number of different constants whose values are fixed and pre-determined by the system specifications. Spatial redundancies exist among these known constants and the problem of redundancy removal can be collectively modeled as an MCM problem. Many existing techniques minimize the hardware implementation cost by reusing the arithmetic operators that are used to

perform parallel constant multiplications. It is noted that temporal redundancies also exist in the MCM block as not all operators are executed concurrently in some applications. Multiplexers can be introduced to time share the adder resources to remove these temporal redundancies among different constant multiplications. Control inputs can be generated to select different inputs to the multiplexers to produce the desired immediate computations for the required constant multiplier output at different instances. Temporal reuse of adders is a relatively new research topic. There is ample opportunity for new design methodologies to improve the area-time efficiency of reconfigurable multiplier block instead of relying on existing methodologies targeting at conventional constant multiplier blocks. Existing algorithms to improve the time sharing of hardware resources are either based on fixed graph topologies optimized only for FPGA mapping, or by exhaustive search through independently minimized directed acyclic graphs of all preset constants. An elegant heuristic design automation algorithm with reduced search complexity will be a valued alternative to overcome the limitation of the current solutions and the proposed algorithm will be presented in Chapter 4 of this thesis.

Power consumption is an equally, if not more important metric as silicon area and throughput rate for application-specific digital filters embedded in portable devices. Efficient model for dynamic power consumption of this specific class of DSP circuits is a good starting point to analyze and develop power-driven optimization algorithms. Unfortunately, research publications on computer-aided design of digital filters simply used adder cost as the basis for power efficiency analysis. While it is impractical to use fine granularity metric for high-level modeling due to the curse of dimensionality, the general correlation between higher adder cost and higher power consumption is not always justifiable in the presence of other dominating factors, such as increased logic depth and switched capacitances due to the sharing of logic operators. Some high-level models [DEM00, DEM02b] were proposed to estimate the switching activities of FIR filters based on the discrete adder cost and adder depth of the multiplier block. These models were too simplistic to capture the main switching activities and other contributing factors to the deviation of dynamic power consumption between two designs. To advance the study of power-efficient digital filter design, new power estimation model that can be

efficiently computed in graph is needed to better estimate the correlated input switching activities and account for the variation of load capacitances due to the removal of redundancy. The new power model will be proposed in Chapter 5 of this thesis.

1.2 Objectives

The objective of this research is to develop a framework of design methodologies to reduce the area-time complexity of hardware implementation of FIR filters. The algorithms will aim at solutions that are optimized at higher level of design abstraction than the gate and circuit levels. Transposed direct form structure will be targeted in this research due to the elbowroom for complexity reduction in its MCM block and the inherently high throughput rate by virtue of its pipelined architecture. To facilitate efficient application of computational transformations to MCM problem, new data structure will be formulated. The goal is to transform the MCM block into a graph representation to study different forms of redundancies and their dependency in the shift-and-add network. New and efficient design methodology will then be developed to maximally exploit the detected redundancies for an overall complexity reduction.

Owing to the advantages and broad applications of reconfigurable and programmable architectures to MCM problem, new methodology will be proposed to optimize the cost of realizing a dedicated Reconfigurable Constant Multiplier (RCM) by assuming its deployment in a real-time system. The aim is to realize a set of predefined constant multiplications with minimal hardware resources by exploring both the topological and temporal dependencies of the operations. The system's latency will be exploited to define the internal throughput and latency constraint of the RCM so that its overall area-time complexities will be minimized. An algorithm should target for behavioural synthesis to avoid the intensive search required by the existing algorithms [DEM03a, TUM07] for reconfigurable filter design. It is also envisioned that the scope of the proposed solution can be more generally applicable to other design problems besides FIR filters.

Due to the importance of power consumption in VLSI design, the factors that contributing to the dynamic power consumption of the MCM block of FIR filters will be analyzed. The work done in reducing the number of logic operators or logic depth may not always have the same positive effects on the power consumption. To evaluate the extent to which the switching power varied with solutions of different logic operator cost and logic depth, a new power estimation model is to be established in this research. It is not the goal of this research to provide an exact measure, but a metric that is better correlated with the relative switching power differences caused by different optimization approaches to the same MCM problem. By gaining a good insight into the impact of different approach of redundancy removal to dynamic power consumption, it is hope that the proposed power analysis model could serve as a cost metric for future power driven algorithm for digital filter synthesis.

1.3 Major Contributions of the Thesis

The work reported in this thesis aims to provide new algorithm in digital filter design, reconfigurable constant multipliers and operator-level switching power estimation model. The attempts have led to following major contributions.

- (1) A multi-root binary partition graph (MBPG) algorithm for minimizing the area-time complexity of FIR filter

A new data structure, called the Multi-root Binary Partition Graph (MBPG) has been introduced. The MBPG representation for the MCM block of FIR filters is constructed by virtue of the entropy concept from information theory. Based on the direct correlation between MBPG and the circuit topology in this analytical model, a new maximum likelihood disjunctive decomposition algorithm has been proposed to yield a minimized MBPG for the multiplierless implementation of FIR filters. This algorithm can be viewed as an efficient divide-and-conquer approach to decompose a set of vertices that represent the filter coefficients into subgraphs that represent different sign digit patterns. As each isomorphic subgraph represents a common subexpression, both simple and super

common subexpressions are eliminated implicitly in the graph synthesis process. The proposed data structure possesses bit-level information of the coefficients in each vertex. The vertices of the reduced MBPG can be reordered without affecting the isomorphic subgraphs to further reduce the full adder cost. The proposed algorithm has been tested on various benchmark filters reported in literature as well as randomly generated coefficient sets. The experimental results show that the proposed algorithm produces the shortest logic depth architecture with substantially minimized number of logic operators by maximizing the reusability of common subexpressions.

(2) A new high level synthesis approach to design area-time efficient reconfigurable constant multiplier

The second major contribution is the proposal of a new algorithm for the design of area-time efficient RCM based on high level synthesis approach. A new data structure called the time-multiplexed data flow graph (TDFG) has been proposed to model this design problem. The operators and edges in this TDFG have a direct correlation with the adders and shifters of the RCM so that the minimization problem is transformed into a high-level synthesis problem of optimal scheduling and allocation of operators in the TDFG. The mobile operators are scheduled into optimal control steps to maximize the time share as many operators in the same control step as possible. New aggregate distance and disparity measures are defined based on the distribution of signed digits in the addends to evaluate the opportunity cost of a scheduling decision. Experimental results on application benchmarks and random coefficient sets show that the proposed RCM design algorithm is capable to produce a more area-time efficient solution over the existing algorithms for the same problem. This is achieved without the exhaustive enumeration and search for independently optimized coefficient graph, which is required by the latest method to achieve similar solution quality.

(3) New area efficient architectures for circular convolution

Two new architectures for the implementation of circular convolution transformation have been proposed. By replacing the area expensive multiplier block by the proposed

RCM design, these new architectures have much lower area cost. The bounds on the throughput rates and input sampling frequencies of these two proposed architectures have been analyzed. As circular convolution has very similar transfer function as FIR filter, benchmark constants sets and FIR filter coefficients have been used for the experimentation. The reduction of area cost by the proposed architectures has been shown to be as high as 38% over the conventional architecture designed with the parallel multipliers.

(4) New dynamic power correlated power index model to estimate power consumption of MCM block

A new operator-level model for the switching power analysis of MCM solutions is proposed. Unlike the existing models, the proposed Power Index (PI) model has considered the propagation of correlated switching activities of the sum and carry signals within and across adders of different widths, as well as the variation of the load capacitances due to sharing of adders in the reduced adder graph. Dynamic power simulations of several benchmark filters in an ASIC design flow show that this PI model is more closely correlated with the actual dynamic power dissipation than the existing models. Using PI analysis, the relative power performance for many FIR filter solutions designed by different algorithms for the same coefficient set can be easily and rapidly evaluated to winnow the low-power solutions for more accurate but time consuming circuit level simulation.

The above contributions have led to the publications listed in the author's publications at the end of the thesis.

1.4 Organization of the Thesis

The thesis is organized into six chapters. Chapter 1 introduces the motivation and objectives of this research work. The achievements resulted from the works presented in this thesis are summarized into four main contributions in this chapter.

Chapter 2 presents the fundamentals of digital filter, including the direct form and transposed direct form structures of FIR filters. The number systems used in the design of digital filters and reconfigurable multipliers, including the sum of power-of-two terms representation, the CSD representation and the Minimal Signed Digit (MSD) representation are introduced. Several important existing algorithms for the design of the MCM block and the reconfigurable constant multiplier are reviewed and discussed.

Chapter 3 introduces the Multi-root Binary Partition Graph (MBPG) data structure for MCM problem and proposes a new vertex decomposition algorithm to maximize common subexpression elimination. Relevant terminologies in information theory are defined and an information theoretic approach to the minimization of MBPG is described. A working example is used to illustrate the design procedure. Experimental results on commonly used benchmark filters and practical channelizers of wideband radio receivers are evaluated and compared with the existing algorithms.

Chapter 4 introduces a new high-level synthesis algorithm for the design of Reconfigurable Constant Multiplier (RCM). The design is modeled as a time-multiplexed data flow graph and the proposed operators scheduling and allocation algorithm is detailed. Its area-time complexity is evaluated by both benchmark filter coefficient sets and randomly generated constant sets and compared against the well known methods for the RCM design. Finally, two new architectures for circular convolution that make use of the RCM design are presented and analyzed.

Chapter 5 presents a new power index model used to estimate the dynamic power consumption of MCM blocks. The key factors affecting dynamic power are addressed by

in this model. Experiment results are presented to show the correlation of proposed power index model with the actual simulated dynamic power. The results are discussed and compared with two other existing models.

Finally, Chapter 6 reviews the methods proposed in this thesis by highlighting their features and merits, discusses some of the remaining problems and suggests potential relevant topics worthy of further investigation.

Chapter 2

Background and Literature Review

Fixed and reconfigurable digital FIR filters have ubiquitous applications in digital signal processing (DSP), telecommunications, computer graphics and control [AIT93, AVI61, BEN92, CHE99, POT96, VIN05a]. Although the two architectures are functionally distinct, their complexity reduction can be modeled and solved as a Multiple Constant Multiplication (MCM) problem defined in Chapter 1. In fact, the interest of exploring common subexpression elimination for complexity reduction of transposed direct form FIR filter architectures has been rekindled by the inception of a more versatile framework of MCM [POT96] in the late 90s. The quest for low complexity fixed coefficient FIR filters also stems from the fact that the stringency of the channel selection filters increases progressively with the oversampling rate of RF digitalization required to provide a very high dynamic range [VIN06, PRE00, SHO05, KWO07]. The power constraint of a battery powered mobile terminal does not permit the channelization functions to be implemented on programmable digital signal processor. Programmability of FIR filter coefficients, if desired, can be realized by a brute force approach to replace each coefficient multiplier by a general multiplier and some mechanism for coefficient update. Since a general multiplier costs significant area overhead over a constant coefficient multiplier, Reconfigurable Constant Multiplier (RCM) based on time-sharing of hardware resources has emerged as an attractive alternative for “fixed programmability” in recent years [DEM03a].

This chapter provides the necessary background information to understand various key contributions to the above topics. To better appreciate the significance of the methodologies proposed in the later chapters, relevant works and important current arts are reviewed. Section 2.1 provides an overview of digital FIR filter. The frequently

adopted number systems in digital FIR filters and reconfigurable multipliers are discussed in Section 2.2. It is followed by the survey of key algorithms on complexity reduction of the MCM block of FIR filters and information theories in Section 2.3. RCM block optimization algorithms and high level synthesis scheduling approaches are reviewed in Section 2.4. The chapter is summarized in Section 2.5.

2.1 Overview of digital FIR filter, direct and transposed direct form

Practical frequency-selective filters are generally implemented with finite impulse response (FIR) filters in direct or transposed direct form structures. Direct form structure is the most intuitive implementation of the following time domain filter transfer function.

$$y(n) = \sum_{i=0}^{N-1} h(i) \times x(n-i) \quad (2.1)$$

where $x(n)$ and $y(n)$ are the input and output signals, respectively at the n -th sampling instance. N is the total number of taps of the filter and $h(i)$ is the coefficient value at the i -th tap.

A direct form structure is a simple tapped delay line in which each delayed versions of the input signal is multiplied by the tap coefficients, and the results are accumulated to form the filter output, as shown in Figure 2.1. Each block of " Z^{-1} " represents one delay element, or a pipelined stage of the calculation. A FIR "tap" is basically a coefficient multiplier and delay element pair. Therefore, the number of taps, N , indicates the amount of memory required to implement the filter, the number of calculations required, as well as the selectivity of the frequency response. The frequency response of a filter is characterized by the passband ripple, stopband attenuation and transition band width. The transition band is the band of frequencies between the passband and stopband edges. A narrower transition band results in a sharper filter. Increasing the filter taps will lead to higher stop-band attenuation, less ripple, and sharper frequency response.

Fig. 2.1. Direct form FIR filter structure

Generally, for an N -tap direct form FIR structure, N multiplications and $(N-1)$ additions are required. The critical path delay is given by:

$$t_{direct} = t_{mult} + (N-1)t_{CPA} \quad (2.2)$$

where t_{mult} is the maximum delay of the tap multiplier and t_{CPA} is the delay of a carry propagation adder.

The delay of the direct form filter increases linearly with the filter length. This has an adverse effect of high area and power consumption. Hawley, et al. [HAW96] observed that by merging the $N-1$ additions, a direct form structure with a tree of carry-save adders (CSAs) can lower the latency of the accumulator to an order, which is logarithmically instead of linearly increasing with the filter length. i.e.,

$$t_{direct} = t_{mult} + (\log_{1.5} N)t_{FA} + t_{CPA} \quad (2.3)$$

where t_{FA} is the delay of a full adder.

Fig. 2.2. Transpose direct form of FIR filter

The critical delay path of direct form structure can be further sped up by retiming. By applying the transposition theorem, which interchanges the input and the output and reverse all signal flows in a signal-flow graph of single input single output (SISO) system, the transfer function is unchanged [OPP89]. The z^{-1} delay elements can be repositioned to obtain the transposed direct form structure as shown in Figure 2.2.

The critical path delay is given by:

$$t_{transposed} = t_{mult} + t_{CPA} \quad (2.4)$$

Although results from [YE04] indicated an ascendancy of direct form architecture in terms of silicon area, transposed direct form has the advantage in throughput rate. Since the critical path delay is independent of the number of taps, transposed direct form can achieve high throughput without adding extra pipeline registers and has been the preferred architecture for high-speed, high order FIR filter implementation. For this reason, this research focuses mainly on the transposed direct form FIR filter structure.

2.2 Number Representations for Filter Coefficients

Different number systems are available for the constant coefficient representation. Each will invoke a different topology of arithmetic operators for the same optimization

problem. Although it is merely a choice of numerical representation, alternative representations possess varying theoretic features that can be manipulated for simplifying the amount of shift and add operations involved in multiplication of a variable with a set of known constants [AKS08]. The algorithmic performance of a design methodology in terms of area and time complexity as well as the quality of its solution are largely affected by certain features of the selected number representation system, such as the canonicity of the representation, the average number of Signed Power of Two (SPT) terms, the number of recurrent substrings and the ease of identifying these features. Conventionally, binary number is commonly adopted to represent integer constants in digital signal processing due to its simplicity and the direct realization of Boolean expressions in electronic circuitries using basic logic gates. It consists of only two symbols, 0 and 1 positionally weighted with a radix of 2. Negative integers are typically represented in binary number system in 2's complement form to avoid the complicated sign and overflow detection. In hardware implementation, addition and subtraction in two's complement form have comparable cost and the total number of hardware resources can be reduced if the use of subtractors can be easily inferred from the positional notation at design time. To employ both adders and subtractors to reduce the hardware cost, radix-2 signed digit, also called binary signed digit representation is usually used to represent the signed fixed point coefficients in design automation algorithms. In this section, some of the frequently used number representations in digital system design are reviewed.

2.2.1 Binary Number Representation

Binary representation is a way of writing numbers using two symbols only. The symbols are usually written as 0 and 1, and are called binary digits or bits. Although numbers written in binary notation are generally much longer than their decimal equivalents, binary notation is employed in digital electronics and computing because digital computers represent numbers in terms of the presence or absence of an electrical pulse by 1 and 0, respectively. More specifically, the usual base-2 system is a positional notation

with a radix of 2. With these notations, an integer x can be written into its binary representation as

$$x = \sum_{i=0}^{W-1} b_i 2^{W-1-i}, b_i \in \{0,1\} \quad (2.5)$$

where W is the wordlength of the representation and i is the position of the digits with b_0 as the most significant bit.

Owing to its straightforward implementation in digital electronic circuitry using logic gates, the binary system is used internally by all modern computers. However, the first problem of this simple representation is that there is no way to distinguish the polarity of the numbers. Some offset bias has to be added to shift the dynamic range of the operations so that all operands involved are unipolar. To implement the multiplierless MCM block with binary representation, therefore, only additions can appear and hence the total operator cost is higher than the way where subtractors can also be involved [DEM95]. The second problem of binary number is that the representation uses more non-zero digits than the signed digit representations for the same integer number. Although the binary representation of a number is unique and having more potential of forming common subexpression due to its two-symbol characteristic according to [AKS08, MAH08], the advantage of it is often offset by the higher number of nonzero power-of-two terms over signed digit representations. Another important and severe drawback is that using binary number representation for filter coefficients cannot guarantee minimum logic depth of the multiplier block. From the latest research conducted by [AKS08] to evaluate the adder cost and logic depth of the multiplier block using different number representations, it is concluded that binary representation can produce a very similar performance in adder cost as signed digits representations, but with a conspicuously higher logic depth. The postulations made in [MAH08] for justifying the merit of binary coefficient for CSE is controversial, as pointed out by [CHA09] recently. Hence, binary representation will not be considered in this thesis for CSE.

2.2.2 Signed Digit Representation

Signed-digit representation of numbers indicates that digits can be prefixed with a minus sign to indicate that they are negative. A signed digit (SD) representation [AVI61] of radix r is a positional number system with a symmetrical digit set, i.e., $a = (a_{n-1}, \dots, a_2, a_1, a_0)$ with $a_i \in \{0, \pm 1, \pm 2, \dots, \pm(r-1)\}$. With $r = 2$, radix 2 signed digit representations includes three symbols as $\{0, \pm 1\}$ and are widely used for digital arithmetic operations, such as Booth coding and modified Booth coding [KIM07, DAS96, HAS96]. By making use of the three symbols, a radix-2 signed digit number can be represented as a sum of Signed-Power-of-Two (SPT) terms as shown in (2.6):

$$x = \sum_{i=0}^{W-1} s_i 2^i, s_i \in \{0, \pm 1\} \quad (2.6)$$

Unlike binary number, the signed digit representation of a constant is not unique. Different valid representations of the same coefficient differ in terms of the total number of the nonzero digits and their positions. Generally, those representations with more nonzero digits intend to cost more adders in the multiplierless implementation of the coefficient multiplier. Hence, the representations with the minimal number of signed digits should be listed and form a special category. These representations are called the Minimal Signed Digit (MSD), with Canonical Signed Digit (CSD) treated as a special subset of MSD [REI60].

2.2.3 Minimal Signed Digit and Canonical Signed Digit Representations

Minimal signed digits representations (MSD) [ARN93] are crucial in reducing the hardware cost and power consumption of MAC operations, from the point of view that these representations can guarantee minimal number of signed digits used to represent a coefficient. Any arbitrary n -bit 2's complement number can be represented in MSD with no more than $(n+1)/2$ nonzero digits. On average, MSD representation uses 33% fewer nonzero digits than the normal binary number [RIE66]. The finite spaces of n terms power-of-two integers have also been discussed in [LIM91]. It was also shown in [REI60] that the expected number of nonzero digits in an n -bit MSD number in the range $[-1, 1)$

tends asymptotically to be $n/3 + 1/9$. Among MSD representations, the CSD form [REI60] is a specific subset, which possesses the following two main features. First, it has the minimal number of nonzero digits. Second, two nonzero digits are never adjacent in a CSD number, which distinguishes it from the other MSD forms. It is these two features that make CSD a unique representation. Any integer number can have one and only one CSD representation.

To generate a CSD number from its decimal representation, usually the decimal number needs to be converted to the binary form first. Hwang [HWA79] proposed one approach to generate the CSD number from the binary form. Let $B = (b_{n-1} b_{n-2}, \dots, b_0)_2$ and $D = (d_{n-1} d_{n-2}, \dots, d_0)_{\text{CSD}}$ be the number to be represented in binary form and in CSD form, respectively. The following procedure describes the traditional method to find the CSD terms from its binary form:

Step 1: Starting at the least significant bit, b_0 of B , set the index $j = 0$ and the initial carry $c_0 = 0$.

Step 2: Generate the next carry c_{j+1} according to the methods of conventional binary arithmetic by examining the two adjacent bits, b_{j+1} and b_j of the vector B according to the value of the carry-in bit, c_j .

Step 3: Use the following equation to generate the j^{th} digit d_j of the vector D : $d_j = b_j + c_j - 2c_{j+1}$.

Step 4: Increment the index j by one and repeat the procedure at *Step 2* until $j = n$.

Another algorithm [HWA89] for the conversion of the binary number to signed digits representations involves the computation of interim difference and 2's complement to CSD by applying partial T2I conversion (conversion of a number to its inverse-2's complement format). Other approaches [HAS96, LIM91] do the conversion by searching for the nearest SPT terms. In a recent work [XU07a], a new approach is proposed based on the hamming weight of the CSD number. The hamming weight is defined as the number of nonzero symbols in a symbol sequence. The characteristics of hamming weight of a CSD number, which dictates the adder cost, are investigated. A new regularly

structure, named Hamming Weight Pyramid (HWP) [XU07a], was proposed to explore and exploit the interesting properties of the CSD numbers and the relationship between the natural numbers and their hamming weights in the CSD and MSD representation. This formulation enables an efficient method of converting decimal number to its CSD form to be developed, and a more intuitive and succinct structure for representing all CSD numbers of a given word length.

If the non-adjacent property of CSD number can be relaxed in the representation, one or several other MSD representations of the same coefficient can be obtained. As these representations are inconsistent, conversion of MSD from binary is not straightforward as CSD. Some methods have been developed to search and generate all MSD representations from the CSD representation [PAR02, GUS02b, DEM04]. Due to the large number of variants, these algorithms are based on exhaustive search, which severely limit their applications. Other approaches based on look-ahead circuits [DAS96, HER95] and different derivations of CSD [AKO97, HAS96] have also been reported to circumvent the conversion problem. The inconsistency and high search complexity is what make the MSD representations less popular than the CSD representation. On the other hand, the multiple facets of MSD representation for each coefficient also increase the potential to find beneficial subexpressions for the CSE algorithms. All in all, it is always possible to find a good trade-off by limiting the searching space of an inconsistent representation and achieve better CSE within that space. Experiments run on random coefficient sets for CSE using binary, CSD and MSD representations [AKS08] concluded that CSE algorithm applied on MSD coefficients could produce solution with lower operator cost than the same algorithm based on CSD coefficients and have minimum delay.

In this research, CSD and MSD representations are considered for the proposed design methodologies of MCM and RCM blocks. The details are presented in Chapter 3 and Chapter 4 of this thesis, respectively.

2.3 MCM Problem and Entropy Theories

In high-level synthesis of VLSI circuits, the use of transformations has led to the development of powerful design automation algorithms for many compute intensive DSP problems. One good example is the transformation of the coefficient multiplication operations in the transposed direct form FIR filter structure into a problem faced by a generic class of applications, known as the multiple constant multiplications (MCM) [BUL91, POT96]. MCM refers to a group of scalar multiplications of different constants with the same variable. The MCM block in transposed direct form FIR filter is shown in Figure 2.3. Many kernel functions associated with wireless communication and multimedia digital signal processing can also be modeled as a MCM problem, such as linear transformations, convolution and correlation functions. Reducing the cost metrics, such as area, critical path delay and power of the MCM block by substituting the compute intensive multiplications by a network of adder/subtractors and shifters with a reduced number of logic operators and logic depth form the basis of various design automation algorithms [DEM98, LI95, MEH95]. For a fixed coefficient set, shifters can be implemented directly and incur no logic cost, so the hardware complexity and throughput are determined mainly by the number of adders used and the adder steps. It is therefore feasible to derive optimization methods that can facilitate automatic generation of silicon-efficient solutions by utilizing the redundancy among the constants such that the resource requirements can be minimized. Existing algorithms for optimizing the design of MCM block can be broadly classified into two main categories, depending on the subexpression representation and the number of non-output fundamentals used for the optimization, dimension of subexpression shifts (in time and spatial distance), and the primary optimization criterion.

Fig. 2.3 Multiple Constant Multiplication Block in FIR Filter

The first and earlier category of MCM optimization algorithms is the graph-dependence (GD) algorithms [BUL91, DEM95, DEM02a, GUS02a, KAN01]. These approaches establish a primitive operator graph for every partial sum and produce the graphs of filter coefficients with minimum adder distance. Another category of approaches which is becoming very popular in recent years is the common subexpression elimination (CSE) algorithms [MAH07, XU05b, HAR96, PAS99, PEI02, POT96]. A common subexpression is a signed digit or binary substring that replicates itself in one or more coefficients. The common subexpressions are sought and their frequencies of occurrences are determined. The most frequent common subexpressions are then eliminated iteratively to maximize the sharing of the products of the input with these common subexpressions. Various GD and CSE algorithms will be reviewed and their core ideas will be discussed in the following subsections.

2.3.1 Graph-Dependence (GD) Algorithms

The graph-dependence (GD) algorithms [BUL88, BUL91, DEM95, DEM02a, GUS02a, KAN01, XU04, XU05a] are based on graph data structure. They do not rely on specific integer number representation but the numerical values of the coefficients and partial sums to reduce the number of elementary arithmetic operations which are required to realize the MCM architecture. The concept of primitive operator and its notations were first introduced by Bull and Horrocks [BUL88, BUL91], who are the pioneers to apply

graph theoretic algorithm to systematically minimize the implementation cost of the multiplier block of transposed direct form FIR filters. One example of a primitive operator graph for the set of constants, {283, 139, 815} is shown in Figure 2.4. The MCM block was referred to as multiplier block when this graph dependence algorithm was introduced. The synthesis of filter coefficients is fully specified in the graph whereby the partial sums called the fundamentals are symbolically encapsulated in the vertices of the graph and are multiplied by a power-of-two factor annotated on the edges. GD algorithms involve the synthesis of a set of connected graphs from the unity source to the sinks, which are the filter coefficients. The fundamentals are generated one at a time depending on the previously generated fundamentals. Such logical graph not only reflects the arithmetic relationship between operators but also the delay of any path in the graph in terms of the delay of a single operator if all operators are assumed to be of equal complexity. Three main constraints of the original primitive operator graph approach [BUL91] have been identified in [DEM95]. First, the partial sums are generated with values less than or equal to the coefficient only, but not exceeding. Second, the fundamental set contains even-valued partial sums. It can actually be reduced to a smaller set of odd-valued partial sums by shifting out the trailing zeros. Lastly, the coefficients are processed in an order which is dependent on numerical values of the coefficients. Improvements have been made in subsequent research to overcome these constraints. One of the important modifications made to the Bull and Horrocks (BH) algorithm is the use of subtractors to reduce the number of logic operators needed. This enhancement shares the same principle as the use of signed digit representation for filter coefficients. Another two major improvements to the basic BH algorithm are the reduction of each partial sum by factors of 2 until it becomes odd, and finding a good order of processing the coefficients. BH algorithm with the above enhancements is referred to as the BHM algorithm [DEM95].

Fig. 2.4 Primitive operator graph for the coefficient set {283, 139, 815}

Based on the concept and data structure of primitive operator graph, a well-known and representative algorithm called the n -Dimensional Reduced Adder Graph (RAG- n) algorithm was proposed [DEM95]. The RAG- n algorithm consists of two main parts: the *optimal part* and the *heuristic part*. If one set of coefficients is completely synthesized by the optimal part of the algorithm, minimum adder cost is assured. The main idea is therefore to synthesize as many fundamentals and coefficients as possible with the optimal part of the algorithm. The solution in terms of adder cost can be guaranteed to be optimal if all coefficients can be synthesized in the optimal part. Only when the coefficients cannot be synthesized by the optimal part will the heuristic part be explored. Under those circumstances, the solutions generated will be sub-optimal in terms of the number of logic operators. In the heuristic part, a new measure of ‘adder distance’ is introduced. The adder distance of a new vertex from an incompletely synthesized graph is the number of extra adders that are needed to generate this vertex from the existing vertices of the incomplete graph. The aim is to find the minimum adder distance between the fundamentals generated by the optimal part of the algorithm and the coefficients that have yet to be synthesized [DEM95]. The adder distance needs to be checked by using precomputed lookup tables generated by an exhaustive search of all possible graph topologies. Minimum Adder Graph (MAG) algorithm [DEM94] was typically used to generate these lookup tables. Soon after the introduction of RAG- n algorithm, Dempster and Macleod developed several other algorithms for one-constant [DEM94, GUS02a] and two-constant [DEM05a] multiplication problems. The above GD algorithms assumed

the use of conventional carry propagation adders (CPAs) for their implementation. Gustafsson [GUS01b, GUS04] proposed modified algorithms based on RAG- n by considering the number of carry-saved adders (CSAs) as opposed to the number of CPAs as the criteria for the optimization of fixed-coefficient filters. The use of CSAs speeds up the critical path by avoiding the carry propagations for multi-operand addition except for the final stage. These algorithms are particularly useful for very high-speed and high precision applications.

Since finding the minimum adder distance for an arbitrary number is itself an NP hard problem, there is a limit imposed on the wordlength of the coefficients and the sizes of the lookup tables. The search complexity is a major constraint of RAG- n algorithm. The limit of coefficient wordlength has been lifted by a quasi-minimal adder distance search method using the hamming weight pyramid (HWP) [XU07a] and the utilization rate of the optimal part of RAG- n algorithm has also been improved by the same group of authors in [XU04, XU05a]. Another major drawback concerning most of the GD algorithms is that they synthesize coefficients sequentially, and the hardware complexity are greedily reduced with little or no regard to other optimization criteria. Generating new partial sums with the fundamentals which have high adder step will increase the logic depth of the MCM block. The asymmetric adder tree makes long critical path delay and creates more opportunity for the generation and propagation of glitches [DEM98, KAN01]. In view of this, some algorithms, such as SLBHM and SLRAG- n were proposed by Kang and Park [KAN01]. They are founded on BHM and RAG- n algorithms by incorporating tree reduction method to limit the growth of logic depths. Dempster et al. [DEM02a] also proposed an algorithm called C1 to limit the search space with constrained logic depth. Basically, the search for cost-1 partial sums has been exhaustively performed to reduce the logic depth. If the requirement on absolute optimal solution of adder cost can be relaxed, C1 is capable of generating much lower logic depth solution than RAG- n with slightly higher hardware cost. However, this is achieved at the expense of very high computational complexity, which severely limited the applicability of the algorithm for large problem size.

In summary, GD algorithms operate on the values of the integer coefficients with no regards to its specific number representation. This can help to generate more possible compositions of the coefficients from the intermediate partial sums for optimization. However, as the search space enlarges, the already complex problem becomes daunting with the additional dimension of decision trade-offs. In GD algorithm, the coefficients are synthesized with a primitive operator graph data structure where intermediate partial sums are generated at minimal adder cost by maximizing the use existing partial sums in fewer shift-add operations. However, as each individual coefficient is sequentially processed, the result is inherently order-dependent for multiple coefficients in an MCM problem because the partial sums are generated according to the order of which the coefficients are processed. When searching for new partial sums with minimal adder distance, only existing partial sums from the processed coefficients are eligible. If the coefficients cannot be synthesized in the optimal part, the order of evaluation can cause a local minimum problem despite the exhaustive search. As will be seen in the next subsection, another feature that distinguishes GD-based and CSE-based algorithms is the dimension of shift. GD algorithms consider only one dimension of partial sums due to the search complexity but some CSE algorithms are able to detect redundancy over two dimensions, from the spatially shifted partial sums and the time delayed partial sums.

