Reconfigurable low complexity digital filter banks for software radio receivers

Raveendranatha Panicker Mahesh

2009

Raveendranatha Panicker Mahesh. (2009). Reconfigurable low complexity digital filter banks for software radio receivers. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/47469>

<https://doi.org/10.32657/10356/47469>

Nanyang Technological University

Downloaded on 26 Apr 2023 18:46:16 SGT

RECONFIGURABLE LOW COMPLEXITY DIGITAL FILTER BANKS FOR SOFTWARE RADIO RECEIVERS

RAVEENDRANATHA PANICKER MAHESH

School of Computer Engineering

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

2009

Acknowledgements

In the production of this work, I owe a great deal to my supervisor, Dr. Vinod A Prasad. His expertise and positive way of thinking has been of great value for me. The quality of the work and its publication success owe much to his inputs, constructive criticism and patient guidance. I am grateful to him for his personal advises which go beyond the technical work.

I thank Dr. Chang Chip Hong for many exciting and informative discussions on circuits and systems for digital signal processing. I also thank my family and friends for their care and support throughout the period of this work. I would also like to thank the School of Computer Engineering for the financial support provided in carrying out my research.

Above all, I thank and dedicate this work to my loving parents, Mrs. and Mr. K. C. Raveendranatha Panicker for the way they brought me up in life.

Contents

Acknowledgementsi

Table of Contentsii

List of Figuresvi

List of Tablesx

List of Abbreviationsxii

Abstractxv

Chapter 11

Introduction1

1.1 Motivation3

1.2 Objectives and Contributions4

1.3 Overview7

Chapter 28

Introduction to Software Defined Radio Receivers8

2.1 Overview of SDR8

2.2 SDR Architecture9

2.3 SDR Functionalities11

2.3.1 Analog-to-Digital Conversion11

2.3.2 Digital Front End12

2.3.2.1 Digital Down Conversion12

2.3.2.2 Channelization.....13

2.3.2.3 Sample Rate Conversion13

2.4 Channelization for SDR Receivers14

2.4.1 Per-Channel Approach16

2.4.1.1 Low-pass Filtering after Digital Down Conversion16

2.4.1.2	Band-pass Filtering before Digital Down Conversion	18
2.4.2	Filter Bank Approach	19
2.4.3	Frequency Domain Filtering Approach	23
2.5	Summary	24
Chapter 3		26
Hardware-Efficient Implementation Approaches for Filters and Filter Banks		26
3.1	Digital Filter Banks	26
3.2	Digital Filters for SDR Receivers	31
3.2.1	Complexity Analysis	34
3.2.2	Adder Complexity Analysis	43
3.2.3	Filter Reconfigurability	45
3.3	Summary	47
Chapter 4		49
Low Complexity Channel Filters using Binary Subexpression Elimination Algorithm ...		49
4.1	Binary Subexpression Elimination Algorithm	49
4.1.1	Binary Horizontal Subexpression Elimination (BHSE)	56
4.1.2	Binary Super Subexpression Elimination (BSSE)	58
4.1.3	Binary Vertical Subexpression Elimination (BVSE)	59
4.2	Illustrative Example	59
4.3	Extension of BSE to High-Level Synthesis	61
4.4	Design Examples	63
4.5	Summary	69
Chapter 5		71
Reconfigurable Low Complexity Channel Filters		71
5.1	Proposed Reconfigurable FIR Filter Architectures	71
5.1.1	Architecture of Constant Shifts Method (CSM)	75
5.1.2	Architecture of Programmable Shifts Method (PSM)	79

5.1.3	Comparison of CSM and PSM	82
5.2	Extension of CSM and PSM to High Level Synthesis	84
5.3	Experimental Results	86
5.3.1	Synthesis Results	86
5.3.2	CSD Based Reconfigurable FIR Filter Architecture	87
5.3.3	Design Results	91
5.4	Implementation Results	94
5.5	Summary	97
Chapter 6		99
Reconfigurable Low Complexity Filter Banks based on Frequency Response Masking Technique		99
6.1	Review of Frequency Response Masking (FRM) Technique	99
6.2	Proposed Reconfigurable Channel Filter	102
6.2.1	Architecture Level Reconfigurability	102
6.2.2	Filter Level Reconfigurability	105
6.2.3	Proposed Filter Architecture	105
6.2.4	Experimental Results	107
6.2.4.1	Design Results	107
6.2.4.2	Synthesis Results	108
6.2.4.3	Implementation Results	109
6.3	Proposed Reconfigurable Filter Bank (FB)	110
6.3.1	Architectural Reconfigurability	110
6.3.2	Filter level Reconfigurability	114
6.3.3	Proposed Channelizer Architecture	116
6.4	Extraction of Fractional Bandwidths Using the Proposed FB Architecture	122
6.4.1	Mode-1 Operation	123
6.4.2	Mode-2 Operation	125
6.4.3	Mode-3 Operation	126
6.5	Experimental Results	127
6.5.1	Qualitative Comparison	127

6.5.2	Quantitative Comparison	129
6.5.3	Design Example	133
6.6	Synthesis Results	135
6.7	Implementation Results	136
6.8	Summary	136

Chapter 7138

Coefficient Decimation based Filter Banks (CDFBs) with Absolute Control over Passband Width and Centre Frequencies		138
7.1	Proposed Coefficient Decimation (CD) Approach	139
7.1.1	Theoretical Background	139
7.1.2	Frequency Response Illustration of CD Approach	140
7.1.3	Proposed CD-Based Reconfigurable FIR Filter Architecture	142
7.1.4	Proposed CD-Based Reconfigurable Filter Bank Architecture	143
7.2	Design Procedure for the Proposed CD Approach	145
7.3	Channelization Comparison – CDFB V/s DFTFB	146
7.4	Complexity Evaluation	149
7.4.1	Multiple Channel Reception Complexity	150
7.4.2	Reconfiguration Complexity	151
7.5	Design Example	152
7.6	Implementation Results	155
7.7	Summary	156

Chapter 8157

Conclusions and Future Work		157
8.1	Conclusions	157
8.2	Future Work	162

Bibliography165