2.3.2 Common Subexpressions Elimination (CSE) Algorithms

The second main category is Common Subexpression Elimination (CSE) based algorithms. CSE has historically been adopted in computer-aided design in high-level synthesis and has now become a very popular dedicated approach to realizing VLSI efficient FIR filters in recent years [HAR96, JAN02, PAR02, PAS99, PEI02, POT96, XU05b, XU07b, YAO04, MAH08, ASK08]. The main idea behind all these algorithms is the maximization of the sharing of common subexpressions, or the elimination of replicated subexpressions in the MCM block by pattern matching. Fig. 2.5(a) shows an example of eliminating the common subexpression, $10\overline{1}$, which has a frequency of occurrence of 3 in a given coefficients set. This subexpression is eliminated by replacing

the pattern $1\ 0\ \overline{1}$ with a symbol S_1 . Together with other signed digits, higher weight common subexpressions can be detected and eliminated as shown in Fig. 2.5(b).

The objective function and the representation of the filter coefficients play important roles in the quality of the final solutions synthesized with CSE algorithms. The hardware complexity of the final implementation for a given filter specifications depends not only on the cost function used to direct the coefficients synthesis process, but also the number system chosen to represent the filter coefficients [PAR01]. Due to the necessity to fix the positional notation in order to seek for good common subexpressions, the popular signed digit representations, CSD [HAR96, JAN02, PEI02, LIU01, PAS99, TAN02, XU05b, XU07b] and MSD [PAR01, PAR02] representations have been widely adopted by CSE algorithms as they possess the minimum number of nonzero digits. As discussed in Section 2.2.2, one significant difference between CSD and MSD is that CSD is a unique representation while MSD may appear in more than one form for the same integer coefficient. Recently, research on CSE algorithm based on binary number system was also reported [MAH08, ASK08]. It was claimed that the use of binary number representation has the potential of finding more common subexpressions as it has only two different symbols.

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & 0 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & S_1 & 0 & 0 & 0 & S_1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & S_1 & 0 & -1 & 0 & 0 \end{bmatrix}$$

(b)

Fig. 2.5. Example of common subexpression elimination

Besides categorization by number representations, the types of common subexpressions or reusable partial sums to form the complete set of coefficients can also be used to characterize the various CSE algorithms reported in the literatures. There are three different kinds of common subexpressions that can be exploited by various CSE algorithms. By representing the set of CSD coefficients in a two-dimensional table [HAR96], common subexpressions can be extracted by identifying horizontal (intra-coefficient), vertical and oblique (inter-coefficient) groups of identical pattern cells. Intra-coefficient common subexpressions are formed by nonzero digits within the same coefficient and inter-coefficient common subexpressions are formed by nonzero digits across different coefficients. However, it should be noted that coefficients of linear phase FIR (LPFIR) filters are symmetric. By symmetry, it implies that half of the coefficients that are identical in the MCM block can be shared directly in the transposed direct form filter structure. Inter-coefficient common subexpressions, which necessitate the insertion of delay elements into the generator of common subexpressions, could destroy the symmetry of the transposed direct form filter structure. Hence, for LPFIR filters, the search for redundant computations in MCM block is normally confined to horizontal common subexpressions [GUS02b, HAR96, PAS99, PEI02, POT96, XU05b]. Jang and Yang [JAN02] showed by means of specific filter example that vertical common subexpressions transcend horizontal common subexpressions in LPFIRs as adjacent coefficients have similar pattern in the most significant bit (MSB) portion. However, Vinod [VIN03] argued that elimination of vertical common subexpressions alone does not guarantee greater hardware savings over the conventional horizontal CSE methods in most practical LPFIR filters. Based on the statistic, they proposed the extraction of the four most common horizontal subexpressions of hamming weight-2 before the remaining non-zero digits are considered for vertical common subexpressions. To date, only one published algorithm [XU05c] has simultaneously utilized all intra- and inter-coefficient common subexpressions, including the oblique common subexpressions and the higher-weight common subexpressions of all sorts to reduce the adder costs of the MCM block.

From 90's, the research on area efficient CSE algorithms has intensified. The seminal method to CSE based on the CSD representation was first proposed by Hartley [HAR96].

The frequency of occurrences of the same subexpression between different coefficients is considered in this algorithm. Multiple coefficient multipliers can be optimized simultaneously by this algorithm. Similar to many other CSE algorithms, Hartley algorithm is recursive. This algorithm demonstrates that by allowing the common subexpressions spanning different delayed copies of the input to be shared, it can lead to better results than are otherwise impossible. Besides Hartley algorithm, there is another very similar algorithm which is often cited as Pasko [PAS99] algorithm. This algorithm also uses CSD representation. Comparing with Hartley's algorithm, two main differences are observed. First, Pasko does not consider vertical/oblique common subexpressions across more than one coefficient. Second, Pasko searches and eliminates the common subexpressions starting with those that have the highest hamming weights, and breaks them down to smaller ones subsequently, whereas Hartley searches for the smallest common subexpressions and combining them into larger ones. Another well known CSE algorithm is called the ITM algorithm [POT96]. The minimum number of shifts needed has also been taken into account in ITM. A specific signed digit formation is utilized in this algorithm. This method finds the maximum number of coincidences between two signed digit coefficients. It looks for the same non-zero digits with positions within the coefficients for the least number of shifts. These methods [HAR96, POT96] tried to improve the optimization by considering either more factors than frequency of occurrences or by extending the search space. However, the improvements of the corresponding solutions are not as significant as expected. This is because gradient decent approaches are sensitive to the execution order even though they are relatively efficient. There is no feedback to allow efficient resubstitution of lower cost common subexpressions in case of conflicts arising from the nullification of potential subexpression elimination caused by an early decision in the execution process.

In the twenty-first century, a number of ingenious heuristic CSE algorithms [YAO04, XU05b, XU05c, XU07b] has been proposed in addition to Hartley, Pasko and ITM. Most of these algorithms are based on CSD representation. While the canonical CSD representation minimizes the search complexity, it also limits the exploration of common subexpressions. To resolve this problem, the algorithm proposed by Yao et al. [YAO04]

relinquished the fixed number representation and allows mutual conversions between signed digit representations and CSD representation in the process of composing common subexpressions. However, this relaxed number representation increases the search complexity tremendously and demands a lot of computing power. A better and new contention resolution algorithm (CRA) [XU05b, XU05c] has been proposed by the author's research group. It uses a new data structure called the admissibility graph to allow the poor decision of common subexpressions to be reverted in case of contention with a good common subexpression in the progress of synthesizing the optimal solution. This algorithm can be calibrated to allow logic depth driven synthesis. In CRA, the logic depth will only be allowed to increase if there is added savings by considering higher hamming weight common subexpressions.

The above algorithms search recursively for common subexpressions. Yet another method called Nonrecursive Signed Common Subexpression Elimination (NRSCSE) algorithm [PEI02] was proposed. It searches for the nonrecursive signed common subexpressions that must be eliminated from the original CSD array. In this method, the hamming weight range of searched common subexpressions is not fixed, but treated as a user specified input parameter. Once the highest weight of the common subexpressions is determined, all common subexpressions generated are restricted to this hamming weight limit. If the weight of subexpressions is confined to two, minimal logic depth can be achieved for the designated multiplier block of the FIR filter. NRSCSE uses this common subexpression hamming weight as a critical cost metric to control the trade-off between the logic operator and the logic depth.

Besides CSD, other number representations are also considered for the search of common subexpressions. One algorithm that uses the MSD number system is the PK algorithm [PAR02]. As MSD representation offers several possible signed digit patterns for a particular coefficient or partial sum value, the opportunity of finding sharable subexpressions with other coefficients and partial sums is higher. This unique characteristic can lead to a more efficient solution if an appropriate MSD representation

is selected for each coefficient of the filter. The PK algorithm relies essentially on the selection of the correct forms of MSD coefficients to reduce the logic complexity.

Recently, Mahesh and Vinod [MAH08] proposed another new CSE algorithm called BCSE which has adopted the binary number representation for the coefficients. This algorithm performs both vertical and horizontal searching and eliminations on weight-two and higher weight common subexpressions that exist in the binary representation of coefficients. The quantitative figures used to justify the merits of binary representation for CSE were shown to be inaccurate and some errors in the reported logic depth also caused the delay penalty to be less visible. These inexact arguments and oversights have recently been pointed out in [CHA09]. It is claimed that using binary representation can actually help to exploit more potential common subexpressions over CSD or MSD representation and without logic depth penalty. However, it has been shown in [CHA09] that binary representation can not guarantee minimum logic depth as CSD and MSD representations. This severe drawback of binary representation on logic depth has been deemphasized by the errors in logic depth results claimed in [MAH08]. In addition, the results reported in [AKS08] also indicate that although binary representation could have slight advantage over CSD in terms of adder cost, CSD or MSD rather than binary representation should be used for high-speed application in view of the logic depth constraint.

One other recent development is pertaining to the refinement of logic operator and logic depth measures for performance benchmarking by considering the number of full adders as opposed to number of adders. This is because the adders in the MCM block are not be of equal complexity due to the amount of shifts involved in the generation of partial sums. The wordlengths of the adders used are different, which will affect the actual implementation cost, delay and power consumption. Some examples of algorithms that consider such fine-grain FA cost include the recently introduced differential coefficient method, coefficient partitioning method and the use of pseudo floating point representation [VIN05b, VIN06].

The following table summarizes the feature of some most representative and commonly cited existing algorithms. In this table, HAR, PAS and NRS denote Hartley algorithm [HAR96], Pasko algorithm [PAS99] and NRSCSE algorithm [PEI02], respectively. Bin is the abbreviation for binary.

Table 2.1 Summary of some most representative and commonly cited MCM optimization algorithms

	GD algorithms				CSE algorithms						
	BH	BHM	RAG	C1	HAR	ITM	PK	PAS	NRS	CRA	BCSE
Objective	LO	LO	LO	LO/LD	LO	LO	LO	LO	LO/LD	LO/LD	LO/LD
Number system	Any	Any	Any	Any	CSD	CSD	MSD	CSD	CSD	CSD	Bin
Starting CSE Hamming weight	2-bit	2-bit	2-bit	2-bit	2-bit	Max	2-bit	Max	Flexible	2-bit	2-bit/3-bit
Coefficients generated sequentially	Yes	Yes	Yes	Yes	No	No	Yes	No	No	No	No

2.3.3 Information Theory

The detection of beneficial common subexpressions to maximize adder sharing without exhaustive enumeration is a probabilistic decision making process based on the statistics of different types of conflicting common subexpressions. After reviewing many existing CSE algorithms, it is conjecture that an effective heuristic algorithm to MCM optimization problem can possibly be founded on the well established information theory with the elegant mathematical underpinning of probability and entropy.

A basic assumption of probability theory is that given enough information, the status of any event can be reduced to a certainty. Randomness is therefore due to the absence of information, and the prediction of an event becomes subjective. The probability distributions assigned to events always represent the lack of information and incomplete

knowledge of them. A person who possesses different information would assign different probabilities to the same event. Probability theory is not about absolute truth, but about plausible inferences which are consistent with certain axioms. It cannot tell how often an event will actually occur in practice. It applies when the only known event is A , and a usable measure of the information can be developed which is obtained from observing the occurrence of an event having probability p . Any particular features of the event will be ignored, and only observe whether it happened. In essence this means that one can think of the event as the observance of a symbol whose probability of occurrence is p . Another way to express this lack of confidence in the estimation is that all probabilities are conditional. In many derivations, these conditions are omitted for brevity, but they exist and affect the probabilities. Conditional information defines the sample space, and different sample spaces are implied by different information content. The logical probability statement $P(A|B) = p$ has a meaning that whenever B is true then A is true with certainty p . This applies regardless of any other conditional probabilities of A and B .

The entropy used to develop the proposed algorithm in Chapter 3 is a probabilistic measure of uncertainty or ignorance. Information, on the other hand, is a measure of a reduction in that uncertainty [HEY01]. Entropy (or uncertainty) and its complement, information, are the most fundamental quantitative measures in cybernetics obtained by extending the more qualitative concepts of variety and constraint to the probabilistic domain. Variety and constraint, the basic concepts of cybernetics, can be measured in a more general form by introducing probabilities [ASH56]. Assume that the set of all possible states is S and the precise state of a system, $s \in S$, is unknown, but only the probability distribution $P(s)$ that the system would be in state s is known. The variety can be expressed as an entropy, H (as originally defined by Boltzmann for statistical mechanics) [KHI57] as

$$H(P) = - \sum_{s \in S} P(s) \cdot \log_2 P(s) \quad (2.7)$$

Entropy reaches its maximum value if all states are equally probable, that is, if no indication whatsoever to assume that one state is more probable than another state. Like variety, H expresses our uncertainty or ignorance about the system's state. It is clear that

$H = 0$, if and only if the probability of some state is 1 (and those of all other states are 0). In that case, it becomes the maximal certainty or there is complete information about what state the system is in. Constraint is defined as one that reduces uncertainty, that is, the difference between maximal and actual uncertainties. This difference can also be interpreted in a different way, as information. Historically, the entropy H was introduced as a measure of the capacity for information transmission of a communication channel [HEY01]. Indeed, if some information about the state of the system is obtained (e.g. through observation), then this will reduce our uncertainty about the system's state, by excluding, or reducing the probability of a number of states. The information, I received from an observation is equal to the degree to which uncertainty is reduced, i.e.,

$$I = H(\text{before}) - H(\text{after}) \quad (2.8)$$

If the observation completely determines the state of the system ($H(\text{after}) = 0$), then the information, I reduces to the initial entropy or uncertainty, H .

The above probability and entropy theories nicely coincide with the uncertainty of finding the common subexpressions for maximal sharing in MCM problem given an initial state. Once coefficients are expressed in signed digits representation, a single digit can often have more than one possibility to form a subexpression by grouping with other signed digits. If one signed digit is fixed in certain common subexpression, its probabilities of forming other subexpressions become zero and the probabilities of occurrences of all correlated common subexpressions will be changed. This is the inspiration that triggers the connection between MCM optimization and the entropy theory and leads to the development of a new CSE algorithm in Chapter 3.

2.4 Reconfigurable Constant Multiplier and Scheduling Algorithms

By time sharing the resources used for the implementation of the multiplier block, designs with higher efficiency over traditional multiplier block can be achieved. This kind of time-multiplexed implementation of the transposed direct form FIR filter is

shown in Fig. 2.6. The resultant multiplier block is called the Reconfigurable Multiplier Blocks (ReMB) in [DEM03a]. In this thesis, the research will study a specific scaling problem frequently encountered in digital signal processing where a sampling input signal is scaled by several predetermined constants at different instances. The application scenarios involved are less restrictive than the multiplier block of digital filter. A reconfigurable architecture controlled by the multiplexer selection signals to remove the temporary redundancy is desired. In this thesis, therefore, a different terminology of Reconfigurable Constant Multiplier (RCM) is used to treat this functional unit as a reconfigurable operator. This RCM block can achieve significant area saving by temporal resource sharing, but there will also be an inevitable compromise in delay when applying RCM to adapt an FIR filter to multiple frequency responses. RCM can also be applied to any numerical computation problems that involve the multiplication of an input variable with one of the N predetermined constants where the delay penalty is not a major concern or it will be absorbed by other more critical paths. One example is the implementation of the recursive part of DCT structure as shown in Figure 6 of [DEM03a]. The arithmetic operators in the multiplier block can be saved at the expense of some additional hardware for memory, sequencing and multiplexing. The objective of RCM is thus to minimize these control overheads and adder redundancy to reduce its overall hardware complexity. Several design automation algorithms for area efficient RCM block have been proposed in recent years. These algorithms can generally be categorized into two main groups. They will be discussed and compared in what follows.

Fig. 2.6. Time-multiplexed implementation of transposed direct form FIR filter

2.4.1 Algorithms Optimized for FPGA Solutions

The first group of ReMB design automation algorithms [DEM03a, DEM04, DEM05b, DEM05c, DEM07] targets at FPGA implementation. These methodologies are optimized based on fixed graph topologies and rely on intensive search to obtain the best solution. These methods, similar to the GD algorithms for MCM block, use integer coefficients that assume no specific number representation. The enormous number of possible compositions of the partial sums of different coefficients increases the search complexity dramatically. Therefore, the graph patterns are constrained by certain basic cell and macrocell structures of the FPGAs.

In 2003, Demirsoy et al. first proposed the basic concept and guidelines for the design of ReMB block [DEM03a]. The basic structure or basic cell adopted for FPGA implementation is shown in Fig. 2.7. It consists of a 2-1 multiplexer connected to an arithmetic operator. The operator can be either an adder, subtractor or adder/subtractor. This basic cell can be easily mapped to FPGA with piggyback hardware cost for the 2-1 multiplexers. For example, each Configurable Logic Block (CLB) of Xilinx Virtex device family contains at least a four-input Look-Up Table (LUT). A full-adder can be implemented using one LUT and it is also possible to fit the 2-to-1 multiplexer to the same LUT to minimize the number of CLBs required to map one bit slice of the basic cell of Fig. 2.7 [DEM04]. With different control signals of the multiplexer, this basic cell can be reconfigured to generate more than one partial sum, leading to better area utilization. The design guidelines proposed by [DEM03a] can be summarized as follows:

- All possible ways of constructing a coefficient by grouping existing partial sums are searched and listed.
- Partial sums that have common input value and shift amount are grouped into a basic cell structure.
- The outputs of the newly constructed basic cells are combined with the existing partial sums to construct the remaining coefficients.

Fig. 2.7 Basic cell structure used in FPGA mapping algorithms

Based on the design guidelines proposed in [DEM03a], Demirsoy et al. [DEM04] demonstrated the efficient implementations of ReMB block by presenting their logic synthesis results and comparing them against the conventional implementation. A half-band 32-tap FIR filter is implemented with the proposed ReMB block and synthesized using a Xilinx Virtex FPGA flow and an ASIC flow using the UMC $0.18\mu\text{m}$ CMOS standard cell library. Reference designs have also been built by deploying standard time-multiplexed architectures and off-the-shelf Xilinx Core Generator system for the FPGA design. Area cost and delay of these designs are compared and it is shown that the ReMB technique can significantly reduce the area for the multiplier circuitry and the coefficient store, as well as the delay [DEM04]. Typically, the use of ReMB technique can reduce the area requirement of FPGA by around 20 % over the implementation with conventional multipliers.

The design methodologies in [DEM03a, DEM04] are developed for Single Input Single Output (SISO) ReMB block using basic design guidelines. Alternative systematic synthesis method [DEM05b, DEM05c, DEM07] has been proposed for both SISO and Single Input Multiple Output (SIMO) ReMB designs using a different notion and approach. This algorithm builds a graph table for each coefficient based on the basic cell depth of the output nodes. From these graph tables, the algorithm searches for basic cells of the same depth and merges them into one single cell under one node. All inputs of cells that merged into a single node are attributes of the node definition. These node definitions are then given scores according to the utilization of the basic structure and the formation of its fundamental sets. Starting with the highest scored node definition, one

node definition is selected for each node such that the commonality of the fundamental sets is maximized. In this way, the area of the ReMB block can be optimized by searching for the node definition with highest total score for each basic cell node. The optimal solution for the SIMO ReMB design can be produced by executing this procedure recursively.

All the above ReMB design algorithms rely on the dedicated cell structure of the FPGA where the 2-1 multiplexer comes ‘free’ as a piggyback component of a macro cell. However, the constraint of mapping to only these dedicated reconfigurable cells also limits the opportunity of sharing individual fundamental logic cells freely in ASIC implementation. In recent years, some researchers started to work on area efficient reconfigurable multiplier design methodology for ASIC implementation without imposing any restriction on the basic design components.

2.4.2 Algorithms Optimized for ASIC Solutions

One very important algorithm that searches for the most optimal fusion of directed acyclic graphs (DAG) to realize the ReMB block was proposed in 2007 [TUM07]. It is referred to as DAGfusion in this thesis. This is the first algorithm that does not rely on the specialized cells of FPGA and is optimized for ASIC technology mapping. In DAGfusion, the ReMB problem is modeled as a generic but time-multiplexed multiplication of a fixed-point input value and one of several selective fixed-point constants. These constants are represented in integer and every integer is expressed as a fixed DAG from an existing algorithm [GUS06]. The core idea is to find the best fusion of the DAGs of all coefficient multipliers. The best solution from among the possible DAGs for the individual coefficient multipliers with the least estimated area cost will be selected as the final DAG for that coefficient. A cost function based on the estimated area of basic cells in an ASIC standard cell library is adopted to guide the selection. Any node in the DAG that has a fanin greater than two mandates a multiplexer insertion. The design of the ReMB is completed by inserting all the required multiplexers and transforming all nodes in DAG into adders/subtractors. With no constraint on the basic cells, DAGfusion is capable of achieving significant saving in area cost in ASIC implementation over all

previously reported algorithms targeted for FPGA implementation, according to the results on benchmark coefficients and random coefficients reported by [TUM07]. However, there are two obvious weaknesses of this algorithm. First of all, the DAG representations for constants are determined and fixed by another algorithm. These DAGs are generated sequentially and individually without a holistic consideration of subexpression sharing in the complete set of constants. The DAGs generated in this manner is local minimal as the duplicated partial sums or common subexpressions for the entire constant sets are not fully exploited. One good example is the fusion of DAGs for the constants, 45 and 19. DAGfusion will fuse the pre-determined DAGs for 45 and 19 which are the first and second graphs from the left of Fig. 2.8(a). The fusion graph is shown as the third graph in Fig. 2.8(a). However, there are other DAG representations for the same constant, like 45 in this example, that can lead to better fusion although the DAG itself may not be minimal. The given DAG generated by [GUS06] can guarantee the minimal adder distance for this single constant, but it may not be the most area efficient choice for the whole constant set. By looking at the alternative DAG of 45 in Fig. 2.8 (b), it is noticed that this DAG possesses a partial sum of 3, which can be shared with the same partial sum in DAG of 19. This simple example shows that some common subexpressions across different constants can be easily omitted by the generation of individual DAG independently before the fusion takes place. Secondly, the search complexity to cover all possible fusion solutions is huge. It becomes intractable when the problem size increases. Limiting the number of iterations in this search will compromise the quality of the solution as the probability of finding a non-optimal fusion is higher.

Fig. 2.8 (a) DAG fusion of constants 45 and 19 by [TUM07] (b) Alternative DAG for 45

Kim et al [KIM07] proposed another algorithm for ReMB design based on Booth encoded multiplier. Similar to [TUM07], this algorithm also aims at ASIC realization with no restriction on the basic cell structures. Instead of integer, binary representations are used for all coefficients and they are listed in a coefficients table. The binary numbers are then Booth encoded into signed digits where shared partial products are identified. To increase the sharing of partial products, the encoded digits are modified to increase the number of zeros, because zeros can be changed to any desired values by a compensation signal [KIM07]. Subsequently, the zeros are changed into other values such that the selected column is the same as one of other columns in the encoded digit table. However, this approach provides no new insight into the area reduction over DAG or adder fusion for the design of ReMB block. It basically performs the generic common subexpression elimination, but in an encoded digit table rather than the conventional binary or CSD coefficients arrays.

2.4.3 High Level Synthesis Scheduling Algorithms

The design methodologies for ReMB block reviewed in the previous two sections assume either dedicated basic cell or pre-determined graph topologies for DAGs which can lead to local minimum problem in the process of optimization. If these constraints are removed and the scheduling and allocation of adders and multiplexers are considered holistically from the generation of the DAGs to the final ReMB mapping, more optimal area-time results can be expected. Such considerations are often found in the context of high level synthesis. In this section, some generic scheduling algorithms will be reviewed for the genesis of the proposed Reconfigurable Constant Multiplier (RCM) design methodology in Chapter 4.

There are many scheduling approaches proposed in synthesis systems to optimally allocate design resources. Generally, these approaches can be categorized into three main groups, independent scheduling/allocation schemes, interdependent scheduling/allocation schemes, and stepwise refinement approaches [PAU89]. In the first category, scheduling and functional unit allocation is independent of any other mobile units to be scheduled. “As soon as possible” (ASAP) [TSE86, GEB87, DAV81] and “as late as possible” (ALAP) [KUN85] are two simplest and most representative scheduling schemes in this category. In ASAP approach, for example, the operators are scheduled to the earliest possible control step, s_i as long as all their left and right predecessor operators have been scheduled at a control step, $s_j < s_i$. This scheduling effectively assigns the operators based on the breadth first search of a balanced DAG construct. In the second category, operation scheduling is done concurrently. The allocation and scheduling order are dependent on other mobile operators. Priorities and urgencies of units are evaluated and taken into consideration in the scheduling decision making. For example, one mobile unit that can be scheduled into more possible control steps is given a lower scheduling urgency, and its priority for scheduling immediately in the next control step is lower. Variation of list scheduling approach [GIR84] and the scheme used in MAHA system [PAR86] are two representative approaches in this group.

The third group is the stepwise refinement approaches. Force-directed scheduling algorithm [PAU89] is one of the well-known methods in this category. A preliminary allocation is performed before a detailed analysis based on a defined scheduling force is used to establish the schedule estimate. As summarized in [PAU89], compared to other scheduling algorithms, force-directed scheduling is more versatile as it considers the global effects of the assignment of more types of operations. More importantly, rather than applying a brute-force assignment, it considers the effect of unscheduled units probabilistically in the decision making procedure. Scheduling decision is made according to the computed force for each individual unit, which includes the self force and the predecessor/successor forces. It is an iterative process. One unit is scheduled in each iteration so that information about the unassigned units can be updated before making the next scheduling decision. Force-directed algorithm creates and computes the Distribution Graphs to show the concurrency of each control step for each type of operation. Distribution Graph of control step, i , out of the m steps is defined as:

$$DG(i) = \sum_{Operationtype} prob(Operation, i) \quad (2.9)$$

By summing up the probabilities of occurrences of all operations of the same type in control step i , the concurrency of an operation in any control step becomes measurable. With the known concurrency distribution, the force of an operation can be computed. The force of an operation in a control step i is defined as [PAU89]:

$$Force(i) = DG(i) \times x(i) \quad (2.10)$$

where $x(i)$ is the change in that the probability of the operation after the assignment.

For an operation whose time frame spans from s_a to s_b , its total self force of assigning into step j is given by [PAU89]:

$$selfForce(j) = \sum_{i=s_a}^{s_b} [Force(i)] \quad (2.11)$$

By considering the predecessor and successor nodes if any, in the data flow graph (DFG), the total force of a node v to be scheduled into step j is then formulated as

$$TotalForce(v, j) = selfForce(j) + PredecessorForce + SuccessorForce \quad (2.12)$$

where *PredecessorForce* and *SuccessorForce* are respectively the forces of the predecessor and successor nodes of node v due to the scheduling of v into step j .

If the total force of scheduling node v into a step j appears to be positive, it indicates that there exists a positive force to push v away from this assigned control step j . The total force of scheduling v into other possible control steps can be similarly computed. Eventually, node v will be scheduled into the control step with the minimum total force. Using this approach, all operations can be assigned to their lowest force step to achieve an overall concurrency-efficient distribution. Therefore, force-directed scheduling approach is possibly one of the best methods to be considered in a scheduling problem which the operations have high mobility and strong inter-precedence dependency in the system. This fundamental idea of operation scheduling is explored for the development of a new design algorithm for RCM block in Chapter 4.

2.5 Summary

In this chapter, the background knowledge related to the realization of MCM and ReMB block in DSP applications, as well as a comprehensive literature survey on the existing methodologies are presented.

The fundamentals of digital FIR filters have been introduced to facilitate the explanation of filter design automation algorithms. Various types of number representations that have been widely adopted in the pivotal common subexpression elimination algorithms for the design of MCM and RCM blocks are discussed. Of which CSD and MSD representations are highlighted due to their guaranteed minimal number of nonzero digits, which is important consideration in view of the trade-off between the search complexity and solution quality in filter coefficient synthesis and optimization problems. Existing MCM and RCM block design methodologies and their main features have been discussed.

The topics presented in this literature review, will pave the way for the proposed design methodologies for the MCM and RCM blocks in the next two chapters.

Chapter 3

Information Theoretic Approach to FIR Filter Design

3.1 Introduction

Digital filtering at high data rate on battery-powered device can not be substantiated by general purpose signal processors without ancillary hardware accelerators. Therefore, the development of powerful computer-aided design tool for high-speed and low complexity digital filters is an alluring goal. For dedicated applications, the programmability of a multiplier is not mandatory and efficient fixed coefficient FIR filters can be realized in fully parallel architectures. The design turnaround time for application-specific FIR filters can be greatly reduced by algorithms that apply computational transformations to increase the performance with reduced implementation cost. As reviewed in Chapter 2, one widely adopted design methodologies [YU07, XU05b, XU07b, PAS99, PEI02, HAR96, MAH06, CHA06, BUL91, DEM95, DEM02a] for reducing the complexity of fixed coefficient FIR filters is to replace the expensive and power-hungry multipliers by simpler arithmetic adders and shifters and reuse the adders by common subexpression elimination (CSE). The number of parallel adders required can be reduced by exploiting the computational redundancy present in the multiple constant multiplications (MCM) [POT96] block.

Pattern matching [XU07b, PAS99, HAR96] is a technique that has historically been adopted in computer-aided design for high-level synthesis. It has become a popular approach to the design of VLSI efficient FIR filters in recent years. The gist of pattern based CSE lies in the detection of patterns and pattern correlation in a set of filter coefficients. Methods that reply on either exhaustive search, steepest descent heuristic or quasi-exhaustive search to eliminate common subexpressions or duplicated patterns of

partial sum from a given coefficient set normally have high computational complexity and are not suitable for large problem since the search for optimal common subexpressions is an NP-complete problem. On the other hand, a scrutiny of the heuristic CSE algorithms shows that some dilemma exists due to the relatively simple exploitation of the statistical distribution of different subexpressions. The subtle pattern correlation has not been formally and sufficiently explored and optimized even in the dependence graph approach. It is noted from information theory that the notion of entropy can be used to assess the amount of redundancy in a message and hence code the message a minimum set of patterns. In this chapter, the formulation of CSE for the design of low complexity FIR filters is revolutionized. The new formulation has led to several distinctive advantages. First, an intuitive data structure called the Multi-root Binary Partition Graph (MBPG) is evolved which has a direct correlation to the circuit topology. Under this data structure, common subexpressions and graph isomorphism are congruence. Isomorphism is an inherent attribute of minimal MBPG and is generated by construction. Second, information theoretic approach has been widely used in data compression, pattern recognition and classification [MIL95, HAL05, KHI57] but never has the elegance of information theory been so closely coupled with graph to tackle the complexity reduction of FIR filters. The concept of entropy and conditional entropy has been applied for the first time in this research to maximize common subexpression sharing by set partitioning. The synthesis of a minimal MBPG provides a meaningful insight into judicious resource sharing in MCM problem based on the rigor of proven probabilistic measures. Finally, as the widths of operands and the number of shifts are annotated on the vertices and edges of the resultant MBPG, heterogeneous adder complexity due to different widths and relative shifts of operands can be readily evaluated by graph traversal. Fine grain reduction of logic complexity and logic depth can be made by reordering the vertices without perturbing the size of MBPG.

This chapter is organized in six sections. In Section 3.2, the CSE problem will be recast as a set partitioning problem and MBPG is introduced. Section 3.3 presents the proposed information theoretic approach to the minimization of MBPG. A strategy to further reduce the implementation cost and critical path delay in terms of full adders is proposed

in Section 3.4. Section 3.5 presents the experimental results on commonly cited benchmark filters and practical channelizers of wideband radio receivers. A summary of this chapter is provided in Section 3.6.

3.2 Multi-root Binary Partition Graph.

A N -tap FIR digital filter is a linear time invariant system governed by a linear convolution of a discrete-time process variable $x(t)$ and a set of N finite-valued coefficients:

$$y(t) = \sum_{i=0}^{N-1} h_i x(t-i) \quad (3.1)$$

where h_i is the i -th coefficient, and $x(t)$ and $y(t)$ are the input and output data sampled at time t . (3.1) lends itself nicely into a fast transposed direct form realization whereby the critical path is limited to only a single multiply-and-add operation irrespective of the filter length, N . By input and output normalization/denormalization, the coefficients can be scaled to a set of constant integers for fixed point implementation.

Fig. 3.1(a) shows the architecture of a 3-tap transposed direct form FIR filter of a scaled integer coefficient set, $H = \{1132, 815, 556\}$. The multiplier block can be deemed as a single-input, multiple-output combinational function. If the multiplier block is rotated by 180° , and remap each constant multiplier into a network of hard-wired shifters and adders, a forest of binary trees can be derived where the roots are the coefficients, the intermediate nodes are adders of partial sums and the leaves are fundamental literals that define the number system for the coefficient representation. Since two's complement adders are generally available in high-level synthesis of digital system, it is natural to assume a minimal signed digit representation of the coefficients to reduce the number of dual polarity arithmetic operators of the tree. Fig. 3.1(b) shows the collapsing of the canonical signed digit (CSD) coefficients into a binary tree. By definition of a tree, there is one and only one path from the root to every intermediate vertex. The operators can be further reduced by merging isomorphic vertices, which leads to a directed-acyclic graph (DAG) shown in Fig. 3.2. This data structure is the backbone of the proposed

information-theoretic approach. It is called the Multi-root Binary Partition Graph (MBPG).

Fig. 3.1. (a) Transposed direct form FIR filter. (b) Adder tree decomposition

Fig. 3.2. (a) Multi-root Binary Partition Graph (b) Parent and children vertex

A MBPG is a DAG, $G = \{V, E\}$ where V represents a set of resources (typically two's complement adders) and E is a set of interconnects. Two types of vertices are defined in V . A non-terminal vertex v has as attributes two children vertices, $l(v)$ and $r(v)$ and an

output value, $c(v)$. The terminal vertices are vertices with no children and they represent the set of basic literals of the number system used to represent the coefficients. An edge, $e \in E$ is a connection from a vertex v_1 to a vertex v_2 . It has a value, e_{v_1} that indicates the amount of shift to be applied to $c(v_1)$ before feeding to v_2 .

Every vertex except the roots has at least one parent. The roots of G have no parent and their output values are equal to the coefficients, h_i . The children vertices are related to the parent vertex, v as follows:

$$c(v) = 2^{e_{l(v)}} c(l(v)) + 2^{e_{r(v)}} c(r(v)) \quad (3.2)$$

where $e_{l(v)}$ and $e_{r(v)}$ are the edge values emanated from v and directed to $l(v)$ and $r(v)$, respectively.

In radix-2 signed digit representation, three constant terminal vertices, 0, 1 and -1 are defined. The constant coefficients, h , are usually provided in a fractional form such that $0 \leq |h| < 1$. By factoring out the unit in the least significant position ($ulp = 2^{-B}$), a fixed point representation of B fractional digits can be obtained.

$$c(v) = \sum_{i=1}^{L_{c(v)}} s_i 2^{-p_i} = ulp \sum_{i=1}^{L_{c(v)}} s_i 2^{-p_i+B} = ulp \sum_{i=1}^{L_{c(v)}} s_i 2^{b_i-1} \quad (3.3)$$

where $s_i \in \{-1, 1\}$ is the sign of the i -th nonzero digit, p_i is the position of the i th nonzero digit after the decimal point and $L_{c(v)}$ is the number of nonzero digits of $c(v)$. Since fractional constant can be obtained by a shift operation at the output, $c(v)$ of any nonterminal vertex, $v \in V$ can now be succinctly represented by a set of signed digits, $\{s_i b_i\}$, where $b_i = B - p_i + 1$, signifying the signs and positions of the scaled signed-power-of-two (SPT) terms.

Using this simplified set representation of $c(v)$, from (3.3), the following set partitioning relations are observed when the coefficient set represented by the roots in MBPG is parsed.

$$\begin{aligned} c(l(v)) + e_{l(v)}, c(r(v)) + e_{r(v)} &\subset c(v), \\ \{c(l(v)) + e_{l(v)}\} \cup \{c(r(v)) + e_{r(v)}\} &= c(v) \\ \{c(l(v)) + e_{l(v)}\} \cap \{c(r(v)) + e_{r(v)}\} &= \emptyset. \end{aligned} \quad (3.4)$$

By recursively partitioning the output value of each vertex into two disjoint subsets of SPT terms, a complete binary tree can be formed and it can be reduced to a MBPG by sharing isomorphic vertices. Two non-terminal vertices, u and v are said to be isomorphic if $c(v) = c(u)$. If we define the number of occurrence of the same vertex as indegree, isomorphic vertices have indegree > 1 and they represent common subexpressions. Therefore, the common subexpression elimination problem in the design of multiplierless digital filter can be recast as a problem of synthesizing a minimal vertex set MBPG by set partitioning. Common subexpressions are generated by construction since isomorphism is an inherent graph theoretic property of MBPG. Unlike other graph dependence approaches and common subexpression elimination methods, the adder topologies are explicitly implied by MBPG and the bit-level information of each operand is readily accessible, making it amenable to fine-grain optimization at operator level and/or further tradeoff in logic operator and logic depth by computation reordering.

3.3 Information Theoretic Approach to MBPG Minimization

The basis of common subexpression elimination in the context of MBPG is the creation of isomorphism. This section presents a new paradigm of graph minimization method substantiated by information theory [HAL05].

If the communication between two vertices of an MBPG is considered as an information parsing process, then the messages being parsed are symbols in the set $\{s_i b_i\}$ of the source vertex, v_i . Since an event with absolute certainty to occur conveys no information, the information content of a source is connected with the reciprocal of expectancy. To maximize the number of occurrences of an event of interest, it is important to quantify the information content of a source in message parsing.

Definition 3.1: Let X be a memoryless source of n elementary messages, X_i , $i = 1, 2, \dots, n$, each with a probability of occurrence of $p(X_i)$ in an event and $\sum_{i=1}^n p(X_i) = 1$. The entropy $H(X)$ is defined as [KHI57]:

$$H(X) = -\sum_{i=1}^n p(X_i) \log_2 p(X_i) \quad (3.5)$$

where $p(X_i) \log_2 p(X_i)$ is taken to be 0 if $p(X_i) = 0$.

$H(X) = 0$ if $p(X_i) = 1$ for only one event, X_i , and $p(X_j) = 0$ for all other events, $X_j \neq X_i$. This is an extreme situation where the outcome of an event can be predicted with complete certainty. Maximum entropy occurs if each event X_i is equally likely, i.e., $p(X_i) = 1/n \forall i = 1, 2, \dots, n$. Entropy is a measure of the amount of uncertainty about X that is resolved when X is observed. In this case, an elementary message is a symbol or a set of symbols associated with a vertex and the event of interest is the formation of isomorphic subgraphs in MBPG. The entropy of a set of coefficients is then related to the minimal average number of pairs of symbols needed to represent it and hence the maximal amount of adder redundancy. A memoryless source implies that each message selected is independent of the previous message. The following definition is useful for evaluating the information content for dependent message parsing.

Definition 3.2: Let X and Y be two sources with n elementary and m elementary messages respectively, X_i and Y_j . Their probability distributions are given by $p(X_i)$ and $p(Y_j)$, respectively for $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m$. X_i and Y_j may be dependent. $p(Y_j | X_i)$ is the conditional probability of the occurrence of Y_j in an event provided that X_i has occurred. The conditional entropy $H(Y | X_j)$ of source Y , provided that X_j has occurred in X , is given by [KHI57]:

$$H(Y | X_i) = -\sum_{j=1}^m p(Y_j | X_i) \log_2 p(Y_j | X_i) \quad (3.6)$$

The probability of existence of isomorphic subgraphs in MBPG is determined from the signed digits of the subexpressions. Two patterns of signed expressions can be composed from any two nonzero digits, $s_i b_i, s_j b_j \in c(v)$. A positive subexpression is one that has $d = |b_i - b_j| - 1$ zeros between two non-zero digits, b_i and b_j , and $s_i \times s_j$ is positive. A negative subexpression is similarly defined except that $s_i \times s_j$ is negative. The PT array of [PEI02], which will be introduced in *Definition 3.3*, is used to keep track of the number of occurrences of the positive and negative subexpressions for the estimation of probability

and conditional probability for (3.5) and (3.6).

Definition 3.3: A *PT* array is a $2 \times (B - 2)$ dimensional array with the entry in the upper (lower) row and the j -th column represent the number of occurrences of positive (negative) subexpressions with j zeros, where $j = 1, 2, \dots, B - 2$ and B is the wordlength of the maximum coefficient in the given coefficient set.

Example 3.1: The CSD coefficients of $H = \{1132, 815, 556\}$ and its *PT* array can be:

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & 0 \end{bmatrix} PT = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 3 & 1 & 1 & 2 & 3 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$PT[s][d]$ is the number of occurrences of positive subexpressions with d zeros if $s = 1$ and negative subexpressions if $s = -1$. For instance, the bottom row of the fifth column of *PT*, denoted by $PT[-1][5] = 3$ since there are three occurrences of the negative subexpressions, $100000\bar{1}$ and $\bar{1}000001$. It should be noted that during the construction of MBPG, the *PT* array is dynamically updated to ensure that the entropy and conditional entropy are calculated based on the latest statistics as the messages carried by the source are parsed.

The following probability computations from the *PT* array are used to evaluate the entropy and conditional entropy in Definitions 3.1 and 3.2.

Let $s_i b_i, s_j b_j \in c(v)$, the probability that $s_i b_i$ and $s_j b_j$ form a subexpression is given by:

$$p(s_i b_i, s_j b_j) = \frac{PT[s_i s_j][|b_i - b_j| - 1]}{\sum_{h \neq i} PT[s_i s_h][|b_i - b_h| - 1]} \quad (3.7)$$

where $s_h b_h$ is other signed digit in $c(v)$ rather than $s_i b_i$.

Example 3.2: The three coefficients in H , named as v_1, v_2, v_3 , in Example 3.1 can be taken as an example to illustrate (3.7). Trailing zero in the CSD number are shifted out and the

edge values are adjusted accordingly. In $\{s_i b_i\}$ notation, $c(v_1) = \{9, 6, -3, -1\}$, $c(v_2) = \{11, -9, 7, -5, -1\}$ and $c(v_3) = \{8, 5, -3, -1\}$ with $e_{v1} = e_{v3} = 2$ and $e_{v2} = 0$.

$$\begin{aligned} & \sum_{j \neq 1} PT[s_i s_j][|b_i - b_j| - 1] \\ &= PT[1][|9-6|-1] + PT[-1][|9-3|-1] + PT[-1][|9-1|-1] \\ &= 2 + 3 + 1 = 6. \end{aligned}$$

From (3.7), for $s_1 b_1 = 9$,

$$p(\{9, 6\}) = PT[1][|9-6|-1]/6 = 0.333,$$

$$p(\{9, -3\}) = PT[-1][|9-3|-1]/6 = 0.5,$$

$$p(\{9, -1\}) = PT[-1][|9-1|-1]/6 = 0.167.$$

With these probabilities calculated, the entropy of 9, $H(9)$, can be computed according to Definition 3.1 as follows:

$H(9) = -\sum_{j \neq 1} p(s_1 b_1, s_j b_j) \log_2 p(s_1 b_1, s_j b_j) = 1.4595$, as $s_1 b_1$ can form subexpression with any other signed digit $s_j b_j$ except itself, i.e., $j \neq 1$. The entropies of the other three signed digits can be similarly computed as: $H(6) = 1.5219$, $H(-3) = 1.4595$ and $H(-1) = 1.5219$. Therefore, the average entropy of v_1 is $H(v_1) = 1.4907$. By repeating the above calculation for v_2 and v_3 , the entropies of v_2 and v_3 can be obtained as $H(v_2) = 1.9153$ and $H(v_3) = 1.5049$.

To construct a minimal MBPG, the roots are first generated in G . The value of each root is the set of {sign, position} of the SPT terms of the CSD coefficient. If the value of a root is equal to 1, -1 or 0, it is assigned to a constant terminal vertex 1, -1 or 0 accordingly. The non-terminal vertices generated are inserted into a list, Q . For each nonterminal vertex, $v \in Q$, two information theoretic decisions will be made. First, when $c(l(v)) = c(r(v)) = \emptyset$, a pair of values, $s_i b_i$ and $s_j b_j \in c(v)$ will be selected based on the entropy measure to give birth to $l(v)$ and $r(v)$ in G . Then, the remaining signed digits of $c(v)$ are successively allocated to either $c(l(v))$ or $c(r(v))$ depending on their conditional entropies. The following propositions are suggested for these operations.

Proposition 3.1: The nonterminal vertex, u with the least entropy(u), where entropy(u) is calculated by the sum of the entropy of each signed digit of u . $u = \arg\left(\min_{v \in G} H(c(v))\right)$ will be decomposed first since the vertex of minimum entropy indicates the maximum likelihood that it subsumes isomorphic subgraphs. Since the PT array changed dynamically in the process of decomposition, it is preferable to generate the most potential isomorphic subgraph as early as possible to prevent their formation from being annihilated.

Example 3.3: By taking the three coefficients in H , v_1 , v_2 and v_3 , as an example, $H(v_1) = 1.4907$, $H(v_2) = 1.9153$ and $H(v_3) = 1.5049$. v_1 is with the least entropy value and hence will be decomposed first.

Proposition 3.2: For any $s_i b_i \in c(v)$, the lower the entropy $H(s_i b_i)$, the higher the confidence that $s_i b_i$ is contained in an isomorphic subgraph subsumed by v . Therefore, the two signed digits that give the least entropy shall be split into $c(l(v))$ and $c(r(v))$, provided that the probability of these two signed digits coexist in the same isomorphic subgraph in G is sufficiently low.

Example 3.4: The signed digits of $c(v_1) = \{9, 6, -3, -1\}$ from Example 3.1 are first sorted in order of increasing entropy into $D = \{9, -3, 6, -1\}$. If the two digits have the same entropy, such as 9 and -3 , they have the same certainty of decomposition decision. The same order of evaluating the entropy values of these digits is assumed. According to Proposition 3.2, $C_l = \{9\}$ and it is necessary to check the probability $p(\{9, -3\})$ before assigning -3 to C_r . Since $p(\{9, -3\}) = 0.5 \leq 0.5$, $C_r = \{-3\}$ and $D(v_1) = \{6, -1\}$. Signed digits that have split into two different children can no longer form a subexpression, but the digits remaining in v_1 still have a chance to form subexpression with 9 and -3 if allocated to the same children vertex later. Hence PT array is updated to reflect the change in the statistic of positive and negative subexpressions. The updated PT array is given by:

$$PT = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 3 & 1 & 1 & 2 & 2 & 1 & 1 & 0 & 1 \end{bmatrix}$$

After the left and right children vertices are created, the conditional probability and conditional entropy for the signed digits remaining in the parent vertex need to be defined before the allocation. Let $m = |c(l(v))|$ and $n = |c(r(v))|$ which are the number of signed digits of left child and right child. The conditional probability that a signed digit, $s_j b_j \in c(l(v))$ forms a subexpression with $s_i b_i \in c(v)$ given that $s_i b_i$ is assigned to $l(v)$ can be computed by:

$$p(s_j b_j | s_i b_i \rightarrow c(l(v))) = \frac{PT[s_i s_j][|b_i - b_j| - 1]}{\sum_{k=1}^m PT[s_i s_k][|b_i - b_k| - 1] + \sum_{l=1}^n PT[s_i s_l][|b_i - b_l| - 1]} \quad (3.8)$$

where $s_k b_k \in c(l(v))$ and $s_l b_l \in c(r(v))$. The conditional probability of a signed digit in the right child of v forming a common subexpression with a designated signed digit, $s_i b_i \in c(v)$ if $s_i b_i$ is assigned to $r(v)$ can be similarly defined.

Proposition 3.3: Let $s_i b_i \in c(v)$, then $s_i b_i$ will be assigned to $l(v)$ if $H(c(l(v)) | s_i b_i \rightarrow c(l(v))) < H(c(r(v)) | s_i b_i \rightarrow c(r(v)))$ and vice versa. This is because the lower conditional entropy between two assignments of $s_i b_i$ implies a greater certainty that the assignment will lead to isomorphic subgraphs whose SPT terms subsume the signed digit $s_i b_i$.

Example 3.5: If 6 of v_1 in the above example is assigned to C_l , from (3.8), $p(9 | 6 \rightarrow C_l) =$

$$\frac{PT[1][|6 - 9| - 1]}{PT[1][|6 - 9| - 1] + PT[-1][|6 - 3| - 1]} = \frac{2}{2+1} = 0.6667 \text{ and the conditional entropy,}$$

$H(9 | 6 \rightarrow C_l) = 0.39$. If 6 is assigned to C_r , it has $p(9 | 6 \rightarrow C_r) = 0.3333$. $H(9 | 6 \rightarrow C_r) = 0.5283$. According to *Proposition 3.3*, 6 is assigned to C_l . Now, $C_l = \{9, 6\}$, $C_r = \{-3\}$ and $D(v_1) = \{-1\}$. The *PT* array is updated to invalidate $\{6, -3\}$.

$$PT = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 2 & 2 & 1 & 1 & 0 & 1 \end{bmatrix}$$

If the last signed digit -1 in D is assigned to C_l ,

$$p(9 | -1 \rightarrow C_l) = \frac{PT[-1][7]}{PT[-1][7] + PT[-1][4] + PT[1][3]} = \frac{1}{1+2+2} = 0.2$$

$$p(6|-1 \rightarrow C_l) = \frac{PT[-1][4]}{PT[-1][7] + PT[-1][4] + PT[-1][1]} = \frac{2}{1+2+2} = 0.4$$

$H(C_l|-1 \rightarrow C_l) = -0.4\log_2 0.4 - 0.2\log_2 0.2 = 0.9931$. Since $H(C_r|-1 \rightarrow C_r) = 0.5288 < H(C_l|-1 \rightarrow C_l)$, then $C_l = \{9, 6\}$, $C_r = \{-3, -1\}$. PT is updated to eliminate the statistics due to the subexpressions, $\{9, -1\}$ and $\{6, -1\}$.

$$PT = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 1 & 2 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Since $D(v_1) = \phi$, v_1 has been completely decomposed. To eliminate the trailing zeros, shift amounts $e_l(v_1) = 8$, $e_r(v_1) = 3$, giving $c(l(v_1)) = \{4, 1\}$ corresponding to subexpression 1001 and $c(r(v_1)) = \{-3, -1\}$ corresponding to the subexpression $\bar{1}0\bar{1} \equiv -(101)$. Neither $c(l(v_1))$ nor $c(r(v_1))$ exists in G , they are inserted in to G as new vertices, v_4 and v_5 , respectively. v_1 will be deleted from Q and v_4 and v_5 will be inserted for further decomposition. As there are only 2 nonzero digits in each of v_4 and v_5 , they will be decomposed into terminal vertices 1 and $\bar{1}$. To this end, $G = \{1, -1, v_1, v_2, v_3, v_4, v_5\}$.

Fig. 3.3 outlines the algorithmic flow for the construction of a minimal MBPG. Propositions 3.1 to 3.3 are exercised in the core decomposition operation. Its pseudo code, **decompose(v)** is detailed in Fig. 3.4.

Fig. 3.3. Flow chart of the proposed information theoretic method.

The graph G is initialized with three terminal vertices, 1, -1 and 0. A list Q is used to store the newly generated vertices and it is initialized with a list of roots. The function **decompose**(v) is called to decompose the vertex, v in Q that has the minimum entropy until Q is empty. In Fig. 3.4, the function **entropy**(v) computes the entropy from the PT array. The two signed digits, s_1b_1 and s_2b_2 with the least entropy and have low certainty to form subexpression themselves are split into two children lists of SPT terms, C_l and C_r . The function **update**(PT, L_1, L_2) updates the PT array by decrementing the occurrences of subexpressions that can be formed by one digit from L_1 and the other from L_2 . The function **conditional_entropy**(C, sb) returns the conditional entropy defined in (3.6) with $Y \equiv C$, $X_i \equiv sb$ and $m \equiv |C|$. When all signed digits of a decomposed vertex, v have been exhausted, the left and right edge values of v are obtained from the least magnitude signed digits of C_l and C_r , respectively. The positions of the SPT terms in C_l and C_r are shifted accordingly before they are assigned to $c(l(v))$ and $c(r(v))$. The function **right_shift**(C, e) subtracts e from the magnitude of every signed digit in C . New vertices, $l(v)$ and $r(v)$ with values C_l and C_r are generated if they have yet existed. The function **exist**(G, C) verifies if a vertex with $c(v)$ equals the set C has already been generated in G to avoid duplication of vertices. If found, the vertex is returned. The function **new_node**(C) creates a new vertex, v with its output value, $c(v)$ equals the set C of SPT terms. Finally, the left and right children rooted at v are inserted into G by the function **insert**($G, v, l(v), r(v)$).

```

decompose( $v$ ) {
  for (each  $s_i b_i$  in  $c(v)$ ) calculate entropy( $s_i b_i$ );
   $D$  = signed digits of  $c(v)$  sorted in ascending order of entropy;
   $s_1 b_1$  = first entry of  $D$ ;  $s_2 b_2$  = second entry of  $D$ ;
  while (entropy( $\{s_1 b_1, s_2 b_2\}$ ) > 0.5)  $s_2 b_2$  = next entry of  $D$ ;
  transfer  $s_1 b_1$  from  $D$  to  $C_l$ ;
  transfer  $s_2 b_2$  from  $D$  to  $C_r$ ;
  update( $PT, s_1 b_1, s_2 b_2$ );
  while ( $D$  is not empty) {
     $sb$  = first entry of  $D$ ;  $H_l$  = conditional_entropy( $C_l, sb$ );
     $H_r$  = conditional_entropy( $C_r, sb$ );
    if ( $H_l < H_r$ ) transfer  $sb$  from  $D$  to  $C_l$ ; update( $PT, C_r, sb$ );
    else transfer  $sb$  from  $D$  to  $C_r$ ; update( $PT, C_l, sb$ );}
   $e(v)$  = minimum magnitude signed digit in  $C_l$ ;
   $C_l$  = right_shift( $C_l, e(v)$ );
  if ( $u = \text{exist}(G, C_l)$ )  $l(v) = u$ ;
  else {  $l(v) = \text{new\_node}(C_l)$ ; insert( $Q, l(v)$ ); }
}

```

```

 $e_r(v)$  = minimum magnitude signed digit in  $C_r$ ;
 $C_r = \text{right\_shift}(C_r, e_r(v))$ ;
if ( $u = \text{exist}(G, C_r)$ )  $\text{right}(v) = u$ ;
else {  $r(v) = \text{new\_node}(C_r)$ ;  $\text{insert}(Q, r(v))$ ; }
 $\text{insert}(G, v, l(v), r(v))$ ;
}

```

Fig. 3.4. Algorithm for vertex decomposition

The process will be repeated and Fig. 3.5 shows the trace of computation towards the final MBPG. The final reduced MBPG of Fig. 3.5 has the same topology of the architecture shown in Fig. 3.2. Since $|V| = 7$ (excludes terminal vertices), seven adders are needed and the critical path has three adder delay.

Fig. 3.5. Generation of minimal MBPG for $H = \{1132, 815, 556\}$

3.4 Shift Inclusive Computation Reordering for Adder Complexity Reduction

Many CSE algorithms [XU07b, PAS99, PEI02, HAR96] and Graph Dependence algorithms [DEM95, BUL91, KAN01, XU05a] assume uniform adder cost and adder delay for all adders employed in their solution. In practice, although the hardwired shifters contribute zero cost to the logic complexity and logic depth of the multiplier block, they cause varying operand displacements to the adders. Therefore, a better metrology is needed to account for the area-time dependency of each discrete adder on operand lengths due to the relative positional shift of its operands [VIN05b]. Since the primary objective of most CSE algorithms is on complexity reduction, and RCA has been known to be the simplest and power efficient adder, it will be used consistently for two's complement addition throughout this chapter. Fig. 3.6 illustrates the full adder (FA) cost required to add/subtract two operands in the vertex, v of MBPG using two's complement ripple carry adder (RCA). Let $b_{msb}(v)$ and $b_{lsb}(v)$ denotes the magnitudes of the most and least significant signed digits of $c(v)$ respectively which are equal to 1. Assume that the wordlength of the input, x to the filter is n bits, then the length of the input operand to the adder is $b_{msb}(child(v)) + n + e_{child(v)} - 1$ with $e_{child(v)}$ trailing zeros, where $child(v)$ is either $l(v)$ or $r(v)$. In Fig. 3.6, every solid dot represents an unknown bit and s represents a sign or sign extended bit. Carry free addition with a string of zeros can be hardwired. To add $l(v)$ and $r(v)$, the number of FAs required is given by $n + msb_{max} - lsb_{max}$ where $msb_{max} = \max\{b_{msb}(l(v)) + e_{l(v)}, b_{msb}(r(v)) + e_{r(v)}\}$ and $lsb_{max} = \max\{b_{lsb}(l(v)) + e_{l(v)}, b_{lsb}(r(v)) + e_{r(v)}\}$. The FA at the least significant bit of RCA can be reduced to a half adder (HA) since there is no carry from the $\max\{e_{l(v)}, e_{r(v)}\}$ -th position. To subtract $r(v)$ from $l(v)$, the same number of FAs are needed if $e_{l(v)} \leq e_{r(v)}$, since the two's complement of a string of e_{max} zeros results in a string of zeros with a carry-out of one which will be fed into the carry-in of the RCA. If $e_{l(v)} > e_{r(v)}$, then the operands can be swapped since $r(v) - l(v) = -(l(v) - r(v))$ with the sign of the output toggled to preserve the same number of FAs; otherwise, $e_{l(v)} - e_{r(v)}$ additional FAs will be needed. Output sign change is also necessitated if both operands are negative since $-l(v) - r(v) = -(l(v) + r(v))$. The toggling of output sign can

be propagated to the next adder in MBPG, making the addition a subtraction and vice versa. In general, the width, W of the RCA for vertex v is given by:

$$W(v) = n + msb_{max} - lsb_{max} \quad (3.9)$$

Fig. 3.6. Full adder cost for addition/subtraction using RCA

The type of adder and its bit width also affects the critical path delay of the multiplier block. It should be noted that the real critical path of an FIR filter predicted by this shift inclusive delay metric may not be the path with the maximum number of discrete adders. Hence, besides the topology of adders in the multiplier block, the internal architecture of each adder needs to be considered to obtain a more accurate estimate of the critical path delay. The worse-case delay of the RCA is equal to $W \times t_{FA}$ where W is the width of the RCA and t_{FA} is the delay of a FA. The path delay of each tap can be computed by a simple depth first traversal of the MBPG. The pseudo code to compute the critical path delay is shown in Fig. 3.7 assuming that $W(v)$ for each vertex, $v \in V$ has already been computed by (3.9) and stored in the vertex.

```

Critical_path_delay(G) {
    max_delay = 0;
    for each (root, u of G) {
        delay = path_delay(v);
        if (delay > max_delay) max_delay = delay;
    }
    return max_delay;
}

path_delay(v) {
    if (v = terminal vertex) return 0;
    return W(v) + max{path_delay(l(v)), path_delay(r(v))};
}

```

Fig. 3.7. Critical path delay computation

The disjunctive decomposition algorithm of Fig. 3.4 aims at synthesizing a multiplier block with reduced number of discrete adders. Further reduction of the FA cost and FA delay can be achieved by reducing the operand width of each non-terminal vertex without jeopardizing the cardinality of the vertex set and the depth of MBPG. This is obtained by computation reordering after the MBPG has succeeded in finding good common subexpressions to reduce the number of discrete adders. The steps involved are given as follows.

Step 1: Select a root of MBPG, make a graph traversal to extract vertices with indegree ≥ 2 . The shift-inclusive input set, $\{c(v(child)) + e_{v(child)}\}$ (i.e., the input operand to the adder) of these vertices are inserted into a list U . The remaining non-zero digits, $s_i b_i$ of the root that can not be composed directly from the common subexpressions in U are inserted as isolated nonzero digits into the list Q .

Step 2: For each pair of signed digit sets, $u_i, u_j \in U$, calculate $msb_{max} = \max\{b_{msb}(u_i), b_{msb}(u_j)\}$ and $lsb_{max} = \max\{b_{lsb}(u_i), b_{lsb}(u_j)\}$. For an isolated non zero digit, $s_i b_i$, $b_{msb} = b_{lsb} = b_i$. The pair of vertices, or subexpressions, $u_i, u_j \in U$ with the minimum $\Delta b = msb_{max} - lsb_{max}$ among all possible pairs from U will be selected to compose a new subgraph rooted at u_k with $u_i, u_j \in child(u_k)$. If there is a tie, the pairs that generates a subgraph with the minimum adder depth (which can be determined by the binary logarithmic of the hamming weight of $c(u_k)$) will be selected. u_i and u_j are replaced by u_k in U .

Step 3: Repeat Step 2 until there are only two vertices in U . These two vertices will compose the root. Repeat Step 1 with the next root until all roots have been recomposed. Since the number of FAs used is determined by the operand width, Step 2 ensures that the average width of the operands generated by the non-terminal vertices composing the coefficients in MPBG has been reduced by reordering the additions. Since isomorphic subgraphs are kept intact, changes, if any, affect only the ordering of some operands and/or their shifts but not the total number of operators of MBPG. In computation reordering, priority is given to low hamming weight subexpressions so that the adder tree so formed is more balanced and the critical path may be simultaneously reduced.

Example 3.6: Consider the MBPG in Fig. 3.5. The final PT array is given by:

$$PT = \begin{bmatrix} 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

It identifies the weight-two common subexpressions for this coefficient set. The three common subexpressions are 101 (v_5), 1001 (v_4) and 100000 $\bar{1}$ (v_6). The shift-inclusive computation reordering is applied to Fig. 3.5 and the FA cost is calculated as follows. For coefficient h_1 , v_1 is a sum of two common subexpressions, so no further FA cost optimization is performed. From (3.9), $W(v_4) = n + \max(4, 1) - \max(4, 1) = n$. $W(v_5) = n + \max(3, 1) - \max(3, 1) = n$ and $W(v_1) = n + \max(9, 3) - \max(6, 1) = n + 3$. Therefore, the adder cost of h_1 is $3n + 3$ FAs. $path_delay(v_1) = W(v_1) + \max\{W(v_4) + 0, W(v_5) + 0\} = (2n + 3) t_{FA}$.

For coefficient h_2 , v_2 is composed of one isolated non-zero digit $\bar{1}$ and two different shifted versions of a weight-two common subexpression 100000 $\bar{1}$ (v_6). Shift inclusive FA cost optimization is performed as follows. In Step 1, $U = \{u_1, u_2, u_3\}$ where $u_1 = \{7, -1\} = 100000\bar{1}$, $u_2 = \{11, -5\} = 100000\bar{1}0000$ and $u_3 = \{-9\} = \bar{1}00000000$. In Step 2, $\Delta b_{12} = \max(7, 11) - \max(1, 5) = 6$, $\Delta b_{13} = \max(7, 9) - \max(1, 9) = 0$, $\Delta b_{23} = \max(11, 9) - \max(5, 9) = 2$. Therefore, $u_4 = \{-9, 7, -1\}$ is composed from u_1 and u_3 . Now, $U = \{u_2, u_4\}$ and the root is generated from u_2 and u_4 . It happens that $u_4 \equiv v_7$ and $u_2 \equiv v_6$, no change is made to

the original composition of v_2 . The FA cost is determined by (3.9). $W(v_6) = n + \max(7, 1) - \max(7, 1) = n$. $W(v_7) = n + \max(7, 9) - \max(1, 9) = n$ and $W(v_2) = n + \max(11, 9) - \max(5, 1) = n + 6$. Therefore, the adder cost of h_2 is $3n + 6$ FAs. $path_delay(v_2) = W(v_2) + \max\{W(v_6)+0, W(v_7) + \max\{W(v_6)+0, 0\}\} = W(v_2) + W(v_7) + W(v_6) = (3n + 6) t_{FA}$. Fig. 3.8 shows the three different compositions of MBPG by different computation orderings by selecting Δb_{12} , Δb_{13} and Δb_{23} in Step 2. The FA cost for each vertex is annotated.

For coefficient h_3 , v_3 is a sum of two common subexpressions and no reordering is needed. Taking into account that v_3 is the sum of v_4 and v_5 which are shared with h_1 , only $W(v_3) = n + \max(8, 3) - \max(5, 1) = n + 3$ additional FAs are needed. $path_delay(v_3) = W(v_3) + \max\{W(v_4) + 0, W(v_5) + 0\} = (2n + 3) t_{FA}$.

The 3-tap filter as a whole requires $7n + 12$ FAs and has a critical path delay of $(3n + 6) t_{FA}$.

(a) Adder cost = $7n+14$, FAs, delay = $3n+8$

(b) Adder cost = $7n+12$, FAs, delay = $3n+6$

(c) Adder cost = $7n+14$, FAs, delay = $3n+8$

Fig. 3.8. MBPG after computation reordering (a) $\Delta b_{12} = 6$ (b) $\Delta b_{13} = 0$, and (c) $\Delta b_{23} = 2$

3.5 Experimental Results

In this section, numerical results are presented to demonstrate the effectiveness of the proposed MBPG algorithm in reducing the implementation complexity of FIR filters. The adder cost and adder delay of solutions by proposed algorithm are compared with several other CSE algorithms [PAS99, PEI02, HAR96, MAH06, DEM95] based on the number of FAs and number of t_{FA} in the critical path of the test filters. The FA cost and FA delay for the solutions of all algorithms are worked out from their shift-and-add topologies using the same computation model of Fig. 3.6. For FA cost evaluation, wordlength of the input signal is needed. An input wordlength of 12 bits for all the experiments are used since this resolution of ADC is very common. The targeted implementation structure is the multiplier block of transposed direct form filter realized with two's complement RCAs. The accumulators in the tap-delay line are not considered because the numbers are equal for all algorithms. Symmetrical filters are realized with folded implementation [PAR99] and only half the number of taps is considered. The performances are evaluated in two different sets of filters.

The first set consists of ten commonly referenced benchmark filters, Filters A to J. Their design parameters are listed in Table 3.1. N is the total number of filter taps, L is the wordlength of the filter coefficients, f_p and f_s denote the pass band and stop band edge frequencies that are normalized to 1, respectively. r_p and r_s denote the pass band ripples and stop band attenuation in dB, respectively. The filter type is specified in the column labeled 'Type', where LP, HP and BP represent low-pass, high-pass and band-pass filters, respectively. The filter coefficients are generated using Parks-McClellan (PM), least square (LS) and weighted least square (WLS) algorithms. "-" indicates unknown coefficient synthesis algorithm. The filter coefficient sets can be excerpted directly from the references listed in the first column, regardless of synthesis algorithms.

Tables 3.2 and 3.3 show the logic complexity and logic depth comparisons of proposed MBPG algorithm with the baseline CSD and Binary implementations, and several other algorithms such as NRSCSE [PEI02], Pasko [PAS99], Hartley [HAR96], BHM [DEM95], C1 [DEM02a] and BCSE [MAH06]. As discussed in Chapter 2, CSD and BIN

refer to the baseline implementations where all filter coefficients are recoded in CSD and Binary format with no further optimization [WAN05] other than the adder tree for each coefficient is balanced to minimize the critical path delay.

Table 3.1 Test filter specifications

Filters	Type	N	L	f_p	f_s	$r_p(\text{dB})$	$r_s(\text{dB})$
A [SAM89]	LP, LS	25	9	0.15	0.25	0.005	46
B [LIM83]	LP, WLS	36	9	0.1	0.14	0.36	30
C [LIM83]	LP, WLS	37	9	0.1	0.14	0.34	30
D [LAS92]	LP, -	43	16	0.2	0.3	0.004	30
E [SAM89]	LP, PM	60	16	0.021	0.07	0.2	60.5
F [LIM83]	LP, WLS	63	16	0.1	0.14	0.48	60
G [VIN07]	LP, PM	120	16	0.2	0.22	0.1	60
H [PAS99]	HP, LS	121	16	0.783	0.74	0.081	80
I [VIN07]	LP, PM	200	12	0.2	0.22	0.1	60
J [RAI94]	BP, -	281	16	0.01-0.99	0-0.002;0.998-1	0.5	60

Table 3.2 FA cost comparison

Filter	CSD	BIN	BHM	Pasko	NRSCSE	Hartley	C1	BCSE	MBPG
A	105	161	78	75	75	75	72	94	75
B	60	100	64	60	60	60	60	64	60
C	60	134	63	60	60	60	60	84	60
D	934	1359	781	723	725	798	619	873	709
E	912	1429	482	563	555	712	546	684	548
F	706	1478	402	461	467	550	362	483	435
G	2031	2750	1021	1289	1290	1542	933	1266	1238
H	1984	2723	969	1170	1213	1549	884	1181	1147
I	682	935	414	455	398	446	375	413	397
J	2337	3059	1189	1355	1278	1650	1059	1284	1266

Table 3.3 FA critical path delay comparison

Filter	CSD	BIN	BHM	Pasko	NRSCSE	Hartley	C1	BCSE	MBPG
A	30	60	27	27	27	27	24	46	27
B	12	26	28	12	12	12	12	28	12
C	12	72	27	12	12	12	12	48	12
D	60	108	170	71	54	52	80	75	54
E	33	108	68	43	47	47	31	53	32
F	50	120	93	49	49	49	44	49	45
G	60	96	127	74	52	58	71	66	51
H	60	96	178	74	50	50	50	59	49
I	60	60	85	59	49	49	49	49	44
J	57	108	132	65	53	53	53	83	52

From the results of Tables 3.2 and 3.3, the proposed MBPG method performs well overall in terms of both logic complexity and logic depth. The average reductions in the FA delay of critical path over CSD, BIN, BHM, Pasko, NRSCSE, Hartley, C1 and BCSE are 12.73%, 56.27%, 58.49%, 23.99%, 9.35%, 7.59%, 5.68% and 33.63%, respectively. On average, the FA costs are reduced by 37.48%, 54.74%, 5.26%, 2.99%, 19.30% and 11.41% over CSD, BIN, Pasko, NRSCSE, Hartley and BCSE, respectively. However, the average FA cost of the proposed method is 3.05% and 12.71% worse than that of BHM and C1. This is because BHM and C1 algorithm search exhaustively from all possible primitive adder graph topologies for the partial sums formed in any single coefficient multiplication. It is therefore capable of finding certain redundancies not identifiable by proposed heuristic pattern-based approach. The premium for the low adder cost of BHM is its low throughput rate. The critical path delays of its solutions are dramatically longer than those of the proposed MBPG. C1 [DEM02a] is one of the best algorithms among the latest proposed enhancements to BHM for its logic depth reduction. It maximizes the use of cost-1 partial sums to reduce the logic depth and searches exhaustively among all existing graphs to select a solution with best area-time performance. This results in a very high computational complexity.

The overall area-time (AT) performance is measured by the product of FA cost and FA delay. The AT results for Filters D-J with more than 40 taps from Table 3.1 are normalized by the AT of BIN (i.e., BIN method has a normalized AT of 1) and plotted in

Fig. 3.9. For each filter, the normalized AT of the eight algorithms are displayed in the order listed in the legend. It is evident that MBPG and C1 have comparable AT complexities, which are much lower than those of all other algorithms.

Fig. 3.9. Normalized AT complexity comparison for Filters D to J. For each filter, the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM

The second set of filters involves the channelizers of wideband receiver operate in the intermediate frequency (IF). The performance of proposed MBPG method is evaluated and compared with other algorithms using two communication filter design examples with the same coefficient wordlength, $L = 12$ and varying filter length, N . They are designed to demonstrate the potential of proposed algorithm in realizing application-specific integrated filters that meet the stringent narrow transition band and high sampling frequency filtering requirements. FIR1 is a linear phase FIR filter (LPFIR) employed in the filter bank channelizer of Digital Advanced Mobile Phone Systems (D-AMPS) in [VIN05a]. The decimation is moved to the left of the band-pass filters using the noble identity and the sampling rate chosen is 34.02MHz. The coefficient sets are provided by the authors of [VIN05a]. The channel filters extract 30 kHz D-AMPS channels from the input signal after downsampling by a factor of 350. The pass band and stop band edges are 30 and 30.5 kHz respectively. The peak pass band ripple is chosen as

0.1 dB. The filter stop band specifications are chosen as in the D-AMPS standard [VIN05a]. According to [VIN05a], the lengths of the LPFIR filters are determined using the following equation from [PRO98].

$$N = \frac{-10 \log_{10} \partial_1 \partial_2 - 13}{14.6 \Delta f} + 1 \tag{3.10}$$

where ∂_1 and ∂_2 are the pass band and stop band ripples, respectively, and Δf is the normalized width of the transition band. The filter lengths, $N = 200, 460, 610, 940$ and 1180 corresponding to stop band attenuation of $-24\text{dB}, -48\text{dB}, -65\text{dB}, -85\text{dB}$ and -96dB , respectively are tested. FIR2 is the channel filter employed in receivers for the Personal Digital Cellular (PDC) standard. The sampling rate of the wide band signal is 25.6 MHz , which covers 1024 channels of 25.5 kHz spacing after down sampling by a factor of 350 . The pass band and stop band edges are 25 and 25.5 kHz , respectively. The peak pass band ripple is chosen as 0.1 dB . The filter stop band specifications are chosen as in the D-AMPS standard and the length of the filter, N is determined by (3.10). $N = 230, 450, 650$ and 800 are used to attain an attenuation of $-30\text{dB}, -60\text{dB}, -80\text{dB}$ and -90dB , respectively for this evaluation.

The FA cost of various algorithms for FIR1 and FIR2 are compared in Tables 3.4, and the FA delays are analyzed in Tables 3.5.

Table 3.4 FA cost comparison of FIR1 and FIR2

Filter	FIR1					FIR2			
N	200	460	610	940	1180	230	450	650	800
CSD	841	761	746	758	839	691	729	786	747
BIN	1036	929	950	948	1041	815	923	1003	916
BHM	500	466	476	476	523	420	462	500	450
Pasko	481	447	429	437	504	395	421	449	422
NRSCSE	460	433	430	430	475	408	430	458	431
Hartley	538	477	474	474	501	446	460	517	490
C1	446	421	421	423	468	402	426	454	421
BCSE	492	449	438	457	499	417	475	479	444
MBPG	454	423	420	420	465	398	424	452	430

Table 3.5 FA delay comparison of FIR1 and FIR2

Filter	FIR1					FIR2			
<i>N</i>	200	460	610	940	1180	230	450	650	800
CSD	30	30	30	30	30	48	48	48	48
BIN	60	60	60	60	60	60	60	60	60
BHM	80	80	84	84	84	63	84	84	63
Pasko	45	45	43	45	46	45	45	45	45
NRSCSE	44	44	44	44	44	48	48	48	48
Hartley	45	45	45	45	42	48	48	48	48
C1	30	30	30	30	30	46	46	46	45
BCSE	60	60	60	60	60	61	61	47	61
MBPG	30	30	30	30	30	42	42	42	42

For FIR1, the average reductions of FA cost by proposed method are respectively 44.66%, 55.49%, 10.61%, 4.94%, 2.04%, 11.38%, -0.14% and 6.51% over CSD, BIN, BHM, Pasko, NRSCSE, Hartley, C1 and BCSE. The solutions produced by MBPG are notably faster. The average reductions of the critical path delay are 0%, 50.00%, 63.57%, 33.00%, 31.82%, 32.38%, 0% and 50.00% over CSD, BIN, BHM, Pasko, NRSCSE, Hartley, C1 and BCSE respectively. For FIR2, in comparison with CSD, BIN, BHM, Pasko, NRSCSE, Hartley, C1 and BCSE, the average FA cost reductions are 42.29%, 53.31%, -1.01%, 6.88%, 1.35%, 10.85%, -0.06% and 6.02%, respectively, and the average critical path delay reductions are 12.50%, 30.00%, 41.67%, 6.67%, 12.50%, 12.50%, 8.19% and 26.02%, respectively. The rates of area reduction and speed improvement vary with the length of the filter. It is possible to have a lower FA cost for filters with a large number of taps compared to filters with relatively fewer taps when there is a substantial reduction of nonzero digits in the CSD coefficients in the former case. In a few cases where the FA costs of proposed algorithm are slightly higher than those of Pasko and BHM, FA delays by proposed algorithm are much lower. For example, MBPG shows a critical delay reduction of 8.19% on average over C1 for FIR2. The AT performances of all algorithms normalized with respect to the AT complexity of BIN for FIR1 and FIR2 are plotted in Figs. 3.10 and 3.11. These figures show that proposed MBPG can generate conspicuously faster transposed direct form FIR filters with low adder cost in general.

Fig. 3.10. Normalized AT complexity comparison for FIR1. For each N , the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM.

Fig. 3.11. Normalized AT complexity comparison for FIR2. For each N , the algorithms from left to right are: MBPG, C1, NRSCSE, Harley, BCSE, Pasko, CSD and BHM

MBPG is more computational efficient than C1. By measuring the execution time of both Matlab programs on the same PC with 3.2 GHz Pentium IV and 512 MB of system memory for all the filters tested, it is found that proposed algorithm is at least thrice faster

than C1. This is because the subexpression statistics in the *PT* table need to be established only once and updated locally thereafter. As the number of non-zero digits per vertex decreases exponentially from the root, the number of computations per vertex reduces rapidly as more vertices and isomorphic subgraphs are formed. The computational complexity of proposed algorithm can be analyzed as follows.

It is cited by [HAR96] that the number of nonzero digits is reduced by an average of 33% in CSD representation over the normal binary representation. Therefore, the average number of nonzero digits of an arbitrary number in CSD representation is $L/3$, where L is the wordlength. The number of computations required to decompose a vertex with j non-zero digits into its children vertices can be divided into two parts. It takes $C_2^j = j(j-1)/2$ probability computations to split a vertex, followed by $(j+1)(j-2)/2$ conditional probability computations to assign the $j-2$ non-zero digits to either the left or right child. Thus, the total number of operations required is $j^2 - j - 1$. For each coefficient with an average number of nonzero digits, $j = L/3$, the number of operations required is $(L^2 - 3L - 9)/9$ in the first level. After the first level of decomposition, the average number of nonzero digits in each child vertex is equal to $L/6$. Assuming the worse case scenario where there is no isomorphic subgraph, there will be two vertices per coefficient in the second level. Therefore, the number of operations required in the second level is given by $(L^2 - 6L - 36)/18$. The number of operations per coefficients becomes $(L^2 - 12L - 144)/36$ with four offspring vertices in the third level. Assuming a balanced tree decomposition, the depth of the tree is given by $\log_2(L/3)$. Since each vertex in the last level has no more than two nonzero digits, no computation is required in the last level. The total number of operations per coefficient is given by:

$$\sum_{i=1}^d \left\{ \frac{L^2}{9} \left(\frac{1}{2} \right)^{i-1} - \frac{L}{3} - 2^{i-1} \right\} = \frac{2L^2}{9} \left\{ 1 - \left(\frac{1}{2} \right)^d \right\} - \frac{L}{3} (d) - (2^d - 1) \quad (3.11)$$

where $d = \left\lceil \log_2 \frac{L}{3} \right\rceil$.

Since $d \leq \log_2(L/3)$, $2^d \leq L/3$. (3.11) is bounded by:

$$\frac{2L^2}{9} \left\{ 1 - \left(\frac{3}{L} \right) \right\} - \frac{L}{3} \left(\log_2 \frac{L}{3} \right) - \left(\frac{L}{3} - 1 \right) = \frac{2L^2}{9} - L - \frac{L}{3} \left(\log_2 \frac{L}{3} \right) + 1 \quad (3.12)$$

In practice, the word length of the coefficients is limited. From (3.12), the average number of operations per coefficient is less than 51 even if $L = 20$. The actual number of operations per coefficient is much lower as there will be shared vertices. Thus, the computation complexity for an N -tap FIR filter is $O(N)$.

In summary, the three propositions in entropy based decomposition have efficiently harnessed isomorphism in the generation of MBPG and the arithmetic operators have been ordered to balance the depth of the graph without annihilating the isomorphic vertices. This is unlike BHM for which the shortening of every single coefficient adder distance with minimal number of adders from existing fundamentals causes an upsurge in logic depth. This critical path extension in BHM has been shown to be more severe as the filter length increases. In fact, high order filters actually benefited more from proposed method as the intricate redundancy of many different subexpressions in a large number of coefficients can be quantitatively displayed by the entropy measure and hence better trade-off can be made without resorting to computational intensive exhaustive search.

3.6 Summary

In this chapter, an entirely different insight and design methodology to the FIR filter design automation problem is provided by exploiting the prowess of information theory on directed-acyclic graph representation of the transposed direct form structure of FIR filters. An appealing Multi-root Binary Partition Graph (MBPG) data structure has been devised for this purpose. Using this data structure, a set of fixed point coefficients can be decomposed into subsets of signed digit patterns whose coding redundancy can be theoretically assessed by their entropy and conditional entropy. To construct a reduced size MBPG, the partitioning of fixed point coefficients into different subsets of signed digits is guided by three propositions. The entropy and conditional entropy calculations

are based on the probability of occurrences of symbol pairs. Therefore, the proposed method is applicable to any positional representation of coefficients. The correlation of operand lengths and adder complexity is also illustrated with two's complement ripple carry adder, from which the number of FAs required to realize each vertex of an MBPG is analytically determined. This makes it possible to exploit the bit level information associated with each vertex to reduce the size of some arithmetic operators by partially modifying the graph topology. Experimental results on benchmark filters reported in the literature and design examples of communication filters based on D-AMPS and PDC cellular standards show that the proposed algorithm is capable of designing FIR filters with an average FA cost reduction of 40.38% over the baseline CSD implementation. The critical path delay has also been significantly reduced by 25.77% on average.

This chapter deals with the design methodology for fixed coefficient FIR filters. There are situations where not all coefficient multiplier outputs need to be generated simultaneously or the outputs of multiple FIR filters can be generated from a single multiple-accumulate architecture by time sharing the common hardware resources for multiple MCM problems. Such reconfigurable multiplier block can also be found in many other kernel functions in addition to FIR filters. The design of area-time efficient reconfigurable constant multiplier will be discussed in the next chapter.

Chapter 4

High-level Synthesis Algorithm for Design of Reconfigurable Constant Multiplier

4.1 Introduction

The preceding chapter presents a new design methodology for low complexity fixed FIR filters. As constant multiplication operations are found in many application domains, dedicated hardware resources and reconfigurable architecture generator capable of delivering a high peak computational density are both desired. Reconfiguration provides advantages in irregular architectures, and previous work has shown that converting multi-function cores to several reconfigurable field programmable gate array (FPGA) cores resulted in core area reduction of around 21% and a performance increase of 14% [MAC01]. In recent years, therefore, design methodologies for Reconfigurable Multiplier Block (ReMB) based on existing CSE and Graph Dependence algorithms have been proposed to reduce the hardware complexity for making a single constant multiplier block reconfigurable. Algorithms like [DEM03a, DEM04, DEM05b, DEM05c, SID04, DEM07] targets specially on FPGA implementation while solutions generated by [KIM07, TUM07] are more suitable for technology mapping to the standard cell library and optimized through an ASIC design flow. As reviewed in Chapter 2, all these existing methodologies use decimal numbers for coefficients representation. Due to the enormous number of alternatives, it is difficult to explore the totality of adder redundancies over all constant multiplications with no pattern exhibited explicitly in the coefficients for detection. Common subexpression sharing may be undermined as the DAG and partial sums of each coefficient multiplier are independently optimized before they are fused. In addition, the best existing algorithm requires exhaustive or quasi-exhaustive searches among existing DAG pairs, which results in high computational complexity.

To solve these problems, a new area-time efficient design methodology for Reconfigurable Constant Multiplier (RCM) block is proposed in this chapter. Fig. 4.1 shows the general structure of a RCM where a serial constant multiplier is shared in several multiplications involving N scalars specified a priori. The delay of the RCM is limited by the system's throughput, T_1 . The control logic operates at an internal rate, T_2 which is d time faster than T_1 . The constant multiplier can be implemented multiplication free by an adder and shifter network. To amortize the processing bandwidth and resource utilization over all multiplications, the logic depth and adder width of the combinatorial adders are to be minimized while eliminating the redundancy among different scalar multiplications. The logic depth is constrained by d and T_2 , which in turn are dependent on the cost of realizing the adders (for the same operand length, faster adder is more costly) in the critical path. These relations have led to the new formulation of RCM design as a programmatic resource scheduling and allocation problem described in this chapter.

Fig. 4.1. Reconfigurable constant multiplier (RCM)

The remaining of this chapter is organized as follows, Section 4.2 exemplifies several classes of system architectures whereby a reconfigurable constant multiplier (RCM) can be most cost effectively integrated without impediment to the system's throughput. In Section 4.3, the adder resources are pre-pensely generated and minimized by common subexpression elimination (CSE). The data dependencies of these adder resources are

modeled by a Data Flow Graph (DFG). A new heuristic algorithm is proposed to schedule the adders in the DFG to maximize the resource utilization over several control steps to generate a more area $\times$ time product efficient RCM architecture than previously reported designs. The experimental results are compared with the existing methods and discussed in Section 4.4. Section 4.5 will presented two new area-efficient architectures to implement circular convolution with the proposed RCM design methodology. This chapter is summarized in Section 4.6.

4.2 Hardware Reduction by Reconfigurable Constant Multiplications.

Making a multiplier reconfigurable for different predefined scalars can provide a significant overall cost savings in many DSP applications. This section suggests several interesting application scenarios that motivate the design of efficient RCM. These scenarios show that ingenious deployment of RCMs can lower the hardware cost significantly at little or no penalty to the overall system performance.

A linear time invariant system without feedback can be described by a linear transform.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1N} \\ c_{21} & c_{22} & \cdots & c_{2N} \\ \vdots & \vdots & \cdots & \vdots \\ c_{N1} & c_{N2} & \cdots & c_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \quad (4.1)$$

where N input data, $\mathbf{X} = \{x_i\}$ are mapped by an $N \times N$ matrix, $\mathbf{C} = \{c_{ij}\}$ to generate the outputs, $\mathbf{Y} = \{y_i\}$. This matrix-vector multiplication is an expensive operation, which is also found in the state-space equations of systems involving feedback. In many DSP applications such as digital filtering [FLO05, AKS08, JOH05, MAH08, MAC04], error detection and correction [WU06], fast Fourier and Consine transforms [CHA07], autoregressive model [SHA06], the matrix, $\mathbf{C}$ consists of constants that are determined a priori from the system specifications. The parallel processing of $\mathbf{Y} = \mathbf{C} \times \mathbf{X}$ requires the replication of N independent sum-of-products (SOP) processing units. Each SOP performs N discrete constant multiplications with different input variables and $N - 1$ additions. In many situations, this quadratic increase in hardware cost is prohibitive due

to the limitation in design areas. However, if the N outputs are merged in time, the $N \times N$ array of SOP units can be collapsed into N parallel scalar multiplications such that each input data is multiplied by only one out of N constant factors at any cycle, as shown in Fig. 4.2. The scalar products in each cycle are summed by an adder tree to produce an output element of Y . The adder tree can be pipelined, if necessary, to improve the throughput. In this example, each column of N processing units is reduced to a single RCM.

Fig. 4.2. RCM blocks in matrix multiplication

Goertzel algorithm has been used to reduce the complex multiplications to real multiplications in the recursive computation of Discrete Fourier Transform (DFT). Recently, it has been shown that Goertzel kernels also offer significant improvement in quantization noise performance for the Discrete Cosine Transform (DCT) used in many multimedia and image compression applications [DEM03b]. The transfer function of an N -point DCT is given by:

$$H(z) = \frac{P_k(1 - z^{-1})}{1 - 2\beta_k z^{-1} + z^{-2}} \quad (4.2)$$

where $k \in [0, N)$ is the frequency bin index. $\beta_k = \cos(k\pi/N)$ and $P_k = 4\beta_k/N$ for all k except $k=1$. For $k=1$, $P_1 = (2\sqrt{2}/N)\beta_1$.

This transfer function can be realized with several resonator configurations of recursive Goertzel filter [DEM03b]. All Goertzel kernel structures involve a pre-multiplication of the difference signal by $2\beta_k$ in the feedback path and a post-multiplication of the output signal by P_k . The two sets of scalars can be pre-determined by the discrete cosine function according to the frequency bin index, k and the transform length, N . The same recursive structure can be employed for all frequency bins by multiplexing it in time. One such architecture using two RCM units is shown in Fig. 4.3. Pre-stored lookup tables have been avoided as the RCM can be implemented multiplication free by combinational adders. To cut down the circuit cost, the RCM can be designed to share the adder resources for different scalar multiplications of different frequency bins by exploiting the time multiplexed system.

Fig. 4.3. Recursive Goertzel filter

Multirate digital signal processing uses decimators and interpolators to change the sampling rates of a system internally to maximize the performance of DSP system while keeping the cost down. Using the Noble relation [VAI93], a complex data flow graph can

be decomposed into several simpler DSP blocks, which can be processed in parallel at a slower rate. An example of a polyphase decomposition is shown in Fig. 4.4. The input signal, x is down-sampled into M subsequences by an M -fold decimator and each of the M decimated input sequences is fed to a sub-filter. Each subfilter is reduced by a factor of M from the original filter length. The decimator generates one output for every M input samples, and each sub-filter can thus be processed at M times slower rate than that of the original input sequence, $x[n]$. In each decimation filter, each MAC implementation is realized by a reconfigurable multiplier block. One input sample will be multiplied with N constants generated from the reconfigurable multiplier block sequentially and the products are stored into the partial sum store. As the input sample frequency has been reduced by M times in each subfilter, the time interval between two samples are much longer. Therefore, the multiplier block in each sub-filter can afford to be processed by a serial-parallel architecture by merging L different coefficient multipliers into a RCM. The adder resources of the L coefficient multipliers can then be shared by processing them serially, provided that the total time taken to produce the L scalar multiples is able to meet the slower output rate of the subfilter. If the subfilter sampling period is not sufficient for one RCM to generate all distinct coefficient multiplier outputs, the filter coefficients can be partitioned into several subsets of distinct constants for an input sample to be concurrently multiplied by a few RCM units.

Fig. 4.4. Polyphase decimation filters ($M=3$).

4.3 Proposed High Level Algorithm for RCM Design

The scenarios depicted in the previous section show that RCM units are amenable for integration into multi-rate digital systems or other subsystems when not all constant multiplications are processed in parallel. A practical consequence of these scenarios is the number of different multiplication operations is bound by the ratio of the system's sampling clock period and the critical path delay of the RCM. The proposed design methodology allows the elementary operators in a RCM to assume different input operands to fulfill a pre-specified set of scalar multiplications by suitably rearranging the input bits through a multiplexer network. The latency of the RCM is thus dependent on the control step granularity and the number of control steps used to execute the most time critical operation among a group of pre-specified multiplications. The former is defined by the delay of the most critical adder and the latter is constrained by the logic depth of the RCM. The resolution of any control step can be minimized by reducing the length of the adders in that control step and the worst case delay of each adder can be independently optimized, if necessary, by different parallel structures at the cost of hardware complexity. To meet the latency constraint with the best utilization of silicon resources, the design methodology emphasizes on a global reduction of adder depth while optimizing the number of adders, and the average sizes of adders and multiplexers. If the latency constraint can not be met by a single RCM, this process itself also identifies under-utilized hardware resources and provides an insight into the partitioning of operations into multiple RCM units in a partially parallel implementation.

4.3.1 Design Problem Formulation

To allow for programmability of different scalar product generations, temporally independent adders are shared across different constant multiplications. This is illustrated with an example of four different constant multiplications using the values from Table II of [POT96]. Fig. 4.5(a) shows the DAG representation of the four constants after eliminating the common subexpressions. Since a RCM makes only one multiplication at a time, the span of the input variable can be divided into as many number of control steps as, if not more than, the number of adders in the critical path of the DAG. The adders are

then scheduled into the control steps without violating the precedence of any connected pair of adders in the DAG. The grey color adders in Fig. 4.5(a) are called the mobile adders. Mobile adders are critical variants to reduce the implementation complexity as they have some flexibility to be scheduled into more than one control steps without increasing the logic depth of the RCM.

Fig. 4.5. Example of RCM design (a) DAGs of constant multipliers (b) Reduced DFG (c) Mapping to RCM

To schedule the adders, the data dependency of DAG is compacted into a primitive Data Flow Diagram (DFG). A DFG, G has a node set V and an edge set E . Each node, $v \in V$ represents an adder, and each edge $e \in E$ is an interconnection between two adders or between an adder and a primary input or output. Each edge is annotated with the output shift and a directed edge implies an intra-precedence constraint of two adders. If the shift amount is negative, it indicates that the node to which this edge is connected is a subtractor. For convenience, adder and subtractor are both called adder. The source and sink of the DFG are the input and output of the RCM, respectively.

In a properly scheduled DFG, each node can only fire when all its predecessor nodes have fired. Upon firing, a node computes the required value and places it on all its output edges. Fig. 4.5(b) shows a primitive DFG by merging some adders that are not executed concurrently. The number of adders has been reduced significantly from 10 to 4, but some adders have more inputs than needed to fire. This happens because every time an adder is eliminated, it is accompanied by an increase in the fanout of some other adder. These contentions are resolved by introducing multiplexers as the point-to-point interconnecting elements between two nodes so that every node, v in a properly scheduled DFG has exactly two predecessor nodes firing at time t and satisfies the following dependency relation.

$$v(t) = 2^{e_{l(v(t))}} l(v(t)) + 2^{e_{r(v(t))}} r(v(t)) \quad (4.3)$$

where $v(t)$ is the output of v at time t . $l(v(t))$ and $r(v(t))$ are the left and right predecessor operators of $v(t)$. e_u denote the amount of shifts applied to $u \in \{l(v(t)), r(v(t))\}$.

To generate a scheduled DFG, cut lines are drawn through the primitive DFG of Fig. 4.5(b) so that only one adder depth is allowed between two cut lines. For every operator that has a fan-in greater than two, multiplexers are placed at the cut edge. Control steps, t_1 , t_2 and t_3 define the boundaries between the outputs of one adder or the source, and the inputs of another adder or the sink, by hard wirings or through multiplexers. The

resolution of each control step needs not be equal. It is determined by the maximum delay of the adders in the critical paths that are allocated to the control step and their maximum input connection delay through the multiplexer, if the inputs are not directly wired to the adders. Fig. 4.5(c) shows the architectural mapping of the RCM from the scheduled DFG. Therefore, the design of an area-time efficient RCM can be viewed as the transformation of a given set of constants into a reduced DFG by scheduling and merging of the mobile adders that are not used simultaneously in the same control step. The quality of the solution is measured in terms of the adder and interconnection cost. The adder cost is contributed by the total number of nodes in the set V of the final DFG while the cost of the interconnects is measured by the number and the size of multiplexers used.

4.3.2 Common Subexpression Elimination

Without affecting the adder depth, the spatial redundancy in the original DAG is converted into temporal redundancy in the primitive DFG to increase the mobility of as many nodes as possible. The nodes in the proposed DFG are atomic in that they encapsulate information about the elementary operations at bit level granularity. To preserve the atomic information of the nodes after removing the redundant adders, the detection and elimination of common subexpressions are better carried out in a positional representation.

As discussed in Chapter 2, signed-digit representations are widely used for coefficient quantization and on-line arithmetic due to its attractive property for digit-serial and distributive operations. Minimum signed digit (MSD) [ARN93] and its canonical signed digit (CSD) subset are two popular symmetric binary signed digit representations. They allow a fixed point number to be represented by a minimal number of signed digits, such that its multiplication with a variable can be realized with reduced depth adder tree. The uniqueness of CSD has been exploited by many CSE algorithms [PAS99, PEI02] to reduce the search space for common subexpressions but the canonicity also limits the number of common subexpressions. For a given magnitude response specification, a signed powers-of-two coefficient set with more sharable subexpressions of 101 and $10\overline{1}$ can be synthesized from CSD or MSD by [XU07b]. On the other hand, the advantage of

having more subexpression sharing with binary representation [MAH08, AKS08] is often offset by the higher number of nonzero power-of-two terms over MSD and CSD. Since solutions for MSD and CSD based on weight-two common subexpressions have the same minimum logic depth of $\lceil \log_2 n \rceil$, where n is the maximum number of non-zero digits in a coefficient, the LDs of CRA [XU05b] and NR-SCSE [PEI02] reported in [MAH08] are incorrect. Experiments run on random coefficient sets for CSE using binary, CSD and MSD representations [AKS08] concluded that MSD is preferred when seeking minimum delay solution. As minimum delay solution is critical for the scenarios discussed in Section 4.2, MSD representation of integers will be considered.

The latency of the RCM is minimized by minimizing the adder depth. If the latency is relaxed, more control steps than the adder depth can be allocated which makes every adder mobile. Finding an optimal CSE solution for a set of integers in the MSD space is an NP complete problem. The search space is reduced by detecting only common subexpressions of hamming weight 2 instead of all possible common subexpressions in the set of integers. These common subexpressions can be categorized into two different types. An even parity subexpression takes the form, IOI, and an odd parity subexpression takes the form, IO $\bar{I}$, where $I \in \{1, \bar{1}\}$ and O is either void or a string of zeros. The number of zeros in O is the distance of the subexpression. This distinction allows the frequencies of different weight-two subexpressions to be tracked by a simple two dimensional *PT* array. This *PT* array has the same definition with *Definition 3.3* in upper (lower) rows and numerical values inside. However, since MSD number system is used in this section, the weight two subexpressions can probably have the two non-zero digits adjacent. This caused this *PT* array's j -th column representing the number of occurrences of positive (negative) subexpressions with $j-1$ zeros, where $j = 1, 2, \dots, B-1$ and B is the wordlength of the maximum coefficient in the given coefficient set.

As an example, an MSD representation for the constants of Fig. 4.5 and its corresponding *PT* array is given by:

$$H = \begin{Bmatrix} 8 & 1 & 5 \\ 8 & 3 & 1 \\ 6 & 2 & 1 \\ 1 & 0 & 5 \end{Bmatrix} = \begin{Bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & \bar{1} \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \bar{1} \\ 1 & 0 & 1 & 0 & 0 & \bar{1} & 0 & 0 & \bar{1} & \bar{1} \\ 0 & 0 & 1 & 0 & 0 & \bar{1} & \bar{1} & 0 & 0 & 1 \end{Bmatrix}$$

$$PT(H) = \begin{bmatrix} 5 & 2 & 3 & 3 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 3 & 2 & 2 & 1 & 3 & 3 \end{bmatrix}$$

An entry in the PT array with value greater than 1 indicates the presence of a common subexpression in C . The number of adders saved by every common subexpression is one less than its frequency in the PT array. The MSD generation method of [PAR02] is adopted to generate all possible sets of MSD numbers for a given set of constants. With the help of the PT arrays, the MSD set with the highest total saving of adders is selected. The common subexpression with the highest value in the PT array is first identified and an adder node, v is created in a DAG to add the two operands corresponding to the input variable shifted by the positions of the two non-zero digits. If there are more than one entry with the same highest value, the one with the least distance, d , is selected to minimize the wordlength of the adder. The distance d and coefficients from which the common subexpression is detected are also recorded. These information are used to determine the amount of shifts applied to the left and right operands of v and the firing time, t of v . The two nonzero digits of the common subexpression are replaced by zeros in all their occurrences in the set of MSD coefficients. The PT array is updated to reflect the removal of this subexpression. This process of searching for the highest frequency subexpression and the creation of adder node in the DAG to eliminate the common subexpression from the MSD coefficient set is repeated until no entry in the PT array is greater than 1. Finally, new nodes are created to add the operands corresponding to the remnant nonzero digits (which are not parts of the common subexpressions) to the outputs of the existing nodes in the DAG accordingly.

4.3.3 Proposed Mobile Operator Scheduling Scheme

After the above global CSE process, the total number of adders in the DAG, G can be further reduced by merging the adders allocated for different constant multiplications. There are many different ways of merging the adders without violating the intra-

precedence relationship of G by routing the inputs to an adder from the output of another adder or the primary inputs through multiplexers. The objective of the proposed adder scheduling scheme is to minimize the total cost of the adders and multiplexers within the given latency constraint. Let $D(G)$ be the adder depth of G . The latency constraint can be sliced into a number of control steps, $t_1, t_2, \dots, t_{max}$ so that $t_{max} \geq D(G)$ and each control step, t_i has a time resolution sufficient to complete the longest latency addition scheduled to that control step. The time resolution of a control step can be reduced by minimizing the word lengths of the merged adders allocated to the control step without resorting to faster but more complex adder structure. This is possible if there exist mobile adders.

An adder is said to be mobile if it can be assigned to more than one control step in a DFG. The mobility of a node, v in G is defined as $\mu(v) = t_{max} - \delta(v)$, where $\delta(v)$ is the number of nodes in the longest path through v from a root to a leaf. The significance of the mobility of a node is that before scheduling, the probability of a node, v appearing in a control step t_i , denoted by $p_v(t_i)$, is equal to $\{\mu(v)+1\}^{-1}$ for $t_i \in [t_e, t_l]$ and zero outside this time window, where t_e and t_l are the earliest and the latest control steps the node can fire due to the intra-precedence relation of the DFG. Scheduling a mobile adder to a particular control step fixes its probability to 1 in that control step and to 0 in all other control steps. The probability displacement before and after scheduling a mobile node, v to a specific control step $t_j \in [t_e, t_l]$ is given by:

$$\Delta p_v(t_i) = \begin{cases} \frac{\mu(v)}{\mu(v)+1} & \text{if } t_i = t_j \\ -\frac{1}{\mu(v)+1} & \text{if } t_i \neq t_j \end{cases} \quad (4.4)$$

where $t_i, t_j \in [t_e, t_l]$.

This change in adder probability is associated with the opportunity cost of merging a mobile adder with the mutually exclusive adders in any control step within the time window defined by $[t_e, t_l]$. Two adders in the same control steps are said to be mutually exclusive if they are not used to generate the same scalar multiplier output simultaneously. Merging two mutually exclusive adders results in a saving of one adder

but the merged adder assumes the length of the longer adder. In other words, the output of the shorter length adder will be expanded by an amount equal to the difference between the two adder lengths. This will increase the bitwidths of its successive adders and multiplexers. To minimize the cost of RCM, it is important to schedule the mobile node to a control step that will minimize the disparities in the bitwidths among mutually exclusive adders.

The output of each node, $v(t) = a(v) \cdot x(t)$, where $x(t)$ is the input variable, x to the RCM, and the constant $a(v)$ can be expressed in a signed digit representation as follows.

$$a(v) = \sum_{i=1}^{h(v)} s_i 2^{w_i} \quad (4.5)$$

where $h(v)$ is the hamming weight [XU07a] of $a(v)$, $s_i \in \{-1, 1\}$ is the i -th signed digit and $w_i \in \{0, 1, 2, \dots, w_{max}\}$ is the weighted bit position of s_i .

From Fig. 3.6 in Section 3.4, if the length of the variable, x , is fixed, the bitwidth of an adder is largely dependent on the difference between the maximum most significant and the maximum least significant signed digit positions of its input operands. Since the signed digits of $a(v)$ are inherited from those of its input operands and the input operands to an adder are commutative, a meaningful metric to assess the disparity of mutually exclusive adders is through the aggregate distance of all signed digits to the leading signed digit position, w_l . Let

$$W(v) = \sum_{i=1}^{h(v)-1} (w_l - w_i) \quad (4.6)$$

The disparity of a node, v in control step t_i is defined as

$$\Delta W_v(t_i) = \left| W(v) - \frac{1}{M_i} \times \left(\sum_{j=1}^{M_i} W(u_j) \right) \right| \quad (4.7)$$

where M_i is the total number of existing nodes in control step t_i that are mutually exclusive to v .

The aggregate distance, $W(v)$ measures how compact the sign digits in $a(v)$ are distributed and the disparity of v is an indicator of the bitwidth variation by merging the existing adders in the same control step with v . Fig. 4.6 shows the disparities of mobile nodes v_4 , v_5 and v_6 in control steps t_1 to t_3 after nodes, v_1 , v_2 and v_3 have been scheduled. Scheduling v_4 to t_1 , v_5 to t_1 or t_2 and v_6 to t_2 will minimize the disparity and reduce the costs of the merged adders and multiplexers.

Fig.4.6. Disparities of operators scheduled in different control steps

To minimize the bitwidth of the merged adders and multiplexers, mobile nodes should be scheduled to control steps with minimal disparity. However, scheduling a mobile node to a particular control step will restrict the mobility of its predecessor and successor nodes and hence the probabilities and disparities of their scheduling in other control steps. The overall effect can be accounted by the probability displacement and the time window reduction of the predecessor and successor nodes when evaluating the opportunity cost of scheduling a mobile node, v in control step $t_i \in [t_e, t_l]$. This opportunity cost is given by:

$$\begin{aligned}
\text{Cost}(v, t_i) = & \sum_{t=\bar{t}_e}^{t_i} \Delta p_v(t) \cdot \Delta W_v(t) + \sum_{u \in \text{parent}(v)} \sum_{t=\bar{t}_e}^{\bar{t}_i} \Delta p_u(t) \cdot \Delta W_u(t) \\
& + \sum_{u \in \text{children}(v)} \sum_{t=\bar{t}_e}^{\bar{t}_i} \Delta p_u(t) \cdot \Delta W_u(t)
\end{aligned} \tag{4.8}$$

where $[\bar{t}_e, \bar{t}_i]$ denotes the reduced time windows of the predecessor or successor nodes in general due to the scheduling of v to control step, t_i . $\text{parent}(v)$ and $\text{children}(v)$ refer to the immediate predecessors and successors of v , respectively.

To minimize the computational complexity, the opportunity costs are evaluated in batches of identical mobility nodes, commencing from the least mobility nodes. When a node with the least opportunity cost is scheduled, the mobilities of all its affected predecessor and successor nodes are recalculated. After all nodes with the least mobility have been allocated into the DFG, the opportunity costs of all mobile nodes with the next higher mobility are evaluated and the least cost mobile node will be scheduled. The process repeats until all nodes in the DAG have been scheduled into the DFG.

4.3.4 Adder Merging and Multiplexer Splitting

After all nodes in the DAG have been scheduled into a DFG, the DFG are reduced by merging mutually exclusive nodes in the same control step. For the calculation of disparity, each node, v in the DAG is annotated with a set of labels, $\phi(v)$ to indicate its phases of computation in the RCM. A RCM is programmed to output only one product of an input variable, x and a constant, c_i in any one phase but each node in a reduced DAG may involve in more than one phase of computation due to the presence of common subexpressions. Using the index of the coefficient as phase number, the criterion for merging two scheduled nodes, v and u is given by $\phi(v) \cap \phi(u) = \emptyset$. Merging two mutually exclusive nodes, v and u introduces at least a multiplexer at either or both inputs of the merged node, $\hat{v}$, where

$$\text{child}(\hat{v}, t) = \overline{c(t)} \cdot \text{child}(v, t) + c(t) \cdot \text{child}(u, t) \tag{4.9}$$

where the control input to the multiplexer $c(t) = 0$ if $t = \phi(v)$ and 1 if $t = \phi(u)$. $child(v, t) \in \{l(v(t)), r(v(t))\}$ and $child(u, t) \in \{l(u(t)), r(u(t))\}$ with the constraint that $l(v(t)) \neq r(v(t))$.

In general, more than one mutually exclusive node can be merged after scheduling. Every legitimate merging increases the indegree of the merged node in the DFG. Large multiplexers for high indegree nodes can be decomposed into smaller multiplexers. The data inputs to the large multiplexer can be split and rescheduled into one or more smaller multiplexers. The split inputs can be merged into existing smaller multiplexers in the earlier control steps instead of forming new multiplexers provided that these inputs are not required by the adders in two consecutive control steps. Multiplexer splitting attempts to speed up the connection time by reducing the fanins of large multiplexers at the expense of increasing the fanins of smaller multiplexers. Fig. 4.7 shows the result of multiplexer load balancing by extracting four data inputs from an 8-to-1 multiplexer into a new 4-to-1 multiplexer so that the interconnection time is reduced in the critical path.

Fig. 4.7. RCM design (a) before (b) after multiplexer splitting

An m -to-1 multiplexer has m data inputs and the number m is also called the fanin of the multiplexer. It should be noted that to reduce the adder width, appropriate amount of shifts have been applied to the inputs and output of each adder during the scheduling and merging processes so that the most significant digit of its $a(v)$ is either 1 or -1 . Thus, the

data inputs to the multiplexer have non uniform bitwidth. Unfortunately, the cost of a multiplexer is a function of its bitwidth and the bitwidth of a multiplexer must be catered to the largest data input [TUM07]. This cost may be reduced by splitting a large multiplexer into two smaller multiplexers. Which data inputs to be split can be judiciously decided by examining the disparity of each fanin, which takes a similar form as (4.7). The disparity of a data input, f_i , to an m -to-1 multiplexer is defined as:

$$\Delta b(f_i) = \left| b(f_i) - \frac{1}{m} \times \left(\sum_{j=1}^m b(f_j) \right) \right| \quad (4.10)$$

where $b(f_i)$ is the bitwidth of the i -th data input of the multiplexer.

$\Delta b(f_i)$ measures the deviation of the bitwidth of a specific data input, f_i from the average bitwidth of a multiplexer. For each multiplexer, MUX with $m \geq 4$, it is first assumed that it is cost effective to decompose it into two multiplexers, MUX₁ and MUX₂. Let F be the set of data inputs to MUX sorted by their bitwidths, i.e., $F = \{f_i\}_{i=1}^m$ with $b(f_i) \leq b(f_j)$ if $i < j$. Initially, the data input sets of MUX₁ and MUX₂ are set to $F_1 = \{f_i\}_{i=2}^m$ and $F_2 = \{f_1\}$, respectively. Then, the disparity of the smallest data input of F_1 , i.e., $\Delta b_1(f_j)$ for $j = \arg\{\min(F_1)\}$ is computed and compared with the average bitwidth of MUX₂, $ave(F_2)$. If $\Delta b_1(f_j) < ave(F_2)$, then $F_1 = F_1 - \{f_j\}$ and $F_2 = F_2 \cup \{f_j\}$. The process is repeated with the next smallest data inputs of F_1 until all data inputs of F_1 have been evaluated. If the final fanin of MUX₂, $|F_2| \geq 2$, MUX will be decomposed into MUX₁ and MUX₂. Otherwise, MUX will be preserved.

The flow chart in Fig. 4.8 summarizes the proposed optimization strategy for the design of RCM.

Fig. 4.8. Flow chart of the proposed algorithm

4.3.5 Design Example

A design example of a RCM for a 8-point Goertzel recursive DCT from Fig. 7 of [DEM03a] is used to illustrate the proposed adder scheduling algorithm. The three constants of the RCM are $C = \{362, 392, 473\}$. The best MSD set with the highest common subexpression frequency happens to be its CSD subset and its PT array is given as follows:

$$C = \begin{Bmatrix} 362 \\ 392 \\ 473 \end{Bmatrix} = \begin{Bmatrix} 1 & 0 & \bar{1} & 0 & \bar{1} & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & \bar{1} & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \bar{1} & 0 & \bar{1} & 0 & 0 & 1 \end{Bmatrix}$$

$$PT(C) = \begin{bmatrix} 0 & 3 & 0 & 0 & 0 & 2 & 0 & 1 & 1 \\ 0 & 3 & 1 & 4 & 1 & 2 & 0 & 0 & 0 \end{bmatrix}$$

Following the CSE procedure described in Section 3.2, the reduced DAG is obtained in Fig. 4.9.

Fig. 4.9. DAG of constant multipliers with $C = \{362, 392, 473\}$.

The number of control steps is assumed to be the minimal adder depth of the reduced DAG for the most stringent timing requirement. Therefore, $t_{max} = 3$. From the DAG, the number of nodes in the longest path through each node is given by: $\delta(v_1) = \delta(v_2) = \delta(v_3) = 3$ and $\delta(v_4) = \delta(v_5) = \delta(v_6) = 2$. The computation phase $\phi(v)$ is defined as the set of indices of the coefficients that contain the output of node v as their partial sum. The computation phase in this DFG are therefore $\phi(v_1) = \{0, 1, 2\}$, $\phi(v_2) = \phi(v_3) = \{0\}$, $\phi(v_4) = \{1\}$, $\phi(v_5) = \phi(v_6) = \{2\}$. The three fixed nodes are identified by $\mu(v_1) = \mu(v_2) = \mu(v_3) = 0$. These three adders will be directly scheduled into control steps, t_1 , t_2 and t_3 , respectively. Since $\mu(v_4) = \mu(v_5) = \mu(v_6) = 1$, it is necessary to compute the opportunity costs of scheduling these mobile nodes into specific control steps within their legitimate time windows.

Consider v_4 . From (4.7), since $a(v_4) = 10\bar{1}0001$, the aggregate distance of v_4 is $W(v_4) = (w_3 - w_1) + (w_3 - w_2) = (6 - 0) + (6 - 4) = 8$. v_4 can be scheduled into either control step t_2 or t_3 . According to (4.8), its disparity in t_2 and t_3 can be calculated as follows. If v_4 is scheduled into t_2 , $\Delta p_{v_4}(t_2) = 0.5$ and $\Delta p_{v_4}(t_3) = -0.5$. Since v_2 is the only scheduled (fixed) node in t_2 and $\phi(v_2) \cap \phi(v_4) = \emptyset$, v_2 and v_4 are mutually exclusive. From $a(v_2) = 10\bar{1}0\bar{1}01$, $W(v_2) = (w_4 - w_1) + (w_4 - w_2) + (w_4 - w_3) = (6 - 0) + (6 - 2) + (6 - 4) = 12$ and $\Delta W_{v_4}(t_2) =$

$|W(v_4) - W(v_2)| = |8 - 12| = 4$. Similarly, v_3 is the only fixed node in t_3 and $\phi(v_3) \cap \phi(v_4) = \emptyset$. From $a(v_3) = 10\bar{1}0\bar{1}0101$, $W(v_3) = (w_5 - w_1) + (w_5 - w_2) + (w_5 - w_3) + (w_5 - w_4) = (8 - 0) + (8 - 2) + (8 - 4) + (8 - 6) = 20$ and $\Delta W_{v_4}(t_3) = |W(v_4) - W(v_3)| = |8 - 20| = 12$. As v_4 has no unscheduled predecessor or successor node, according to (4.9), the opportunity cost of scheduling v_4 into step t_2 is $\text{Cost}(v_4, t_2) = 0.5 \times 4 + (-0.5) \times 12 = -4$. If v_4 is scheduled into t_3 , $\Delta p_{v_4}(t_2) = -0.5$ and $\Delta p_{v_4}(t_3) = 0.5$. Hence, $\text{Cost}(v_4, t_3) = (-0.5) \times 4 + 0.5 \times 12 = 4$.

Consider v_5 . Since $a(v_5) = \bar{1}001$, $W(v_5) = 3$. v_5 can be scheduled into either t_1 or t_2 . Since the only scheduled adder in t_1 , v_1 , is involved in all phases of computation, $\phi(v_1) \cap \phi(v_5) \neq \emptyset$. There is no fixed node that is mutually exclusive with v_5 in t_1 and $\Delta W_{v_5}(t_1) = |W(v_5) - 0| = 3$. Also, v_2 is the only fixed node in t_2 and $\phi(v_2) \cap \phi(v_5) = \emptyset$. Thus, v_2 and v_4 are mutually exclusive and $\Delta W_{v_5}(t_2) = |W(v_5) - W(v_2)| = 9$. If v_5 is scheduled into t_1 , $\Delta p_{v_5}(t_1) = 0.5$ and $\Delta p_{v_5}(t_2) = -0.5$ and its successor node, v_6 , can still be scheduled into t_2 and t_3 . There is no change in the probability of v_6 and $\Delta p_{v_6}(t_2) = \Delta p_{v_6}(t_3) = 0$. According to (4.9), $\text{Cost}(v_5, t_1) = 0.5 \times 3 + (-0.5) \times 9 = -3$. If v_5 is scheduled into t_2 , $\Delta p_{v_5}(t_1) = -0.5$ and $\Delta p_{v_5}(t_2) = 0.5$. Its successor node, v_6 , has to be scheduled into t_3 . Therefore, $\Delta p_{v_6}(t_2) = -0.5$ and $\Delta p_{v_6}(t_3) = 0.5$. Since $a(v_6) = 1000\bar{1}0\bar{1}001$, $W(v_6) = 19$. There is only one fixed node, v_2 , in t_2 and $\phi(v_2) \cap \phi(v_6) = \emptyset$. Thus, $\Delta W_{v_6}(t_2) = |W(v_6) - W(v_2)| = 7$. v_3 is the only fixed node in t_3 and $\phi(v_3) \cap \phi(v_6) = \emptyset$. Thus, $\Delta W_{v_6}(t_3) = |W(v_6) - W(v_3)| = 1$. According to (4.9), $\text{Cost}(v_5, t_2) = [(-0.5) \times 3 + 0.5 \times 9] + [(-0.5) \times 7 + 0.5 \times 1] = 0$.

Now, consider node v_6 . Since v_6 can be scheduled into either t_2 or t_3 . If v_6 is scheduled into t_2 , $\Delta p_{v_6}(t_2) = 0.5$ and $\Delta p_{v_6}(t_3) = -0.5$. Its predecessor node, v_5 , has to be scheduled into t_1 . Thus, $\Delta p_{v_5}(t_1) = 0.5$ and $\Delta p_{v_5}(t_2) = -0.5$. $\text{Cost}(v_6, t_2) = [0.5 \times 7 + (-0.5) \times 1] + [0.5 \times 3 + (-0.5) \times 9] = 0$. If v_6 is scheduled into t_3 , $\Delta p_{v_6}(t_2) = -0.5$ and $\Delta p_{v_6}(t_3) = 0.5$. v_5 can still

be scheduled into t_1 and t_2 . Since there is no change in the probability of v_5 , $\Delta p_{v_5}(t_1) = \Delta p_{v_5}(t_2) = 0$. According to (4.9), $\text{Cost}(v_6, t_3) = (-0.5) \times 7 + 0.5 \times 1 = -3$.

The traces of computation are summarized in Table 4.1. In Stage 1, the minimum opportunity cost is -4 . Consequently, v_4 is scheduled into t_2 and become a fixed node. The opportunity costs of scheduling v_5 and v_6 need to be re-evaluated after updating their mobilities. The traces of computation are shown in Stage 2 of Table 4.1, from which v_6 is scheduled into t_3 with the minimum opportunity cost. After fixing v_6 in t_3 , the only mobile node is v_5 . It can be scheduled in either t_1 or t_2 as computed in Stage 3. The scheduling of v_5 into t_1 has less opportunity cost, hence v_5 is scheduled into t_1 .

Table 4.1. Traces of computation of the proposed scheduling algorithm.

Stage	v	$W(v)$	t	$\Delta W_v(t)$	$\text{Cost}(v, t)$
1	v_4	8	t_2	4	-4
			t_3	12	4
	v_5	3	t_1	3	-3
			t_2	9	0
	v_6	19	t_2	7	0
			t_3	1	-3
2	v_5	3	t_1	3	-2
			t_2	7	-2
	v_6	19	t_2	9	2
			t_3	1	-4
3	v_5	3	t_1	3	-2
			t_2	7	2

After all mobile nodes have been fixed, the mutually exclusive nodes in the same control steps are merged. Multiplexers are inserted between the nodes with fanin greater than two and their immediate predecessor nodes or primary inputs. Since all multiplexers have fanin less than 4, no multiplexer splitting is performed. The optimized DFG and the RCM circuit are shown in Fig. 4.10.

Fig. 4.10. RCM of the design example (a) DFG (b) Circuit architecture

4.4 Experimental Results and Discussion

In this section, the area and time complexities of the RCM solutions generated by proposed method are evaluated based on ASIC implementation and compared against other methods in the literature [DEM03a, DEM04, DEM05c]. These competing ReMB approaches are designed to utilize the feature that XILINX Virtex FPGA implementations where a Look-up Table (LUT) is programmed to have a multiplexer at one input of the adder/subtractor without any extra area cost [DEM03a]. Although this objective is not for ASIC mapping, results of these approaches on ASIC implementation are presented to proof their performance on ASIC and show the motivation of developing new design automation algorithm for ASIC mapping. In the first part of the experiments, four commonly cited sets of constants in the reconfigurable multiplier blocks are considered. Set A is obtained from Fig. 6 of [DEM04], Sets B and C are taken from Fig. 7 and Fig. 8 of [DEM03a], respectively and Set D is taken from Fig. 6 of [DEM05c]. Unitless cost function in [TUM07] is adopted for the area estimate. In this estimate, the cost of an operator is equal to $a \cdot k$, where k is the bitwidth of the operator and a is a constant of proportionality. The value of a can be characterized by the implementation

technology. Technology mapping of typical operators using a commercial $0.18\ \mu\text{m}$ standard cell library [TUM07] yields $a_{m-1\text{ multiplexer}} = 14 \cdot m$, $a_{\text{adder}} = 67$, $a_{\text{subtractor}} = 75$ and $a_{\text{adder/subtractor}} = 98$ measured in μm^2 . With Artisan TSMC $0.18\ \mu\text{m}$ standard cell library, $a_{m-1\text{ multiplexer}} = 13.3 \cdot m$ and $a_{\text{adder}} = 69.9$, which show that the same cells from different libraries developed for the same process technology have very similar areas. With Avant! Passport $0.35\ \mu\text{m}$ standard cell library, $a_{m-1\text{ multiplexer}} = 1.17 \cdot m$, $a_{\text{adder}} = 5.0$, $a_{\text{subtractor}} = 5.66$ and $a_{\text{adder/subtractor}} = 7.0$, measured in terms of equivalent gates.

Table 4.2. Area cost estimation for data set A

Algorithm	Adder	Subtractor	Adder/Subtractor	Multiplexers	(I) $0.18\ \mu\text{m}$	(II) $0.35\ \mu\text{m}$
DAG Fusion [TUM07]	0	0	$1 \times (12\text{-bit})$ $1 \times (14\text{-bit})$	$1 \times \text{Mux}_{3\text{-to-1}}(14\text{-bit})$ $1 \times \text{Mux}_{7\text{-to-1}}(16\text{-bit})$	4704	361.7
ReMB [DEM04]	$1 \times (10\text{-bit})$	$1 \times (12\text{-bit})$	$1 \times (10\text{-bit})$ $2 \times (11\text{-bit})$ $1 \times (12\text{-bit})$ $1 \times (16\text{-bit})$	$1 \times \text{Mux}_{2\text{-to-1}}(8\text{-bit})$ $1 \times \text{Mux}_{2\text{-to-1}}(9\text{-bit})$ $2 \times \text{Mux}_{2\text{-to-1}}(10\text{-bit})$ $1 \times \text{Mux}_{2\text{-to-1}}(11\text{-bit})$ $1 \times \text{Mux}_{2\text{-to-1}}(12\text{-bit})$ $1 \times \text{Mux}_{2\text{-to-1}}(16\text{-bit})$	9578	715.3
ASAP [CHE09a]	0	0	$1 \times (12\text{-bit})$ $1 \times (14\text{-bit})$	$1 \times \text{Mux}_{3\text{-to-1}}(14\text{-bit})$ $1 \times \text{Mux}_{4\text{-to-1}}(11\text{-bit})$ $1 \times \text{Mux}_{4\text{-to-1}}(16\text{-bit})$	4648	357.0
Proposed	0	0	$1 \times (12\text{-bit})$ $1 \times (14\text{-bit})$	$1 \times \text{Mux}_{3\text{-to-1}}(14\text{-bit})$ $1 \times \text{Mux}_{4\text{-to-1}}(11\text{-bit})$ $1 \times \text{Mux}_{4\text{-to-1}}(16\text{-bit})$	4648	357.0

The numbers of operators and multiplexers used in proposed designs in RCM and their area costs, ‘I’ and ‘II’ estimated based on $0.18\ \mu\text{m}$ and $0.35\ \mu\text{m}$ cell libraries, respectively, are presented in Tables 4.2 to 4.5. The results are compared with those reported in [DEM03a, DEM04, DEM05c] by ReMB methods, [TUM07] by DAG Fusion algorithm, and [CHE09a] by the author’s preliminary ASAP scheduling approach. The bitwidth of the input signal to the RCM is assumed to be 8 bits for all designs. It should be noted that the result of DAG fusion for Set C is different from that of [TUM07] because of an erratum in [TUM07]. The inputs to the top 2-1 multiplexer in Fig. 16(c) of [TUM07] should be multiplied by “16 and 4”, instead of “4 and 2”. This has been verified using the online tool provided by authors of [TUM07]. The results reported in Table 4.4 are based on the corrected design of DAG fusion.

Table 4.3. Area cost estimation for data set B

Algorithm	Adder	Subtractor	Adder/Subtractor	Multiplexers	(I) 0.18 μ m	(II) 0.35 μ m
DAG Fusion [TUM07]	1 $\times$ (19-bit)	1 $\times$ (16-bit)	1 $\times$ (12-bit)	2 $\times$ Mux _{2-to-1} (12-bit) 1 $\times$ Mux _{2-to-1} (19-bit) 1 $\times$ Mux _{3-to-1} (16-bit) 1 $\times$ Mux _{3-to-1} (20-bit)	6365	496.0
ReMB [DEM03a]	1 $\times$ (11-bit) 1 $\times$ (12-bit) 1 $\times$ (17-bit)	0	1 $\times$ (11-bit)	3 $\times$ Mux _{2-to-1} (11-bit) 1 $\times$ Mux _{2-to-1} (14-bit)	5074	386.7
ASAP [CHE09a]	2 $\times$ (17-bit)	0	1 $\times$ (10-bit) 1 $\times$ (17-bit)	1 $\times$ Mux _{2-to-1} (13-bit) 2 $\times$ Mux _{2-to-1} (17-bit)	6240	486.5
Proposed	1 $\times$ (17-bit)	1 $\times$ (11-bit) 1 $\times$ (12-bit) 1 $\times$ (14-bit)	0	1 $\times$ Mux _{2-to-1} (14-bit) 1 $\times$ Mux _{2-to-1} (16-bit) 1 $\times$ Mux _{2-to-1} (17-bit) 1 $\times$ Mux _{3-to-1} (18-bit)	5986	467.3

Table 4.4. Area cost estimation for data set C

Algorithm	Adder	Subtractor	Adder/Subtractor	Multiplexers	(I) 0.18 μ m	(II) 0.35 μ m
DAG Fusion [TUM07]	0	1 $\times$ (16-bit)	1 $\times$ (12-bit)	2 $\times$ Mux _{2-to-1} (12-bit) 1 $\times$ Mux _{3-to-1} (16-bit) 1 $\times$ Mux _{3-to-1} (18-bit)	4476	349.7
ReMB [DEM03a]	0	1 $\times$ (15-bit)	1 $\times$ (12-bit) 1 $\times$ (15-bit)	2 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (15-bit)	4751	355.7
ASAP [CHE09a]	0	0	1 $\times$ (10-bit) 1 $\times$ (14-bit)	1 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (11-bit) 1 $\times$ Mux _{3-to-1} (16-bit)	3612	273.0
Proposed	0	0	1 $\times$ (10-bit) 1 $\times$ (14-bit)	1 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (11-bit) 1 $\times$ Mux _{3-to-1} (16-bit)	3612	273.0

Table 4.5. Area cost estimation for data set D

Algorithm	Adder	Subtractor	Adder/Subtractor	Multiplexers	(I) 0.18 μ m	(II) 0.35 μ m
DAG Fusion [TUM07]	1 $\times$ (10-bit)	0	1 $\times$ (14-bit)	2 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (14-bit)	2994	227.3
ReMB [DEM05c]	1 $\times$ (13-bit)	1 $\times$ (11-bit)	1 $\times$ (13-bit)	1 $\times$ Mux _{2-to-1} (8-bit) 1 $\times$ Mux _{2-to-1} (12-bit)	3530	265.0
ASAP [CHE09a]	0	0	1 $\times$ (10-bit) 1 $\times$ (14-bit)	1 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (14-bit)	2996	224.0
Proposed	0	0	1 $\times$ (10-bit) 1 $\times$ (14-bit)	1 $\times$ Mux _{2-to-1} (10-bit) 1 $\times$ Mux _{2-to-1} (11-bit)	2940	217.0

From Tables 4.2 to 4.5, the proposed RCM design algorithm generates lower cost solutions than the classical ReMB approaches. On average, the proposed algorithm is 19.3% more area efficient than ReMB algorithms, [DEM03a, DEM04, DEM05c], which are originally designed to target on FPGAs. This indicates that the ReMB approaches may not be as efficient on ASIC implementation as they are on FPGA. For ASIC implementation, a more insightful conclusion can be drawn from the comparison with the solutions generated by DAG Fusion [TUM07], which is not tailored to dedicated configurable logic cell structure. On average, the proposed algorithm is still 7% more area efficient than this latest and most competitive method [TUM07]. This saving is mainly contributed by the holistic consideration of opportunity costs in scheduling mobile adders. Merging adders with minimal disparity has successfully reduced the overall bitwidth of the adders and multiplexers. As shown in Tables 4.3 to 4.5, the bitwidths of the arithmetic operators and multiplexers of the proposed designs are always smaller than those of [TUM07]. In addition, the proposed multiplexer splitting also helps. For example, in Table 4.2, instead of having a 16-bit 7-to-1 multiplexer [TUM07], proposed algorithm divides the large fanin multiplexer into multiplexers of lower fanin and bitwidth. The improved scheduling method also outperforms ASAP in datasets B and D.

To corroborate the merits of proposed algorithm, the designs are technology mapped to TSMC 0.18 μ m standard cell library using Synopsys Design Compiler. The synthesized areas and delays are tabulated in Tables 4.6 and 4.7, respectively. Ripple Carry Adders (RCAs) and Carry Look-ahead Adders (CLAs) are used to demonstrate the two different scenarios of using the faster but more costly adders versus using the slower but simpler adders for the RCM design under the same timing constraint. Due to the multiplexer overheads for reconfigurability and the reduced number of adders, the overall area increment caused by the use of CLA over RCA is not as significant.

Table 4.6. Synthesized areas of data sets A-D (in μm^2)

Data sets	Algorithms	RCA	Improvement	CLA	Improvement
A	ReMB	17360.48	62.67%	18108.92	61.00%
	DAGfusion	7477.75	13.35%	8212.88	14.01%
	Proposed	6479.83		7061.95	
B	ReMB	11991.68	-0.72%	12520.57	-2.47%
	DAGfusion	11901.86	-1.48%	12294.37	-4.36%
	Proposed	12078.16		12829.92	
C	ReMB	9749.68	38.76%	9972.55	34.82%
	DAGfusion	7221.61	17.32%	7693.96	15.52%
	Proposed	5970.89		6499.78	
D	ReMB	8086.48	28.14%	8226.19	26.04%
	DAGfusion	6127.23	5.16%	6300.2	3.43%
	Proposed	5811.22		6083.99	

Table 4.7. Synthesized delays of data sets A-D (in ns)

Data sets	Algorithms	RCA	CLA
A	ReMB	9.25	6.06
	DAGfusion	6.65	4.45
	Proposed	6.72	4.6
B	ReMB	9.81	6.06
	DAGfusion	9.15	6.49
	Proposed	8.36	5.89
C	ReMB	9.17	6.1
	DAGfusion	8.15	5.2
	Proposed	7.27	4.99
D	ReMB	7.61	4.69
	DAGfusion	7.95	4.79
	Proposed	7.85	4.78

The area and delay values are combined into an overall area×time (AT) complexity evaluation. The AT product of each design are shown in Table 4.8. The results demonstrate the effectiveness of proposed algorithm in reducing both the critical path delay and the logic area for different RCM designs. On average, this proposed algorithm generates RCM designs of 41.09% and 35.53% lower AT complexities than ReMB with RCA and CLA implementations, respectively. Comparing with DAGfusion, AT products by proposed algorithm are 12.63% and 9.74% lower with RCA and with CLA

implementations, respectively. The AT complexity comparisons are plotted in Fig. 4.11. The reduction of AT product is attributed in part to the CSE using MSD representation and in part to the minimization of average adder disparity in each control step. The balanced adder graph after CSE and the reduction in adder/subtractor and multiplexer bitwidths play a crucial role in shortening the critical path delay.

Table 4.8. AT products of data sets A-D (in $\mu\text{m}^2\times\text{ns}$)

Data sets	Algorithms	RCA	Improvement	CLA	Improvement
A	ReMB	160584.4	72.88%	109740	70.40%
	DAGfusion	49727.04	12.43%	36547.32	11.12%
	Proposed	43544.46		32484.97	
B	ReMB	117638.4	14.17%	75874.65	0.40%
	DAGfusion	108902	7.28%	79790.46	5.29%
	Proposed	100973.4		75568.23	
C	ReMB	89404.57	51.45%	60832.56	46.68%
	DAGfusion	58856.12	26.25%	40008.59	18.93%
	Proposed	43408.37		32433.9	
D	ReMB	61538.11	25.87%	38580.83	24.62%
	DAGfusion	47811.48	4.59%	30177.96	3.63%
	Proposed	45618.08		29081.47

Fig. 4.11. AT complexity comparison of proposed RCM, ReMB methods and [TUM07] with RCA (upper) and CLA (lower) adders for data sets A-D.

Table 4.9. Synthesized Area (A)(μm^2), Delay (T)(ns) and AT product ($\mu\text{m}^2 \times \text{ns}$) of $2 \times \alpha$ multipliers in 8-point/16-point type A Goertzel recursive DCT structures.

DCT attributes	Algorithms	RCA			CLA		
		A	T	AT	A	T	AT
8-point, $w=8$	DAGfusion	10854.0	7.3	79668.7	11446.1	5.8	66731.0
	Proposed	10824.1	7.6	82696.2	11592.5	5.1	58542.1
8-point, $w=12$	DAGfusion	15863.6	10.3	163395.1	16914.7	6.9	116880.9
	Proposed	16349.3	9.3	151394.1	17433.7	6.8	117677.2
8-point, $w=16$	DAGfusion	25985.5	14.8	385369.9	28044.9	8.8	246794.9
	Proposed	24811.1	12.6	313370.7	27163.4	7.3	198835.9
16-point, $w=8$	DAGfusion	13771.3	9.5	130827.3	14463.2	6.8	98349.7
	Proposed	10787.5	7.9	85221.4	11422.9	5.5	62825.7
16-point, $w=12$	DAGfusion	21298.9	11.1	235353.3	22463.2	7.2	161285.6
	Proposed	23830.3	11.6	276908.4	25340.5	7.4	186252.8
16-point, $w=16$	DAGfusion	35239.9	14.6	514502.3	37405.4	9.1	339640.7
	Proposed	34554.6	13.2	454739.1	37804.5	7.9	299033.9

The RCM of the $2 \times \alpha$ multipliers of type A Goertzel recursive DCT structure in [DEM03b] are implemented for two different DCT transform lengths, $N = 8$ and $N = 16$. The coefficients are generated in three levels of precision, $w = 8, 12$ and 16 bits. Every design is implemented with two types of adders, RCA and CLA. The synthesis results are presented in Table 4.9. Since [TUM07] is the latest and most relevant method for ASIC implementation, the AT product is compared with [TUM07] in Fig. 4.12. On average, the proposed algorithm reduces the AT product by 8.51% and 10.60% over DAG fusion method [TUM07] based on RCA and CLA, respectively. The AT products are also evaluated with the constant sets from seven FIR filters in Table 4.10. The results show that the proposed method is able to generate more area-time efficient solutions in general.

Fig. 4.12. AT complexity comparison of proposed RCM and [TUM07] for type A Goertzel recursive DCT with RCA (upper) and CLA (lower) adders, where w is the coefficient wordlength.

Table 4.10. AT products of FIR filter coefficient sets (in $\mu\text{m}^2 \times \text{ns}$)

FIR filter	Algorithms	AT product (RCA)	AT product (CLA)
FIR1 [JAI91]	DAGfusion	26490.7	19084.7
	Proposed	18087.1	12793.4
FIR2 [JAN02]	DAGfusion	17195.4	13336.5
	Proposed	15497.5	11809.4
FIR3 [VIN03]	DAGfusion	19208.2	14389.7
	Proposed	14900.6	11357.7
FIR4 [ZHA88]	DAGfusion	103410.2	80104.6
	Proposed	111724.1	74093.8
FIR5 [LAS92]	DAGfusion	106368.3	76512.9
	Proposed	95732.5	64922.0
FIR6 [VIN03]	DAGfusion	97721.1	65528.4
	Proposed	103662.9	67406.6
FIR7 [LIM83]	DAGfusion	31688.2	21375.8
	Proposed	29929.8	21205.8

The proposed algorithm is further compared against DAGfusion algorithm with the help of the online tool [SPIRAL] using large sets of randomly generated constants. The number of constants in each set, $N = 10$ and 20 , and the wordlength of the constants, $w = 8, 10$ and 12 bits. The average AT products with different N and w are plotted in Fig.4.13.

Fig. 4.13. Comparison of average AT products on randomly generated constant sets, (a) $N=10$ (b) $N=20$.

From Fig. 4.13, the average reduction of area-time cost of proposed algorithm over DAGfusion algorithm is lower (about 4%) for $w \leq 10$ and more significant (up to 25%) for $w = 12$. This is because by scheduling mobile operators based on adders disparity features, mutually exclusive mobile adders are more likely to be scheduled into the same control step and merged by the proposed algorithm. Due to the inclination to merge adders of similar disparity values, the merged adders are more likely to have smaller adder width. In addition, the proposed multiplexer splitting based on multiplexer disparities has also helped to reduce the fan-in of large bitwidth multiplexers.

Besides the solution quality, computational effort is another important criterion for design automation algorithms. The complexity of proposed algorithm is dominated by the scheduling decisions of the mobile adders. The computational complexity of proposed algorithm can be evaluated as follows.

Let $L(v)$ be the wordlength of $a(v)$ for an arbitrary node, $v \in V$ of the DAG, $G(V, E)$. It is well known that MSD reduces the number of nonzero digits of an integer by 33.3% on average from its binary representation [REI60]. Assuming an equal probability of '1' and '0' bits in a binary number, the expected number of signed digits in $a(v)$ is $\hat{h}(v) = (1 - 1/3) \times 1/2 \times L(v) = 1/3 \times L(v)$. The average number of iterations in the computation of $W(v)$ is equal to $\hat{h}(v) - 1$. Hence, the complexity for the aggregate distance computation is $O(|V| \times L(v))$.

Since only one mobile node is scheduled in each iteration, the total number of iterations required to compute the disparity and opportunity cost is equal to the number of mobile nodes. In each iteration, the number of operations required to compute the disparity of a mobile node can be enumerated by the total number of fixed nodes in all its legitimate control steps. In the worst case, every node in the DFG is mobile initially and can be scheduled into any control step. $|V|$ iterations are required. The number of fixed nodes is zero in the first iteration and incremented by one in each iteration until all the $|V|$ mobile nodes become fixed in the last iteration. The number of operations is given by the sum of

an arithmetic progression of $|V|$ terms with a common difference of 1 starting from 0. This sum is equal to $|V|/2 \times (|V| - 1)$. The complexity for the disparity computation is thus $O(|V|^2)$.

From (4.9), the opportunity cost is computed by summing the product of the disparity and probability displacement over all legitimate control steps. Each iteration requires at most t_{max} multiplications and $t_{max}-1$ additions, where t_{max} is the number of control steps. Let H be the maximum hamming weight of all constants. Then, $t_{max} = \lceil \log_2(H) \rceil$ if assuming that $t_{max} = D(G)$. The number of operations required for the computation of opportunity cost is given by $|V| \times (2^{\lceil \log_2(H) \rceil} - 1)$. It should be noted that the hamming weights, H of the constants in MSD representation is usually less than 10, The complexity of opportunity cost computation is $O(|V|)$.

A small fraction of nondeterministic computations due to the opportunity costs of predecessor and/or successor nodes and the update of mobilities can be omitted since the aggregate distance and disparity computations of all nodes required for their evaluation have been conservatively included in the above analysis. The total computational complexity of proposed algorithm is $O(|V| \times L(v) + |V|^2 + |V|)$. In practice, $L(v)$ is finite and can be safely assumed to be lower than 20. Hence, the computational complexity of proposed algorithm is $O(|V|^2)$.

This complexity is far less than the brute force approach of DAG Fusion which searches exhaustively for all possible pairs of DAGs for the best fusion. The runtime for fusing N DAGs is given by $O(\text{Num_iterations} \times N \times \text{Runtime}(\text{FusePairDAGs}))$, where $\text{Runtime}(\text{FusePairDAGs}) = O\left(\frac{n!(mN+n)}{\lceil \log_2(m+1) \rceil! (n-m)!}\right)$ and Num_iterations is the number of different orderings of fusing a pair of DAGs, with n and m being the numbers of adders in the first and second DAGs, respectively [TUM07]. The first DAG is obtained iteratively by the previous fusion of N DAGs, thus n is comparable to m and in the worst case, $m = n$. The maximum Num_iterations for N DAGs is $N!$ [TUM07]. Limiting Num_iterations to reduce the runtime will compromise the solution quality. The overall

complexity can be approximated to $O(N^2 N! n!)$. This is much higher than proposed algorithm overall runtime complexity of $O(N^2 n^2)$ if the $|V|$ adders are amortized over the N constants so that the average number of adders contributed by each DAG is n .

4.5 Application of RCM in Circular Convolution

In previous section, a new design methodology for area-time efficient reconfigurable constant multiplier (RCM) has been proposed. In this section, two new circular convolution architectures are presented with the help of this new design methodology.

Circular convolution is one of the important and commonly used transformations in many DSP applications, such as image processing, audio signal processing, artificial reverberation, etc. [PRO96, CHE05, MEH06]. A circular convolution of two functions, for instance, is often expressed in terms of the periodic extension of one or both functions. By periodic extension, it means that a new function is formed by shifting the original function by some multiples of a period, T , and adding all the copies together. Therefore, the hardware cost for a circular convolution is dominated by the number and complexity of the multipliers and adders. These parallel multipliers can always be modeled as a MCM block. Design automation algorithms for MCM problem discussed in Chapter 3 can be used to reduce the redundancies of the MCM adders, such as the use of RAG-n algorithm [DEM95] for the implementation of circular convolution in [MEY04]. However, it can be noted that the multiplier block in circular convolution can also be substituted by a more area efficient RCM block.

4.5.1 Present Architecture for Circular Convolution

The output of a linear time invariant system can be defined as a convolution of two discrete time signals, the system's input, $x[n]$, and the system's response, $h[n]$.

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} (x[k]h[n-k]) \quad (4.11)$$

When given two discrete-time sequences with finite length, they cannot be simply multiplied together as in the above linear convolution formula as n can fall beyond the sequence length N . However, if the sequences are extended periodically, they will loop back and still have nonzero values for $n \geq N$. The convolution defined in this way is called the circular convolution. It is also known as cyclic or periodic convolution [ROM06]. Mathematically,

$$y[n] = \sum_{m=1}^N \left(x[m] h[(n-m)_N] \right) \quad (4.12)$$

Thus, the circular convolution is normally expressed as:

$$y[n] \equiv (x[n] * h[n]) \quad (4.13)$$

From the definition of (4.12), circular convolution can be transformed into a matrix multiplication as

$$\begin{bmatrix} y[1] \\ y[2] \\ y[3] \\ \vdots \\ y[N] \end{bmatrix} = \begin{bmatrix} h_N & h_{N-1} & h_{N-2} & \cdot & \cdot & \cdot & h_1 \\ h_1 & \cdot & \cdot & \cdot & \cdot & \cdot & h_2 \\ h_2 & \cdot & \cdot & \cdot & \cdot & \cdot & h_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdot & \cdot & \cdot & h_N \end{bmatrix} \times \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ \vdots \\ x[N] \end{bmatrix} \quad (4.14)$$

$$y[j] = \sum_{i=1}^N h_{N-i+j} \times x[i] \quad (4.15)$$

where $j \in [1, N]$

From (4.14), it is clear that an N -point circular convolution on two discrete sequences needs $N \times N$ multipliers. However, it can be noted that the $N \times N$ multiplicand matrix, h , involves only N instead of $N \times N$ unique constants, which are cyclically rotated and multiplied with different samples of the input sequence $x[n]$. An efficient architecture for circular convolution is presented in [MEY04] by using N multipliers only, as shown in Fig. 4.14. The values of the multipliers are the N elements in h . The input sequence, x is firstly loaded into the registers. After one cycle of multiplication, the input sequence rotates and each input sample is moved to the next position and multiplied with the next

element of h . Using this rotating sequence, every input sample is multiplied with every member of h by using only N multipliers.

Fig. 4.14. Conventional architecture of circular convolution

4.5.2 Proposed Architectures for Circular Convolution

In Fig. 4.14, conventional architecture requires N multipliers, N cyclic shift registers and N adders. Although a fully pipelined Distributed Arithmetic or transposed filter structure using the most area-efficient RAG-n algorithm [DEM95] can be used to substitute the multiplier block for lower area cost, there is still redundant resource among the logic operators which are not in use at all time.

Equation (4.14) can be decomposed into:

$$\begin{aligned}
 \begin{bmatrix} y[1] \\ y[2] \\ y[3] \\ \vdots \\ y[N] \end{bmatrix} &= \begin{bmatrix} h_N & h_{N-1} & h_{N-2} & \cdot & \cdot & \cdot & h_1 \\ h_1 & \cdot & \cdot & \cdot & \cdot & \cdot & h_2 \\ h_2 & \cdot & \cdot & \cdot & \cdot & \cdot & h_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdot & \cdot & \cdot & h_N \end{bmatrix} \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ \cdot \\ \cdot \\ x[N] \end{bmatrix} \times \begin{bmatrix} h_N \times x[1] \\ h_1 \times x[1] \\ h_2 \times x[1] \\ \cdot \\ \cdot \\ h_{N-1} \times x[1] \end{bmatrix} + \begin{bmatrix} h_{N-1} \times x[2] \\ h_N \times x[2] \\ h_1 \times x[2] \\ \cdot \\ \cdot \\ h_{N-2} \times x[2] \end{bmatrix} + \dots + \begin{bmatrix} h_1 \times x[N] \\ h_2 \times x[N] \\ h_3 \times x[N] \\ \cdot \\ \cdot \\ h_N \times x[N] \end{bmatrix} \\
 &= M_1 + M_2 + \dots + M_N
 \end{aligned}
 \tag{4.16}$$

$$\text{where } M_i = \begin{bmatrix} h_{N+1-i} \times x[i] \\ h_{N+2-i} \times x[i] \\ h_{N+3-i} \times x[i] \\ \vdots \\ h_{N+N-i} \times x[i] \end{bmatrix}$$

Every output of (4.16) is a sum of N products obtained from different input samples and different constants of h . As such, each M array can be implemented by using one single RCM. It is noticed that the N constants in different M arrays are actually the same integer value shifted by different amounts. Therefore, the circular convolution can be implemented by an architecture with only a single RCM. However, it should be noted that replacing the N multipliers by one RCM unconditionally could possibly affect the throughput rate of the circular convolution system. This delay penalty requires careful implementation of RCM to fulfill the specific delay constraints of the total system. The size N of the circular convolution is also important as it dictates the total time to reconfigure and generate N different multipliers. In what follows, two schemes of circular convolution targets at applications with different delay constraints are proposed.

a. Type A Architecture

Type A architecture is suitable for circular convolution with low throughput rate. In this architecture, each product of (4.15) is obtained by multiplying one input sample with one corresponding constant of h . After summing up the first N products to obtain the first output, the h sequence is circularly shifted by one register position and the same multiply-accumulate process is executed again. Here, RCM is adopted to generate the desired constant multiplication of the input sample x with the h sequence. The proposed architecture is shown in Fig. 4.15.

Fig. 4.15. Proposed Type A architecture for circular convolution.

N iterations are required to generate a complete output sequence. Each iteration generates N products and they are summed up to obtain one output value. As shown in Fig. 4.15, switch S_1 is initially set to the 'load' state to latch the N input samples into N registers. After this, S_1 is set to 'run' so that the RCM is reconfigured to enable the multiplication of the input sample $x[1]$ and h_N in the first clock cycle. The tristate output buffer is disabled by S_2 while the product is stored in the accumulator memory. In the next clock cycle, $x[1]$ is circularly shifted back into the last register. The RCM is reconfigured to produce the product of h_{N-1} and the current input sample, $x[2]$. This second product is accumulated with the previously stored product and the sum is stored in the accumulator memory. This iteration continues until all the N products are accumulated in the memory.

The output buffer is then enabled by S2 to transfer the value stored in the memory, $y[1]$ to the output. The memory is cleared before next iteration commences. At this juncture, $x[1]$ has been shifted back to become the first input sample for the second iteration. The RCM is reconfigured to produce the product of $x[1]$ and the appropriate constant according to (4.15). After N clock cycle, $y[2]$ is produced in the accumulator memory and buffered to the output. This process repeats for N iterations to generate the complete output sequence. Unlike the conventional architecture, only one single RCM block is needed in the proposed architecture, which provides significant cost saving especially for large problem size. However, to complete an N point circular convolution, N iterations, each of N clock cycles, are required. The total delay to process N input samples is hence N^2 clock cycles for Type A architecture as opposed to only N cycles for the conventional architecture. If f is the input sampling frequency, then the total delay is N^2/f . The throughput rate of Type A architecture is therefore limited by $\frac{N}{(N^2/f)} = f/N$. This delay penalty causes the proposed Type A architecture to be applicable to systems that can afford to have a slower throughput rate.

There is also an upper bound on the input sample frequency f for Type A architecture. If the critical path delay of the RCM is t_1 , from Fig. 4.12, t_1 must be shorter than $1/f$ so that the RCM can complete the current multiplication and be reconfigured before the next input sample is shifted in. From the simulation results of Section 4.4, it is noticed that the typical critical path delay of a RCM is in the order of 10 ns for 16-bit constants. Hence, $1/f \geq 10\text{ns}$ and the maximum input sampling frequency of Type A architecture is around 100MHz.

b. Type B Architecture

Although Type A architecture is efficient in area cost, the time needed to run the N iterations is a severe drawback. From (4.16), the outputs of the convolution are equal to the sum of N arrays, M_1 to M_N , each of dimension $N \times 1$. The same input sample is applied simultaneously to all the entries of an array but different input samples are applied to

different arrays at any point of time. Hence, in type B architecture, the multiplications of the same input sample and the sequence of h constants in an M array of the RCM can be completed in one clock cycle. Unlike the Type A architecture, this input sample does not loop back at each clock cycle but is held constant until all constant multiplications have been completed. The Type B architecture is shown in Fig. 4.16.

Fig. 4.16. Proposed Type B architecture for circular convolution.

Only one iteration is needed and the total delay is N clock periods for Type B architecture as opposed to N^2 clock periods for Type A architecture. When $x[1]$ is shifted in initially, the RCM is reconfigured N times to generate a sequence of products of the constants h_i ($i = 1, 2, \dots, N$) and $x[1]$. These N products are demultiplexed and stored into N output registers, R_1 to R_N according to (4.16). In the next clock cycle when $x[2]$ is shifted in, the

RCM is reconfigured in the same manner to produce h constant multiplications with the variable, $x[2]$. These N products are N -way demultiplexed and accumulated with the previously stored products in the corresponding output registers. This process continues until the last input sample is sampled into the RCM and a complete output sequence of $y[1]$ to $y[N]$ is obtained. Compared with Type A architecture, the number of output registers and accumulators has been increased from 1 to N and an additional N -way demultiplexer is employed. Although the area cost is higher than that of Type A architecture, the total delay has been reduced significantly to only N clock cycles. As the period of each clock cycle is $1/f$, the total delay is N/f and the throughput rate of Type B architecture is therefore limited by $\frac{N}{(N/f)} = f$. This throughput rate is N times faster than that of Type A architecture.

On the other hand, the input of Type A architecture has to be sampled at a slower rate than Type B architecture. As every input sample will be multiplied with all the N constants in one clock period in Type B architecture, the minimum clock period for the input sampling becomes $N \times t_1$. With the same assumption that the RCM block has a delay of $t_1 = 10$ ns, the input sampling frequency, f has to be less than $100/N$ MHz.

4.5.3 Experimental Results and Discussions

This section presents the experimental results of the proposed architectures and compares them with those of the conventional architecture for the circular convolution. The cost of the accumulator and output registers employed in the proposed architectures are assumed to have similar area cost to the $N-1$ accumulators and latches in the conventional architecture. To evaluate the area saved by using the RCM block for circular convolution, the area cost due to the use of MCM block in conventional architecture is compared with the RCM block for Type A architecture and the RCM block cum N -way demultiplexer for Type B architecture. In view of the comparable logic structure of an N -way multiplexer and an N -way demultiplexer, they are assumed to have the same area cost.

By comparing (4.12) with (2.1), it is noticed that the transfer functions of circular convolution and the direct form FIR filter are very similar. Therefore, we can use the commonly cited FIR filter coefficient sets for the circular convolution operations to compare the performances of different implementations of the same circular convolution operation. For the conventional implementation, the most well known area-efficient RAG-n [DEM95] algorithm is used to design the constant coefficient multiplier block. Similar to Section 4.4, the same area cost estimation function of [TUM07] is adopted for the comparison and the input wordlength is assumed to be 12-bit. Table 4.11 lists the area costs of the proposed architectures and the conventional architecture implemented with RAG-n multiplier block.

Table 4.11. Comparison of estimated area costs based on FIR filter coefficient sets

Filters	Circular convoluter with RAG-n multiplier block	Type A	Type B
FIR1 [ZHA88]	23430	12220	17512
FIR2 [LAS92]	39547	12034	21296
FIR3 [VIN03]	34506	12536	18269
FIR4 [DEM02a]	22081	15273	19277
FIR5 [SAM89]	5112	3580	6044

Four other data sets from Table 4.5 are also used for the circular convolution. The area cost for the proposed architectures and the RAG-n multiplier block are compared in Table 4.12.

Table 4.12. Comparison of estimated area costs based on reconfigurable constant multiplier data sets A-D

Data set	Circular convolver with RAG-n multiplier block	Type A	Type B
A	5325	4648	6664
B	7029	5986	6742
C	4757	3612	4284
D	5680	2940	3556

From the above analysis, it is concluded that the proposed architectures are more area efficient than the conventional circular convolution architecture. Even though conventional architecture is implemented by the most area efficient RAG-n algorithm, the average areas saved by the proposed architectures are still very significant. From Tables 4.11 and 4.12, it is obvious that the area cost of Type B architecture is always higher than that of Type A architecture because of the additional N -way demultiplexer. This offset to the area saved by Type B architecture is dependent on the fanout and bitwidth of the demultiplexer. Therefore, we can conclude that Type B is more suited to applications with high throughput rate. This architecture provides around 15.49% area improvement over conventional architecture. Type A architecture, on the other hand, is more suitable for low throughput rate applications. It provides a more significant area reduction of around 38.0% over the conventional architecture.

4.6 Summary

For many general DSP problems, the need for multiple constant multiplications is mandatory. This chapter presented a new insight into the design of reconfigurable constant multipliers where the adder resources can be time multiplexed to reduce the hardware cost. Comparing with existing design methodologies which are mostly based on the reduced adder graph or matured techniques in the design of the multiplier block of digital filters, the concept conceived in this chapter is new and unique. It is the first proposal that delineates the RCM architecture as a DFG and solves this design problem by an efficient heuristic scheduling algorithm. A disparity measure is defined to assess the cost of merging two or more temporally correlated adders by evaluating the compactness of signed digit distribution among mutually exclusive nodes in the same control step. This is used in conjunction with their probability displacement to determine the opportunity cost of scheduling a mobile adder into a control step, from which an optimized scheduling decision is made. Since the depth of the adder tree is first minimized by a global CSE, the RCM resulting from the scheduled DFG is minimized in the $\text{area} \times \text{delay}$ sense. Synthesis results on some practical RCMs and randomly generated sets of constants show that the proposed algorithm produces solutions which are up to

25% more area-time efficient than the latest and best reported solutions known in the literature for 12-bit constant sets. Two new circular convolution architectures by adopting this proposed RCM design methodology are also presented and discussed.

In the design of MCM block, due to the lack of an efficient model for active power consumption estimation, the switching power is often assumed to be directly correlated with the logic depth and/or logic complexity. Therefore, the merits of different algorithms are evaluated and compared quantitatively based on the number of logic operators and logic depth of the solutions they generated as were done in Chapters 3 and 4. Power consumption is an important design parameter nowadays and there is a need for a better way of benchmarking the power consumption of the solutions generated by different design methodologies. In the next chapter, we will propose a newly dynamic power estimation model for the MCM block.

Chapter 5

New Power Index Model for Switching Power Analysis

5.1 Introduction

As discussed in Chapter 2, many algorithmic approaches [POT96, DEM95, HAR96, CHA08, PEI02, PAS99, DEM02a, DEM04, TUM07] have been developed to automate the design of low complexity MCM or RCM. In Chapter 3 and 4, new area-time efficient design methodologies for Multiple Constant Multiplication (MCM) and Reconfigurable Constant Multiplier (RCM) have been proposed. These algorithms are mainly driven by the adder cost and focus on reducing the adder cost and logic depth. Although high throughput and low area are still a popular pursuit, power consumption has become an as important, if not more important criterion in ASIC design due to the rapid proliferation of battery powered devices into the consumer electronic market. Reducing the logic complexity and logic depth, to some extent, will minimize the power consumption. However, the extent and causality to their correlation have not been satisfactorily quantified. There exist examples of lower adder cost solutions that consume more power than the higher adder cost solutions, which show that the weakly correlated metrics of adder cost and adder depth is inadequate to evaluate quality the of design automation algorithm in terms of power performance, let alone the formulation of power driven optimization algorithms for FIR filter design. There is, therefore, a pressing demand to set up a good power dissipation model for this generic class of MCM problems. In this chapter, a new high-level measure for modeling the dynamic power dissipation of reduced adder graph of FIR filter is presented. This new Power Index (PI) model is established based on the input correlated switching activities of FA, the propagation of

FA transitions and the varying load capacitances due to the sharing of adders in a reduced adder graph. Dynamic power simulations of ASIC implementation of benchmark filters indicate that the proposed PI measure is well correlated to the actual dynamic power dissipation and more accurate than the existing Glitch-Path Count (GPC) and Glitch-Path Score (GPS) measures.

This chapter is organized as follows. Section 5.2 provides an overview and discussion of existing MCM block power estimation models. In section 5.3, the derivation and proposition of the new power index model are introduced. One illustration example to show the calculation by proposed PI model is given in section 5.4. Power synthesis results on a group of benchmark FIR filters are presented to evaluate the correlation of proposed model in section 5.5. Section 5.6 is going to summarize this chapter.

5.2 Problems with Existing Dynamic Power Estimation Models

Gate level and transistor level power estimation model are too compute intensive for MCM problems. This issue was noted by some researchers and alternative coarse grain metrics have been proposed. In [DEM00], a unitless cost metric called GPC is proposed to indicate the relative switching power consumption of MCM blocks represented in the form of primitive operator graphs [DEM95]. This method relies on two major contributing factors to the switching activities of MCM blocks, which are the discrete adder cost and the logic depth. The argument is that the carry chain in the adders will generate invalid results before the adders are settled to the correct value. These spurious transitions will inevitably increase the power consumed by the adders. In addition, the adders deep in the delay path would also inherit the spurious output transitions from the preceding adders, which cause more transitions at their inputs [DEM00]. Hence, it is conjecture that the total number of adders and logic depth have a direct correlation with the dynamic power dissipation.

Based on these argument, the authors of [DEM00] suggested that for a particular node i in the MCM block, the GPC generated by it is equal to the sum of the GPCs of its two input nodes plus one, i.e.,

$$GPC_i = GPC_{i_left} + GPC_{i_right} + 1 \quad (5.1)$$

where GPC_{i_left} and GPC_{i_right} are the GPC of the left and right inputs of node i , respectively.

In (5.1), GPC_{i_left} and GPC_{i_right} represent the amounts of switchings inherited from the predecessor nodes and the addition of 1 is due to the switching occurred in the node itself. Therefore, the total GPC of the MCM block that has N logic operators in it is given by:

$$GPC_{MCM} = \sum_{i=1}^N GPC_i \quad (5.2)$$

GPC provides a coarse indicator on the extent of node switching activities based on simplistic assumption. It has ignored many variables that will also influence the switching power such as the probabilities of input switching, the word-length of the operator and the fan-outs. As the result, the maximum estimation error of GPC can be as high as 60% as mentioned by the authors in another publication [DEM02b]. The GPC model is obviously too rudimentary and insufficient for comparing the relative power consumption of different MCM block solutions.

To improve the GPC method, the authors of [DEM00] proposed another power estimation method called GPS [DEM02b]. GPS is more accurate and elegant than GPC by taking the wordlengths of the operators into consideration. GPS of an operator, i in the MCM block can be recursively defined as:

$$GPS_i = r(k^\alpha) + 0.67(GPS_{i_left} + GPS_{i_right}) \quad (5.3)$$

The variables in (5.3) are computed using the following steps [DEM02b]:

Step 1: Compute r , the effective bitwidth of operator i .

The length of an operator, i is given by $n = \text{floor}(\log_2(x)+1) + \text{input word length}$, where x is the partial product produced by the operator, i . The floor function rounds the number to the nearest integer towards minus infinity. Knowing that some less significant bits can be directly hardwired, the effective adder length r is calculated by $r = n - e$, where e is the difference in the shift amounts of the two inputs. If the operation is a subtraction, the adder length is given by $r = n$.

Step 2: Scaling factor of GPS inherited from previous adders.

For logic-depth values greater than 1, the GPS coming from the previous adders should be considered as the sum of the GPS for the carry and sum outputs. Hence, an empirically derived scaling factor of 0.67 is applied to the GPS that affecting the next level adder. This factor is derived from the experiments [DEM02b].

Step 3: Relate the GPS with the logic depth of node, i .

Another fact that has been observed from the real timing data is that the transitions generated by each adder will increase as the logic depth increases. Hence, another empirically derived coefficient (k^α) is applied where k is the logic depth and $\alpha = 0.55$ is a constant.

The total GPS of the entire N -adder MCM block is obtained by summing up the GPS calculated for each node by (5.3), and is given by:

$$GPS_{MCM} = \sum_{i=1}^N GPS_i \quad (5.4)$$

From simulation results of [DEM02b], GPS shows reasonably better accuracy than GPC. However, a maximum estimation error of 20% still exists. The main problem with GPC and GPS models is that these measures are based on the number of transitions. The experimental results in [DEM00] and [DEM02b] that used to validate the models are also based on the number of simulated transitions instead of the actual dynamic power consumed in the FIR filters. The total number of transitions is but only a part of the

dynamic power dissipation. Hence, a more accurate dynamic power estimation model, which addresses several other key contributing factors of dynamic power of MCM at the operator or full adder level, is highly desired.

5.3 CMOS Switching Power and Proposed PI Model

The total power consumption of CMOS integrated circuits is due to four main contributing factors, namely capacitive power dissipation, short circuit power dissipation, standby power dissipation and leakage power dissipation [PAR99]. The first two factors are collectively referred to as dynamic power and the latter two factors are known as static power. The major source of dynamic power dissipation is the capacitive power or switching power dissipation due to transitions at the circuit nodes [CHA95, PAR99, NOL92]. Switching power is dissipated by charging and discharging the load capacitances when the transistors turn on or off. For the current CMOS technology and the synthesis results reported in this thesis, switching power remains the dominant factors of the total power consumption. Therefore, the proposed model will focus only on the estimation of the switching power consumption of the MCM block of FIR filters.

5.3.1 Formulation of Switching Power

The logic gate is realized with a pull-up network and a push-down network of transistors in classical CMOS implementation. The transistors in the pull-up or push-down network are either connected in series, parallel or a combination of both. Depending on the inputs, only the pull-up or push-down network will turn on, causing the output capacitance of the logic gate to be charged from the supply voltage or discharged to ground. Except for some rare instances, where some transistors in the pull-up network and the push-down network are both turned on simultaneously, a short-circuit path is created momentarily through these transistors and a large current will be flown between the power supply and ground, causing the short-circuit power dissipation. An inverter is the simplest and most fundamental logic structure of CMOS integrated circuit for the study of the switching power, where one cycle of switching at the input of the circuit involves a rising and a

falling outputs. A total charge of $Q = C_L V_{DD}$ is dissipated when the output node is charging or discharging, where C_L is the load capacitance and V_{DD} is the supply voltage. A CMOS inverter is shown in Fig. 5.1, where $i_{DD}(t)$ and f_{sw} are the instantaneous current and the input switching frequency, respectively.

Fig. 5.1 A CMOS inverter

By defining the node transition activity factor or simply switching activity, α as the average number of times the node makes a transition in one clock period, the switching power can be computed as [CHA95]:

$$P_{switching} = \frac{1}{2} \alpha C_L V_{DD}^2 f_{clk} \quad (5.5)$$

5.3.2 Switching Activity of Ripple Carry Adder

To analyze and estimate the switching power of MCM block, switching activities of adders with varying bit widths in the reduced adder graph of fixed coefficient FIR filter have to be calculated first. The output switching probabilities of a fundamental bit slice that make up a discrete adder should be considered. The switching probabilities of the sum (S) and carry (C) outputs of a full adder (FA) are modeled in [JOH04, JOH08] as follows:

$$\alpha(C_i) = 0.5 \alpha(X_i) \alpha(Y_i) \quad (5.6)$$

$$\alpha(S_i) = 0.5 (\alpha(C_{i-1}) + \alpha(S_{i-1}) + \alpha(X_i) + \alpha(Y_i)) - 2\alpha(C_i) \quad (5.7)$$

$$\alpha(CS_i) = \alpha(S_i) \quad (5.8)$$

where X_i and Y_i are i -th bits of the operands. $\alpha(S_i)$ and $\alpha(C_i)$ are the probabilities that a transition occurs exclusively in the sum and the carry outputs, respectively. $\alpha(CS_i)$ is the probability that a transition occurs in both the carry and sum outputs.

The switching activities of the carry and sum outputs of a FA are calculated in [JOH04] as follows:

$$SW(C_i) = \alpha(C_i) + \alpha(CS_i) \quad (5.9)$$

$$SW(S_i) = \alpha(S_i) + \alpha(CS_i) \quad (5.10)$$

This method presented in [JOH04] has been adopted in [JOH08] to estimate the switching activity of single adder multiplier and applied to the case where a signal is added to a shifted version of itself.

The propagation of switching activities in the sum and carry chains differs from adders to adders in an adder tree and is dependent on the structure of the adder. For simplicity, the Ripple Carry Adder (RCA) is used. The switching activity of an M -bit RCA can be expressed as:

$$\begin{aligned} \alpha &= \sum_{i=1}^M SW(C_i) + \sum_{i=1}^M SW(S_i) \\ &= \sum_{i=1}^M (\alpha(C_i) + \alpha(CS_i)) + \sum_{i=1}^M (\alpha(S_i) + \alpha(CS_i)) \end{aligned} \quad (5.11)$$

As the carry input to the FA in the least significant bit of every RCA is always 0, $\alpha(C_0) = 0$. It is also assumed that the initial inputs, X_1 and Y_1 , to the FA at these bit positions are random and uncorrelated, i.e., $\alpha(X_1) = \alpha(Y_1) = 0.5$. Based on these assumptions and Equations (5.6) to (5.8), the switching activities of those RCAs in the adder graph that receive the primary inputs or their shifted version can be easily computed. Other RCAs that have at least one of their inputs generated from the outputs of another RCA need to be separately analyzed. This will be discussed in the next section.

5.3.3 Proposed Power Index (PI) Model

In the proposed model, two types of switching activity propagation between two FAs in an adder graph are analyzed. The first type concerns the propagation of switching activities from the carry output of one FA into an input of another FA in the same RCA. This type of switching activity propagation is called the intra-RCA propagation. The second type of switching activity propagation refers to the propagation of switching activities from the sum output of a FA in a RCA to the input of a FA in another RCA. This is called the inter-RCA propagation. These two types of transition propagation are shown in Fig. 5.2. Given the adder graph of the solution to a MCM problem, the intra-RCA and inter-RCA propagations can be unambiguously identified.

Fig. 5.2. Inter-RCA and intra-RCA propagation

Another important factor that has been omitted in GPC and GPS is the variation of load capacitance due to common subexpression sharing in the minimization of MCM problem. The effect of load capacitance is modeled by considering the fan-out of each adder in the adder graph. As RCA is formed by cascaded FAs, the fan-out of each FA that makes an intra-RCA propagation is always one. For inter-RCA propagation, the fan-out of each FA involved is equal to the total number of RCAs driven by it. The total load capacitance of a FA in the j -th RCA of the adder graph can be expressed as:

$$C_L(j) = \sum_{i=1}^w C_{FA_i} \quad (5.12)$$

where w is the fan-out of the FA of interest and C_{FA_i} is the input capacitance of the i -th FA driven by it.

From (5.5), $P_{switching} \propto (\alpha \times C_L)$ for a fixed supply voltage and a given clock frequency. From (5.12), it can be rewritten to $P_{switching} \propto \left(\alpha \times \sum_{i=1}^w C_{FA_i} \right)$. If the input capacitances of every FA are assumed to be identical, it becomes $P_{switching} \propto (\alpha \times w)$. Therefore, the total switching power of a N RCA adder graph can be expressed as

$$P_{switching_MCM} \propto \sum_{j=1}^N (\alpha_j \times w_j) \quad (5.13)$$

where N is the total number of RCAs in the adder graph and α_j and w_j are the switching activity and fan-out of the j th RCA respectively.

By ignoring the constant of proportionality, a simple first order estimation of the switching power, called the power index (PI), is defined as follows:

$$PI_{MCM} = \sum_{j=1}^N (\alpha_j \times w_j) \quad (5.14)$$

By substituting (5.11) into (5.14), it becomes

$$\begin{aligned} PI_{MCM} &= \sum_{j=1}^N \left(\sum_{i=1}^{M_j} (\alpha(C_i) + \alpha(CS_i)) \times w_j + \sum_{i=1}^{M_j} (\alpha(S_i) + \alpha(CS_i)) \times w_j \right) \\ &= \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(C_i) + \alpha(CS_i)) \times w_j + \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(S_i) + \alpha(CS_i)) \times w_j \end{aligned} \quad (5.15)$$

where M_j denotes the bit width of the j -th RCA in the adder graph.

For intra-RCA switching activity propagation, the fan-out due to the carry output of every FA is 1, so (5.15) can be simplified to

$$PI_{MCM} = \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(C_i) + \alpha(CS_i)) + \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(S_i) + \alpha(CS_i)) \times w_j \quad (5.16)$$

PI can be decomposed into two parts.

$$PI_{MCM} = PI_{intra-RCA} + PI_{inter-RCA} \quad (5.17)$$

where $PI_{intra-RCA} = \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(C_i) + \alpha(CS_i))$ is due to the intra-RCA switching propagation through the carry signal and has implication on logic operator complexity.

$PI_{inter-RCA} = \sum_{j=1}^N \sum_{i=1}^{M_j} (\alpha(S_i) + \alpha(CS_i)) \times w_j$ is due to the inter-RCA switching propagation through the sum signal and has implication on logic depth.

This power index can be computed directly to the adder graph of a FIR filter to analyze its switching power. An adder graph is a unified representation of the solution to an MCM problem where each node represents a discrete adder, and in this case, a RCA. To apply the power index model, each node is assigned a unique positive integer identifier and is annotated with four parameters and a switching activity matrix to keep track of the propagation of switching activities of each FA, as shown in Fig. 5.3.

Fig. 5.3. Adder graph annotation and switching activity matrices

The four parameters are: t_l , t_r , s_l and s_r . The parameters, t_l and t_r of a node are the identifiers of its left and right predecessor nodes, respectively. Since each RCA has two operands, each of which can either be fed from the output of a RCA or from the primary

input. The primary input is assigned an identifier of 0 to distinguish it from other node identifiers. The parameters s_l and s_r of a node are the amount of shifts applied to the outputs of its left and right predecessor nodes, respectively. The switching matrix associated with each node v_j stores the switching activities, $SW(C_i)$ and $SW(S_i)$ (see Equations (5.9) and (5.10)) of its M_j FAs. The power index for any given MCM architecture can be computed as follows:

Step 1: Map the architecture into a unified adder graph representation and annotate each node of the graph with the four parameters.

Step 2: Do a breadth-first traversal to each node. Read the parameters of the visited node and apply Equations (5.6) to (5.10) to compute the sum and carry switching activities of every FA. Store the results in its switching activity matrix.

Step 3: Use (5.20) and the values in the switching activity matrix of node v_j to compute the PI of the j -th RCA.

Step 4: Repeat Steps 2 to 4 until the PIs of all nodes have been computed. For efficient computation, the PI of each node is computed in a breadth-first manner in Step 2 so that the switching activities of the predecessor nodes have been computed before they are used for the computation of the power index in the current node.

5.4 An Illustrative Example

The computation of the power index is illustrated by an example of a 7-tap transposed direct form FIR filter shown in Fig. 5.4. It is generated by the MBPG algorithm presented in Chapter 3. The wordlength of the input signal is assumed to be 8 bits in this example.

Fig. 5.4. Illustrative example of PI calculation on a 7-tap FIR filter

Before computing the PI, the adder graph of the MCM block and the switching activity matrices need to be constructed. The adder graph for this MCM block is shown in Fig. 5.5. There are a total of six RCAs indicated by the nodes labeled as v_1 to v_6 . The nodes, v_1 to v_4 , are situated at adder depth one while the nodes v_5 and v_6 are situated at adder depth two. According to the definitions given in Section 5.3, the four parameters, t_l , t_r , s_l and s_r , annotated on each node can be derived from Fig. 5.4. Based on the primary input wordlength of 8 bits and the amounts of input shifts to the RCAs, the bitwidths of all RCAs can be computed to be $M_1 = 10$, $M_2 = 11$, $M_3 = 10$, $M_4 = 12$, $M_5 = 16$ and $M_6 = 16$.

Fig. 5.5. Adder graph annotation for illustrative example

According to (5.15), to compute the total PI of the MCM block, the PIs of all the RCAs in the adder graph of the MCM block need to be calculated and summed up. In this example, the PI of v_1 , which represents the partial sum of 3, will be calculated first. Both inputs of v_1 are fed directly from the input to the filter. Hence $\alpha(C_0) = 0$ and $\alpha(X_1) = \alpha(Y_1) = 0.5$. By (5.6) and (5.7), $\alpha(C_1) = 0.5 \times 0.5 \times 0.5 = 0.125$ and $\alpha(S_1) = 0.5 \times (0 + 0 + 0.5 + 0.5) - 2 \times 0.125 = 0.25$. By (5.12), the probability of switching for the carry and sum outputs is given by $\alpha(CS_1) = \alpha(S_1) = 0.25$. By (5.9) and (5.10), the switching activities for the carry and sum outputs of the full adder at the least significant bit are $SW(C_1) = 0.125 + 0.25 = 0.375$ and $SW(S_1) = 0.25 + 0.25 = 0.5$, respectively. Similarly, for next full adder at the next bit position, $\alpha(C_2) = 0.5 \times 0.5 \times 0.5 = 0.125$, $\alpha(S_2) = 0.5 \times (0.125 + 0.25 + 0.5 + 0.5) - 2 \times 0.125 = 0.4375$ and $\alpha(CS_2) = \alpha(S_2) = 0.4375$. The switching activities for this full adder are $SW(C_2) = 0.125 + 0.4375 = 0.5625$ and $SW(S_2) = 0.4375 + 0.4375 = 0.875$, respectively. The same calculation is repeated for the remaining 8 full adders and the switching activity matrix of v_1 can be computed to be:

i	1	2	3	4	5	6	7	8	9	10
$SW(C_i)$	0.375	0.563	0.656	0.703	0.727	0.738	0.744	0.747	0.749	0.749
$SW(S_i)$	0.5	0.875	1.063	1.156	1.203	1.227	1.238	1.244	1.247	1.248

By (5.16) and the values in the above switching activity matrix, the total switching activities of the sum and carry signal for v_1 is calculated to be 11 and 6.75, respectively. It should be noted that v_1 has a fanout as 2 in the adder graph. Each transition of the sum output will cause the input capacitances of both RCAs driven by it to be charged or discharged as well. The PI of v_1 is therefore equal to $11 \times 2 + 6.75 = 28.75$ according to (5.16). Because switching of the sum signal will be propagated into another RCA and switching of carry signal will be propagated through the full adders within the same RCA, 11×2 is the inter-RCA PI and 6.75 is the intra-RCA PI for v_1 . Similar to v_1 , the other three RCAs, v_2 , v_3 and v_4 , at adder depth one have both inputs fed directly from the primary inputs. Therefore, by repeating the calculation above, the PI for v_2 , v_3 and v_4 are found to be 19.75, 28.75 and 21.75, respectively.

Besides, the RCAs at the next adder depth receive their inputs from the sum outputs of the RCAs in the first level instead of the primary input. In this case, $\alpha(X)$ can no longer be assumed to be 0.5, but is equal to the switching activity of the sum output, $SW(S_i)$ of its driving RCA. As shown in Fig. 5.5, v_5 has one input received from primary input and the other input driven by the output of v_1 . Therefore, $\alpha(Y) = 0.5$ and $\alpha(X) = SW(S_i)$ of v_1 . The adder, v_5 consists of 16 full adders and the PI is computed from the full adder at the least significant bit position. By (5.6) to (5.10), the switching activity matrix of v_5 is calculated to be:

i	1	2	3	4	5	6	7	8
$SW(C_i)$	0.25	0.375	0.438	0.469	0.484	0.492	0.621	0.779
$SW(S_i)$	0.5	0.75	0.875	0.938	0.969	0.984	0.992	1.121
i	9	10	11	12	13	14	15	16
$SW(C_i)$	0.905	0.992	1.047	1.08	1.1	1.111	1.117	1.121
$SW(S_i)$	1.279	1.405	1.492	1.547	1.58	1.6	1.611	1.617

By (5.16) and the values in the above switching activity matrix, the total switching activities of the sum and carry signals for v_5 is calculated to be 19.259 and 12.38, respectively. Since v_5 has a fanout of 1, the PI of v_5 is equal to $19.259 \times 1 + 12.38 = 31.639$ according to (5.16), with the inter-RCA PI equals 19.259 and the intra-RCA PI equals 12.38. The last RCA v_6 has one input fed by the primary input and the other input is received from the sum output of v_3 . Therefore, $\alpha(Y) = 0.5$ and $\alpha(X) = SW(S_i)$ of v_3 . By the similar computation as v_5 , the PI of v_6 is calculated to be 34.64.

The PIs for the six RCAs in this example are summarized as follows:

v_1	v_2	v_3	v_4	v_5	v_6
28.75	19.75	28.75	21.75	31.64	34.64

Therefore, by (5.16), the total PI of this MCM block is obtained to be 165.28, which includes the total inter-RCA PI of 110.77 and the total intra-RCA PI of 54.51.

On the other hand, the GPC and GPS results for this example are computed to be 8 and 97.37, respectively, which is conspicuously different from the PI. To verify how well these models correlate with the synthesized power results, it is necessary to run experiments on more benchmark filters and normalize the results obtained from these three models to the synthesized results. If the normalized ratios obtained are more stable with smaller variance, the model can be deemed to be more accurate. The details of these experiments are presented in the next section.

5.5 Results and Discussion

In this section, experimental results on five commonly cited benchmark FIR filters in the literature [PAS99, JAN02, VIN03, SAM89, LIM83] are analyzed. FIR1 and FIR2 are the filters in Figure 2 of [SAM89] and Example 1 of [VIN03], respectively. The coefficient sets of FIR3, FIR4 and FIR5 are obtained from Table VI of [PAS99], Figure 1 of [JAN02] and Table VI of [LIM83], respectively. Each filter is implemented by different multiplierless architectures obtained from the following algorithms: BHM [DEM95], Hartley [HAR96], MBPG [CHA08], NRSCSE [PEI02] and PASKO [PAS99]. GPC, GPS and proposed PI are computed for the adder graphs of these architectures. The estimated dynamic power dissipation are also obtained by technology mapping of these filter architectures to TSMC 0.18 μ m digital library using Synopsys Design Compiler. The results are tabulated in Table 5.1.

Table 5.1. GPC, GPS, PI and simulated power of FIR1 to FIR5

Algorithms	Filters	Glitch Path Count	Glitch Path Score	Power Index	Dynamic Power (mW)
BHM	FIR1	11	122	367.77	3.8263
	FIR2	110	1590	2970	9.2719
	FIR3	306	4090	6550	30.6703
	FIR4	2	16	106.22	1.7635
	FIR5	8	79	489.69	5.3803
HARTLEY	FIR1	9	111	336.48	3.8266
	FIR2	31	1220	2540	9.1904
	FIR3	113	3320	7050	31.7429
	FIR4	3	34	104.08	1.7202
	FIR5	13	137	534.57	5.3796
MBPG	FIR1	8	97	345.42	3.8297
	FIR2	55	1260	2300	9.4745
	FIR3	158	2630	5680	31.3023
	FIR4	2	18	110.90	1.7752
	FIR5	5	52	480.16	5.3796
NRSCSE	FIR1	9	111	336.48	3.8266
	FIR2	43	1130	2420	8.9431
	FIR3	113	3320	7050	31.7429
	FIR4	3	34	104.08	1.7202
	FIR5	13	137	534.57	5.3796
PASKO	FIR1	8	111	336.48	3.8297
	FIR2	59	1040	2500	8.4971
	FIR3	177	2480	6440	30.1718
	FIR4	3	34	104.08	1.7202
	FIR5	13	137	534.57	5.3796

Comparison of the relative power consumption deviations of the same filter from different algorithms provides some insight into the error margin of different switching power estimation. From Table 5.1, PI appears to be better correlated with the simulated dynamic power than GPC and GPS. An error comparison method in [DEM02b] is adopted to average out the random variances. The ratios of GPC, GPS and PI to their simulated power dissipation results for all filters are computed. Then, the ratio of each power estimate metric is normalized by its respective mean ratio on the same filter so that a normalized ratio of 1 represents perfect correlation. The normalized ratios for these

three power estimation models are summarized and plotted in Fig. 5.6 (a) to (e) for FIR1 to FIR5.

(c)

(d)

Fig. 5.6. Normalized ratios of GPC, GPS and PI to dynamic power for (a) FIR1 (b) FIR2 (c) FIR3 (d) FIR4 (e) FIR5.

From Fig. 5.6, the normalized ratios of the proposed PI always have the least variances to 1. Its variances are much lower than those of GPC and GPS. This is strong evidence that the proposed PI model is more correlated with the dynamic power dissipation than the other two models. These experiments also corroborate that GPS is more accurate than GPC in general, which is the same conclusion as [DEM02b]. This PI model is established based on input correlated switching activities of FA, the propagation of FA transitions and the varying load capacitances due to the sharing of adders in a reduced adder graph. Dynamic power simulations of ASIC implementation of benchmark filters indicate that the proposed PI measure is well correlated to the actual dynamic power dissipation and more accurate than the existing GPC and GPS measures.

5.6 Summary

In this chapter, a new high-level measure for modeling the dynamic power dissipation of reduced adder graph of FIR filter has been proposed. This PI model is established based on input correlated switching activities of FA, the propagation of FA transitions and the

varying load capacitances due to the sharing of adders in a reduced adder graph. Simulation results on the ASIC implementations of the benchmark filters indicate that the proposed PI measure is well correlated to the actual dynamic power dissipation and more accurate than the existing GPC and GPS measures.

In addition to the adder cost and logic depth, this PI model can be used to compare and evaluate the power performance of solutions generated by various design methodologies before the physical synthesis is performed on the best solution. It is envisioned that this newly established power index model can be tailored to the development of new power driven design methodology for MCM problems which is currently lacking.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

Transformations in high-level synthesis of Very Large Scale Integration (VLSI) circuits has led to the development of powerful optimization methods for many compute intensive digital signal processing problems. One important class of applications is digital filters which can be modeled as a generic multiple constant multiplication (MCM) problem by decomposing the computational intensive multiplier in their implementation into shifters and simpler arithmetic operators. The first part of this research has therefore focused on the development of optimization algorithms of multiplierless finite impulse response (FIR) digital filters. Another emerging research topic that has become popular in recent years is the reconfigurable constant multiplier. Research on new design methodology for area-efficient reconfigurable constant multiplier has been presented in this thesis. Additionally, this research also delves into the high-level (higher than gate level) dynamic power estimation technique for MCM block. A correlated power index model has been successfully established.

Multiplication of a variable by a set of constants has been a core operation appears abundantly in FIR filter design and low chip area, high-speed FIR filters are in great demand in many digital signal processing applications. Design methodologies that can deliver high quality solutions for rapid implementation into application-specific integrated circuit with reduced area and delay performance are desirable. In this research, a new Multi-root Binary Partition Graph (MBPG) data structure for optimizing the implementation of the multiplier block of FIR filters has been established. The well-

established information theoretic measures i.e., entropy and conditional entropy has been exploited by virtue of this data structure to optimally splitting and merging the signed digits in the coefficient sets into a MBPG with a significantly reduced number of vertices. Through tracking of dynamically revised weight-2 subexpression statistics, non-terminal vertices are competing for decomposition to ensure that the reuse of most potential weight two subexpressions is maximized as early as possible. The proposed algorithm solved the CSE problem by applying an information theoretic approach to quantitatively evaluate the redundancies of subexpression space from a set of coefficients. Unlike CSE algorithms based on other graph data structures, the symbol-level information carried in each vertex and the graph isomorphism of MBPG promise further fine-grain optimization in a reduced search space, which is demonstrated by a shift-inclusive vertex reordering to minimize the width of the two's complement adder. This new probabilistic method has been detailed in Chapter 3 of this thesis. Large sets of randomly generated filters and benchmark filters in the literature are used to evaluate and compare the critical path delays and area costs of the synthesized circuits. Experimental results show that the proposed algorithm is capable of producing more area-time efficient FIR filter solutions, with on average 19.3% lower area cost and 61.03% shorter delay than the existing design algorithms.

Chapter 4 of this thesis has presented a new approach to the design of low complexity and high-speed Reconfigurable Constant Multiplier (RCM). A Time-multiplexed Data Flow Graph (TDFG) has been established and a new scheduling methodology is proposed to assign all mobile operators into control steps according to their opportunity costs. The scheduling of the mobile operators in the TDFG maximizes the likelihood of merging compatible operators and hence lowers the overall adder and interconnecting multiplexer costs. This approach is the first proposal that delineates the RCM architecture as a DFG and minimizes its overall area-time complexity by an efficient heuristic operator scheduling algorithm. In addition, this is also the first methodology that attempt to overcome the deficiency of local search without using a fixed graph pattern for the partial sums and without resorting to exhaustive search among all minimal adder graphs. From the simulation results obtained from the benchmark coefficient sets and randomly

generated coefficients, it has been shown that this new RCM design methodology can produce solutions with up to 25% better area-time performance than the best existing method target for the same platform of implementation and without resorting to exhaustive search of high computational complexity. This RCM designed by this methodology has also been adopted in two new architectures for hardware implementation of circular convolution. These two architectures have reduced the cost of implementation significantly by 38% and 15.49%, respectively over the conventional parallel multipliers implementation, provided that their maximum achievable throughput rates limited by the coefficient wordlength and convolution size are fulfilled.

Despite the importance of power consumption for portable electronics, high-level power estimation techniques suitable for fast evaluation of alternative solutions in digital filter design have been lacking. In this thesis, a new Power Index (PI) model to estimate the dynamic power consumption of MCM block has been presented in Chapter 5. Comparing with the limited number of available approaches in the literature, the distinctive advantage of this proposed model is that it takes into account correlated input switching activity propagations of full adders within and across adders of different widths, as well as the variation of load capacitances due to sharing of adders in reduced adder graph. Simulation on dynamic power consumption of synthesized circuits have been analyzed and compared with two existing power estimation models has confirmed the closer correlation and higher accuracy of the proposed PI model.

The objectives set forth in Section 1.2 have been fulfilled with the four major contributions highlighted in Section 1.3 and summarized above. In conclusion, the outputs of this thesis have achieved the milestones towards the goal of establishing a comprehensive set of technology independent design methodologies that leads to shorter design turnaround time of the ASIC solutions to digital signal processing problems that can be optimized based on MCM structure.

6.2 Future Works

Based on the discovery and analysis by the research presented in this thesis, some potential extension and derivatives to new research topics and enhancement to the proposed design methodologies can be envisioned. A few promising directions for immediate future work are briefly discussed in this section. Some of these areas are currently being investigated by other members of the author's research group.

Design methodologies to optimize MCM block of fixed FIR filters by reducing adder cost have been adequately researched. The area cost and critical path delay of the MCM block of the transposed form FIR filters have been traditionally used and continue to be used as a quality measure of the hardware implementation of fixed FIR filters. From physical synthesis results, it is found that the structural adders and latches occupy a much higher proportion of the total area cost of moderate and large FIR filters. Unfortunately, no optimization approach has been targeted at the structural adders to date. This could be due to the thought that there is little exploitable redundancy in the tapped delay line. Nevertheless, due to the dominance of the structural adders to the total cost of the FIR filter, a small reduction in its implementation cost is worth an order of magnitude savings in the MCM block. Therefore, one of the future works is to identify the redundancies present in a series of consecutive carry propagation adders by investigating the input correlation between the current MCM block outputs and the preceding partial sums. New high level design methodology will then be developed to minimize the total area-time complexity of the MCM block and structural adder chain of the FIR filter.

In this thesis, CSD and MSD number systems are employed for the representation of the entire coefficient set in the proposed design methodologies for the MCM and RCM blocks, respectively. As discussed in Chapter 2, binary number system is another representation that can provide more chance of common subexpressions elimination, but this advantage is offset by the higher number of nonzero bits in the representation. If a given coefficient set can have the coefficients represented in mixed number format, and has the flexibility to convert from one representation to another in the course of the

optimization process, more common subexpressions are likely to be eliminated. Allowing multiple formats of subexpression representation and localized partial sum transformation from one form of representation to another could also reduce the number of spare non-zero digits that are not sharable by the common subexpressions at the expense of a reasonable increase in computational complexity. To reduce the search space, a preliminary assessment of the coefficients can be performed to determine the best representation form of each coefficient prior to the commencement of each iteration of the optimization algorithm.

In Chapter 5 of this thesis, a new input correlated Power Index model has been proposed to estimate the switching power of the MCM block. With this new model, the switching power for the adders and the entire MCM block can be accurately assessed before the logic synthesis. Instead of minimizing the adder cost and logic depth and passively accept the solution generated for whatever extent of power reduction it could accomplished, a more active approach is to incorporate power index as a criterion to drive the optimization algorithm. As the present power index model is not developed for this purpose, some modification is necessary to make it more efficient and amenable to power driven synthesis.

Author's Publications

Journal Papers

- [1] **Jiajia Chen** and Chip-Hong Chang, "High-level synthesis algorithm for the design of reconfigurable constant multiplier," *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, Regular Paper, vol. 28, no. 12, pp. 1844-1856, December 2009.
- [2] Chip-Hong Chang, **Jiajia Chen** and A. P. Vinod, "Information theoretic approach to complexity reduction of FIR filter design," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 8, pp. 2310-2321, September 2008.

Conference Papers

- [1] **Jiajia Chen**, Chip-Hong Chang and Hanhua Qian, "New power index model for switching power analysis from adder graph of FIR filter," in *Proceedings International Symposium on Circuits and Systems*, pp. 2197-2200, Taipei, Taiwan, 24-27 May 2009.
- [2] **Jiajia Chen**, Chip-Hong Chang and Ching-Chuen Jong, "Time-multiplexed Data Flow Graph for the design of configurable multiplier block," in *Proceedings International Symposium on Circuits and Systems*, pp. 1145-1148, Taipei, Taiwan, 24-27 May 2009.
- [3] **Jiajia Chen** and Chip-Hong Chang, "A tool for automated synthesis and optimization of integrated FIR filters," *IET Workshop on Microelectronics and Embedded Systems*, Singapore, January 2007.

- [4] U. Meyer-Baese, **Jiajia Chen**, Chip-Hong Chang and A. G. Dempster, "A comparison of pipelined RAG-n and DA FPGA-based multiplierless filters," in *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 1555-1558, Singapore, 4-7 December 2006.

- [5] **Jiajia Chen**, Chip-Hong Chang and A. P. Vinod, "Design of high-speed, low-power FIR filters with fine-grained cost metrics," in *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 756-759, Singapore, 4-7 December 2006.

- [6] Chip-Hong Chang, **Jiajia Chen** and A. P. Vinod, "Maximum likelihood disjunctive decomposition to reduced multirooted DAG for FIR filter design," in *Proceedings International Symposium on Circuits and Systems*, pp. 613-616, Kos, Greece, 21-24 May 2006.

- [7] Fei Xu, **Jiajia Chen**, Chip-Hong Chang and Ching-Chuen Jong, "A modified reduced adder graph algorithm for multiplier block minimization in digital filters," in *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 705-708, Tainan, Taiwan, 6-9 December 2004.

Bibliography

- [AIT93] D. Ait-Boudaoud and R. Cemes, "Modified sensitivity criterion for the design of powers-of-two FIR filters," *Electronics Letters*, vol. 29, pp.1467–1469, August, 1993.
- [AKO97] D. A. Akopian, O. Vainio, S. S. Agaian, and J. T. Astola, "SBNR processor for stack filters," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 3, pp. 197-208, March 1997.
- [AKS08] L. Aksoy, E. Costa, P. Flores and J. Monteiro "Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 6, pp. 1013-1026, June 2008.
- [ARN93] S. Arno and F. S. Wheeler, "Signed digit representations of minimal Hamming weight", *IEEE Transactions on Computers*, vol. 42, no. 8, pp. 1007-1010, August 1993.
- [ASH56] W. R. Ashby, *An Introduction to Cybernetics*, London: Chapman & Hall, 1956.
- [AVI61] A. Avizienis, "Signed-digit representation for fast parallel arithmetic," *IRE Transactions on Electronic Computer*, vol. 10, pp. 389-400, 1961.
- [BAR09] R. Barrak, A. Ghazel and F. Ghannouchi, "Optimized multistandard RF subsampling receiver architecture," *IEEE Transactions on Wireless Communications*, vol. 8, no. 6, pp. 2901-2909, June 2009.
- [BAU04] V. Baumgarte, G. Ehlers, F. May, A. Nüchel, M. Vorbach and M. Weinhardt, "PACT XPP—A Self-Reconfigurable Data Processing Architecture," *Journal of Supercomputing*, vol. 26, no. 2, pp. 167-184, November 2004.
- [BEN92] R. Barrak, A. Ghazel and F. Ghannouchi, "Optimized multistandard RF subsampling receiver architecture," *IEEE Transactions on Wireless Communications*, vol. 8, no. 6, pp. 2901-2909, June 2009.
- [BUL91] D. R. Bull and D. H. Horrocks, "Primitive operator digital filters," *IEE Proceedings G Circuits, Devices and Systems*, vol. 138, no. 3, pp. 401-412, June 1991.

-
- [CAP04] A. Cappeli, A. Lodi, C. Mucci, M. Toma and F. Campi, "A dataflow control unit for C-to-configurable pipelines compilation flow," *IEEE Symposium on Field-Programmable Custom Computing Machines*, pp. 332-333, 20-23 April 2004.
 - [CHA95] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits", *Proceedings of IEEE*, vol. 83, no. 4, pp. 498-523, April 1995.
 - [CHA06] C. H. Chang, J. Chen, and A. P. Vinod, "Maximum likelihood disjunctive decomposition to reduced multirooted DAG for FIR filter design," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 613-616, Kos, Greece, 21-24 May 2006.
 - [CHA07] C. Chao and K. K. Parhi, "High-throughput VLSI architecture for FFT computation," *IEEE Transactions on Circuits and Systems II*, vol. 54, no. 10, pp. 863-867, October 2007.
 - [CHA08] C. H. Chang, J. Chen and A. P. Vinod, "Information theoretic approach to complexity reduction of FIR filter design," *IEEE Transactions on Circuits and Systems*, vol. 55, no. 8, pp. 2310-2321, September 2008.
 - [CHA09] C. H. Chang and M. Faust, "On "A New Common Subexpression Elimination Algorithm for Realizing Low-Complexity Higher Order Digital Filters"," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, submitted.
 - [CHE05] H. C. Chen, J. I. Guo, C. W. Jen, and T. S. Chang, "Distributed arithmetic realisation of cyclic convolution and its DFT application," in *Proceedings IEE Circuits, Devices and Systems*, pp. 615-629, 9 December 2005.
 - [CHE09a] J. Chen, C. H. Chang and C. C. Jong, "Time-multiplexed data flow graph for the design of configurable multiplier block," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 1145-1148, Taipei, Taiwan, 24-27 May 2009.
 - [CHE09b] J. Chen, C. H. Chang and Hanhua Qian, "New power index model for switching power analysis from adder graph of FIR filter," in *Proceedings*

-
- International Symposium on Circuits and Systems*, pp. 2197-2200, Taipei, Taiwan, 24-27 May 2009.
- [CHE09c] J. Chen and Chip-Hong Chang, "High-level synthesis algorithm for the design of reconfigurable constant multiplier," *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, Regular Paper, vol. 28, no. 12, pp. 1844-1856, December 2009.
- [CHE99] C. L. Chen and A. N. Willson Jr., "A trellis search algorithm for the design of FIR filters with signed-powers-of-two coefficients," *IEEE Transactions on Circuits and Systems II*, vol. 46, no. 1, pp. 29-39, Jan., 1999.
- [DAS96] S. K. Das and M. C. Pinotti, "Fast VLSI circuits for CSD coding and GNAF coding," *Electronics Letters*, vol. 32, pp. 632-634, 1996.
- [DAV81] S. Davidson, D. Landskov, B. D. Shriver and P. W. Mallett, "Some experiments in local microcode compaction for horizontal machines," *IEEE Transactions on Computers*, vol. C-30, no. 7, pp. 460-477, July 1981.
- [DEH00] A. DeHon, "The density advantage of configurable computing," *IEEE Transactions on Computers*, vol. 33, no. 4, pp. 41-49, April 2000.
- [DEM95] A. G. Dempster and M. D. Macleod "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Transactions on Circuits Systems II*, vol. 42, no. 9, pp. 569-577, September 1995.
- [DEM98] A. G. Dempster and M. D. Marleod, "Comments on 'Minimum number of adders for implementing a multiplier and its application to the design of multiplierless digital filters'," *IEEE Transactions on Circuits Systems II*, vol. 45, no. 2, pp. 242-243, February 1998.
- [DEM00] S. S. Demirsoy, A. G. Demspter and I. Kale, "Transition analysis on FPGA for multiplier-block based FIR filter structures," in *Proceedings IEEE International Conference on Electronics, Circuits, and Systems*, vol. 2, pp. 862-865, Lebanon, 17-20 December 2000.
- [DEM02a] A. G. Dempster, S. S. Demirsoy and I. Kale, "Designing multiplier blocks with low logic depth," in *Proceedings IEEE International Symposium on Circuits and Systems*, Phoenix, Arizona, pp. 773-776, 26-29 May 2002.

-
- [DEM02b] S. S. Demirsoy, A. G. Dempster and I. Kale, "Power analysis of multiplier blocks," in *Proceedings IEEE International Symposium on Circuits and Systems*, Phoenix, Arizona, pp. 297-300, 26-29 May 2002.
 - [DEM03a] S. S. Demirsoy, A. G. Dempster and I. Kale, "Design guidelines for reconfigurable multiplier blocks," in *Proceedings IEEE International Symposium on Circuits and Systems*, Bangkok, Thailand, pp. 293-296, 25-28 May 2003.
 - [DEM03b] S. S. Demirsoy, R. Beck, A. G. Dempster and I. Kale, "Reconfigurable implementation of recursive DCT kernels for reduced quantization noise," in *Proceedings IEEE International Symposium on Circuits and Systems*, Bangkok, Thailand, pp. 289-292, 25-28 May 2003.
 - [DEM04] S. S. Demirsoy, I. Kale, and A. G. Dempster, "Efficient implementation of digital filters using novel reconfigurable multiplier blocks," *Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, pp. 461-464, 7-10 November 2004.
 - [DEM05a] A. G. Dempster and M. D. Macleod, "Multiplication by two integers using the minimum number of adders," in *Proceedings IEEE International Symposium on Circuits and Systems*, Kobe, Japan, pp. 1814-1817, 23-26 May 2005.
 - [DEM05b] S. Demirsoy, I. Kale, and A. G. Dempster, "Synthesis of reconfigurable multiplier blocks—Part I: Fundamentals," in *Proceedings IEEE International Symposium on Circuits and Systems*, Kobe, Japan, pp. 536-539, 23-26 May 2005.
 - [DEM05c] S. Demirsoy, I. Kale, and A. G. Dempster, "Synthesis of reconfigurable multiplier blocks—Part II: Algorithm," in *Proceedings IEEE International Symposium on Circuits and Systems*, Kobe, Japan, pp. 540-543, 23-26 May 2005.
 - [DEM07] S. Demirsoy, I. Kale and A. Dempster, "Reconfigurable multiplier blocks: structures, algorithms and applications," *Journal of Circuits, Systems and Signal Processing*, vol. 26, no. 6, pp.793-827, December 2007.

-
- [EST02] G. Estrin, "Reconfigurable computer origins: The UCLA fixed-plus-variable computer," *IEEE Annuals of the History of Computing*, vol. 24, no. 4, pp. 3-9, October – December 2002.
 - [FAL05] F. Fallah and M. Pedram, "Standby and active leakage current control and minimization cmos VLSI circuits," *IEICE Transactions on Electron*, vol. 88-C, no. 4, pp. 509-519, April 2005.
 - [FEN09] Z. Feng and P. Li, "Performance-Oriented Parameter Dimension Reduction of VLSI Circuits," *IEEE Transactions on very large scale integration systems*, vol. 17, no. 1, pp. 137 -150, January 2009.
 - [FLO05] P. Flores, J. Monteiro, and E. Coista, "An exact algorithm for the maximal sharing of partial terms in multiple constant multiplications," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 13–16, 6-10 November 2005.
 - [GEB87] C. H. Gebotys and M. I. Elmasry, "A VLSI methodology with testability constraints," in *Proceedings 1987 Canadian Conference VLSI*, Winnipeg, October, 1987.
 - [GIR84] E. F. Girczyc and J. P. Knight, "An ADA to standard cell hardware compiler based on graph grammars and scheduling," in *Proceedings IEEE International Conference on Computer Design*, pp. 726-731, October 1984.
 - [GUN99] J. E. Gunn, K. Barron, and W. Ruczczyk, "A low-power DSP core-based software radio architecture," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 574-590, April 1999.
 - [GUS01a] O. Gustafsson, H. Johansson, and L. Wanhammar, "An MILP approach for the design of linear-phase FIR filters with minimum number of signed-power-of-two terms," in *Proceedings European Conference of Circuit Theory Design*, Espoo, Finland, August, 2001.
 - [GUS01b] O. Gustafsson, H. Ohlsson, and L. Wanhammar, "Minimum-adder integer multipliers using carry-save adders," in *Proceedings IEEE International Symposium on Circuits and Systems*, Sydney, Australia, vol. 2, pp. 709-712, 6-9 May 2001.

-
- [GUS02a] O. Gustafsson, A. G. Dempster, and L. Wanhammar, "Extended results for minimum-adder constant integer multipliers," in *Proceedings IEEE International Symposium on Circuits and Systems*, Scottsdale, Arizona, USA, pp. I-73-I-76, 26-29 May 2002.
 - [GUS02b] O. Gustafsson and L. Wanhammar, "ILP modelling of the common subexpression sharing problem," in *Proceedings IEEE 9th International Conference on Electronics, Circuits and Systems*, vol. 3, pp. 1171-1174, 15-18 September 2002.
 - [GUS04] O. Gustafsson, A. G. Dempster, and L. Wanhammar, "Multiplier blocks using carry-save adders," in *Proceedings IEEE International Symposium on Circuits and Systems*, Vancouver, Canada, vol. 2, pp. 473-476, 23-26 May 2004.
 - [GUS06] O. Gustafsson, A. G. Dempster, K. Johansson, M. D. Macleod, and L. Wanhammar, "Simplified design of constant coefficient multipliers," *Circuits, Systems, and Signal Processing*, vol. 25, no. 2, pp. 225-251, April 2006.
 - [HAL05] E. Halperin and R. M. Karp, "The minimum-entropy set cover problem," *Theories of Computer Science*, vol. 348, no. 2, pp. 240-250, December 2005.
 - [HAR96] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Transactions on Circuits and Systems II*, vol. 43, no. 10, pp. 677-688, October 1996.
 - [HAR07] B. P. Harish, N. Bhat and M. B. Patil, "On a generalized framework for modeling the effects of process variations on circuit delay performance using response surface methodology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 3, pp. 606-614, March 2007.
 - [HAS96] R. Hashemian, "A new method for conversion of a 2's complement to canonic signed digit number system and its representation," in *Conference Record of Thirtieth Asilomar on Signals, Systems and Computers*, vol. 2, pp. 904-907, 3-6 November 1996.
 - [HAU97] Hauser, John R. and Wawrzynek, John, "Garp: A MIPS Processor with a Reconfigurable Coprocessor," in *Proceedings of the IEEE Symposium on*

-
- Field-Programmable Custom Computing Machines*, pp. 24–33, 16–18 April 1997.
- [HAW96] R. A. Hawley, B. C. Wong, T. J. Lin, J. Laskowski and H. Samueli, “Design techniques for silicon compiler implementations of high-speed FIR digital filters,” *IEEE Journal of Solid-State Circuits*, vol. 31, no. 5, pp. 656–666, May 1996.
- [HEN02] T. Hentschel, “Channelization for software defined base-stations,” *Journal of Annals of Telecommunications*, vol. 57, no. 5–6, pp. 386–420, May–June 2002.
- [HER01] J. P. Heron, R. Woods, S. Sezer, and R. H. Turner, “Development of a run-time reconfiguration system with low reconfiguration overhead,” *Journal of VLSI Signal Processing Systems*, vol. 28, no. 1–2, pp. 97–113, May 2001.
- [HER95] A. Herrfeld and S. Hentschke, “Look-ahead circuit for CSD-code carry determination,” *Electronics Letters*, vol. 31, pp. 434–435, 1995.
- [HEY01] F. Heylighen and C. Joslyn, “Cybernetics and second order cybernetics,” in *Encyclopedia of Physical Science & Technology*, vol. 4, New York: Academic Press, pp. 155–170, 2001.
- [HWA79] K. Hwang, *Computer Arithmetic: Principles, Architecture, and Design*. New York: John Wiley & Sons, 1979.
- [JAI91] R. Jain, P. T. Yang and T. Yoshino, “FIRGEN: A computer-aided design system for high performance FIR filter intergrated circuits,” *IEEE Transactions on Signal Processing*, vol. 39, no. 7, pp. 1655–1668, July 1991.
- [JAN02] Y. H. Jang and S. J. Yang, “Low-power CSD linear phase FIR filter structure using vertical common sub-expression,” *Electronics Letters*, vol. 38, no. 15, pp. 777–779, July 2002.
- [JOH04] K. Johansson, O. Gustafsson, and L. Wanhammar, “Power estimation for ripple-carry adders with correlated input data,” in *Proceedings International Workshop Power Timing Modeling, Optimization Simulation*, pp. 662–674, Santorini, Greece, September, 2004.
- [JOH05] K. Johansson, O. Gustafsson and L. Wanhammar, “A detailed complexity model for multiple constant multiplication and an algorithm to minimize the

- complexity," in *Proceedings European Conference on Circuit Theory Design*, Cork, Ireland, pp. 465-468, 28 August - 2 September 2005.
- [JOH08] K. Johansson, O. Gustafsson and L. Wanhammar, "Switching activity estimation for shift-and-add based constant multipliers," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 676-679, Seattle, Washington, USA, 18-21 May 2008.
- [KAI66] J. F. Kaiser, *Digital filters: systems analysis by Digital computer*, John. Wiley & Sons, 1966.
- [KAN01] H. J. Kang and I. C. Park, "FIR filter synthesis algorithms for minimizing the delay and the number of adders," *IEEE Transactions on Circuits and Systems II*, vol. 48, no. 8, pp. 770-777, August 2001.
- [KHI57] A. I. Khinchin, *Mathematical Foundations of Information Theory*. New York: Dover Publications Inc., 1957.
- [KIM07] Y. E. Kim, K. J. Cho and J. G. Chung, "Low power small area modified Booth multiplier design for predetermined coefficients," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 3, pp. 694-697, March 2007.
- [KUN85] S. Y. Kung, H. J. Whitehouse, and T. Kailath, *VLSI and Modern Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.
- [KWO07] G. Kwon, J. Lee, J. Jin and S. Ko, "Noise-Robust Modulation Identification Method for Adaptive Receiver Based on Software Defined Radio," *IEEE Transactions on Consumer Electronics*, vol. 53, no.3, pp. 1211-1216, August 2007.
- [LAS92] J. Laskowski and H. Samueli, "A 150-MHz 43-tap half-band FIR digital filter in 1.2 μ m CMOS generated by silicon compiler," in *Proceedings IEEE Custom Integrated Circuits Conference*, pp. 11.4.1-11.4.4, Boston, MA, 3-6 May 1992.
- [LI93] D. Li, J. Song, and Y. C. Lim, "A polynomial-time algorithm for designing digital filters with power-of-two coefficients," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 84-87, Chicago, 3-6 May 1993.

-
- [LI95] D. Li, "Minimum number of adders for implementing a multiplier and its application to the design of multiplierless digital filters," *IEEE Transactions on Circuits and Systems II*, vol. 42, no. 7, pp. 453-460, July 1995.
 - [LIM83] Y. C. Lim and S. R. Parker, "Discrete coefficient FIR digital filter design based upon an LMS criteria," *IEEE Transactions on Circuits and Systems*, vol. CAS-30, pp. 723-739, October, 1983.
 - [LIM91] Y. C. Lim, J. B. Evans, and B. Liu, "Decomposition of binary integers into signed power-of-two terms," *IEEE Transactions on Circuits and Systems*, vol. 38, pp. 667-672, 1991.
 - [LIU01] M. Liu, C. Chen, D. Shin, C. Lin and S. Jou, "Low-power multiplierless FIR filter synthesizer based on CSD code," in *Proceedings IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 666-669, May 2001.
 - [MAC01] J. Macbeth and P. Lysaght, "Dynamcially reconfigurable cores," in *Proceedings International Conference on Field-Programmable Logic and Applications*, vol. 2147, pp. 462-472, January 2001.
 - [MAC04] M. D. Macleod and A. G. Dempster, "Common subexpression elimination algorithm for low-cost multiplierless implementation of matrix multipliers," *Electronics Letters*, vol. 40, no. 11, pp. 651-652, May 2004.
 - [MAH06] R. Mahesh and A. P. Vinod, "A new common subexpression elimination algorithm for implementing low complexity FIR fillters in software defined radio receivers," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 4515-4518, Kos, Greece, 21-24 May 2006.
 - [MAH08] R. Mahesh and A. P. Vinod, "A new common subexpression elimination algorithm for realizing low-complexity higher order digital filters," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 2, pp. 217-229, February 2008.
 - [MEH95] M. Mehendale, M. Mehendale, S. D. Sherlekar, and G. Vekantesh, "Synthesis of multiplierless FIR filters with minimum number of additions," in *Proceeding IEEE/ACM International Conference of Computer-Aided Design*, Los Alamitos, CA, pp. 668-671, 1995.

-
- [MEH06] P. K. Meher, "Hardware-efficient systolization of DA-based calculation of finite digital convolution," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 8, pp. 707-711, August 2006.
 - [MEY04] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*. New York: Springer, 2004.
 - [MIC94] G. De Micheli, *Synthesis and Optimization of Digital Circuits*. New York: McGraw-Hill, Inc., 1994.
 - [MIL95] D. Miller, A. Rao, K. Rose, and A. Gersho, "A maximum entropy approach for optimal statistical classification," in *Proceedings IEEE Workshop Neural Networks for Signal Processing*, pp. 58-66, 31 August – 2 September 1995.
 - [MIR96] E. Mirsky and A. Dehon, "MATRIX: a reconfigurable computing architecture with configurable instruction distribution and deployable resources," in *Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines*, pp. 157-166, 17-19 April 1996.
 - [NOL92] T.G. Noll and E. De Man, "Pushing the performance limits due to power dissipation of future ULSI chips," in *Proceedings IEEE International Symposium on Circuits and Systems*, vol. 4, pp.1652-1655, 3-6 May 1992.
 - [OPP89] A. P. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, Prentice-Hall, 1989.
 - [PAR86] A. C. Parker, J. Pizarro and M. Mlinar, "MAHA: A program for datapath synthesis," in *Proceedings 23rd Design Automation Conference*, Las Vegas, pp. 461-466, July 1986.
 - [PAR99] K.K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*, John Wiley & Sons, 1999.
 - [PAR01] I. C. Park and H. J. Kang, "Digital filter synthesis based on minimal signed digit representation," in *Proc. IEEE Design Automation Conf.*, Las Vegas, Nevada, pp. 468-473, Jun. 2001.
 - [PAR02] I. C. Park and H. J. Kang, "Digital filter synthesis based on an algorithm to generate all minimal signed digit representations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, pp. 1525-1529, December 2002.

-
- [PAS99] R. Paško, P. Schaumont, V. Derudder, S. Vernalde and D. Ďuračková, "A new algorithm for elimination of common subexpressions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 1, pp. 58-68, January 1999.
 - [PAU89] P. G. Paulin and J. P. Knight, "Force-directed scheduling for behavioral synthesis of ASIC's," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 6, pp. 661-679, June 1989.
 - [PEI00] M. M. Peiro, and L. Wanhammar, "High-speed, low-complexity FIR filter using multiplier block reduction and polyphase decomposition," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 367-370, Geneva, Switzerland, 28-31 May 2000.
 - [PEI02] M. M. Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm," *IEEE Transactions on Circuits and Systems II*, vol. 49, no. 3, pp. 196-203, March 2002.
 - [POT96] M. Potkonjak, M. B. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplicaitons: Efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 2, pp. 151-165, February 1996.
 - [PRE00] L. L. Presti, "Efficient modified-sinc filters for sigma-delta A/D converters," *IEEE Transactions on Circuits and Sytems II*, vol. 47, no. 11, pp. 1204-1213, November 2000.
 - [PRO96] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice- Hall, 1996
 - [RRO98] J. G. Prokias and D.G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*. New Jersey: Prentice-Hall, 1998.
 - [RAI94] T. Raita-Aho, T. Saramaki and O. Vainio, "A Digital Filter chip for ECG Signal Processing," *IEEE Transactions on Instrumentation and Measurement*, vol. 43, no. 4, pp. 644 – 649, August 1994.

-
- [REI60] R. W. Reitwiesner, "Binary arithmetic," in *Advances in Computers*, New York: Academic Press, vol. 1, pp. 231-308, 1960.
 - [SAM89] H. Samuelli, "An improved search algorithm for the design of multiplier-less FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044-1047, July 1989.
 - [SAM90] H. Samuelli, "The design of multiplierless digital data transmission filters with powers-of-two coefficients," in *Proceedings IEEE International Telecommunications Symposium*, pp. 425-429, Rio de Janeiro, Brazil, September 1990.
 - [SHA06] Y. Shao and C. H. Chang, "A Kalman filter based on wavelet filter-bank and psychoacoustic modeling for speech enhancement," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 121-124, Kos, Greece, May 21-24, 2006.
 - [SHO05] T. Shono, Y. Shirato, H. Shiba, K. Uehara, K. Araki and M. Umehira, "IEEE 802.11 wireless LAN implemented on software defined radio with hybrid programmable architecture," *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, pp. 2299-2308, September 2005.
 - [SID04] N. Sidahao, G. A. Constantinides and Peter Y. K. Cheung, "Multiple Restricted Multiplication," in *Proceedings International Conference on Field-Programmable Logic and Applications*, pp. 374-383, 11 August 2004.
 - [SIN00] H. Singh, M. Lee, G. Lu, F. J. Kurdahi, N. Bagherzadeh and E. M. Chaves Filho, "MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications," *IEEE Transactions on Computers*, vol. 49, no. 5, pp. 465-481, May 2000.
 - [SPIRAL] Spiral Project: Software/Hardware Generation for DSP Algorithms. (Online). <http://www.spiral.net>.
 - [TAN02] Z. Tang, J. Zhang and H. min, "A high-speed, programmable, CSD coefficient FIR filter," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 4, pp. 834-837, November, 2002.

-
- [TSE86] C. Tseng and D. P. Siewiorek, "Automated synthesis of data paths in digital systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 5, no. 3, pp. 379-395, July 1986.
 - [TUM07] P. Tummeltshammer, J. Hoe, and M. Püschel, "Time-multiplexed multiple-constant multiplication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 9, pp. 1551-1563, September 2007.
 - [TUR02] Turner R. H., "Functionally diverse programmable logic implementations of digital signal processing algorithms", PhD Thesis, Queen's University of Belfast, August 2002.
 - [TUT02] Tuttlebee and H. W. Wally, *Software Defined Radio: Enabling Technologies*. New York: Wiley, 2002.
 - [VAI93] P. Vaidyanathan, *Multirate Systems and Filter Banks*. Prentice Hall, New Jersey, 1993.
 - [VIN03] A. P. Vinod, E. M. Lai, A. B. Premkuntar, and C. T. Lau, "FIR filter implementation by efficient sharing of horizontal and vertical common subexpressions," *Electronics Letters*, vol. 39, no. 2, pp. 251-253, January, 2003.
 - [VIN05a] A. P. Vinod, and E. M. Lai, "On the implementation of efficient channel filters for wideband receivers by optimizing common subexpression elimination method," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 2, pp. 295-304, February, 2005.
 - [VIN05b] A. P. Vinod and E. M. K. Lai, "An efficient coefficient-partitioning algorithm for realizing low-complexity digital filters," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 12, pp. 1936-1946, December 2005.
 - [VIN06] A. P. Vinod and E. M. K. Lai, "Low power and high-speed implementation of FIR filters for software defined radio receivers," *IEEE Transactions on Wireless Communications*, vol. 5, no. 7, pp. 1669-1675, July 2006.

-
- [WAN05] Y. Wang and K. Roy, "CSDC: a new complexity reduction technique for multiplierless implementation of digital FIR filters," *IEEE Transactions on Circuits and Systems I*, vol. 52, no. 9, pp. 1845-1853, September 2005.
 - [WU06] K. Wu and R. Karri, "Algorithm - level computing with shifted operands – a register transfer level concurrent error detection technique," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 3, pp. 413-422, March 2006.
 - [XU04] F. Xu, J. Chen, C. H. Chang and C. C. Jong, "A modified reduced adder graph algorithm for multiplier block minimization in digital filters," in *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 705-708, Tainan, Taiwan, 6-9 December 2004.
 - [XU05a] F. Xu, C. H. Chang and C. C. Jong, "Modified reduced adder graph algorithm for multiplierless FIR filters," *Electronics Letters*, vol. 41, no. 6, pp. 302 – 303, March 2005.
 - [XU05b] F. Xu, C. H. Chang and C. C. Jong, "Contention resolution algorithms for common subexpression elimination in digital filter design," *IEEE Transactions on Circuits and Systems II*, vol. 52, no. 10, pp. 695-700, October, 2005.
 - [XU05c] F. Xu, C. H. Chang and C. C. Jong, "I²CRA: contention resolution algorithm for intra- and inter-coefficient common subexpression elimination," in *Proceedings IEEE International Symposium on Circuits and Systems*, Kobe, Japan, pp. 1823–1826, 23-26 May 2005.
 - [XU07a] F. Xu, C. H. Chang and C. C. Jong, "Hamming Weight Pyramid - A New Insight into Canonical Signed Digit Representation and Its Applications," *Journal of Computer Systems Science & Engineering*, vol. 33, no. 3, pp. 195-207, May 2007.
 - [XU07b] F. Xu, C. H. Chang and C. C. Jong, "Design of low-complexity FIR filters based on signed-powers-of-two coefficients with reusable common subexpressions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 10, pp. 1898-1907, October 2007.

-
- [YAO04] C. Y. Yao, H. H. Chen, T. F. Lin, C. J. Chien and C. T. Hsu, "A novel common-subexpression-elimination method for synthesizing fixed-point FIR filters," *IEEE Transactions on Circuits and Systems I*, vol. 51, no. 11, pp. 2215-2221, November 2004.
- [YE04] Z. Ye, R. K. Satzoda, U. Sharma, N. Nazimudeen and C. H. Chang, "Performance evaluation of direct form FIR filter with merged arithmetic architecture," in *Proceedings 2nd IEEE International Workshop on Electronic Design, Test and Applications*, pp. 407-409, Perth, Australia, January 2004.
- [YOS90] T. Yoshiono, R. Jain, P. T. Yang, H. Davis, W. Gass and A. H. Shah. "A 100-MHz 64-tap FIR digital filter in 0.8- μ m BiCMOS gate array," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 6, pp. 1494-1501, December 1990.
- [YU07] Y. J. Yu and Y. C. Lim, "Design of Linear Phase FIR Filters in Subexpression Space Using Mixed Integer Linear Programming," *IEEE Transactions on Circuits and Systems I*, vol. 54, no. 10, pp. 2330-2338, October 2007.
- [YUR99] A. Yurdakul and G. Dundar, "Multiplierless realization of linear DSP transforms by using common two-term expressions," *Journal of VLSI Signal Processing*, vol. 22, no. 3, pp. 163-172, September 1999.
- [ZHA88] Q. Zhao and Y. Tadokoro, "A simple design of FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 5, pp. 566-670, May 1988.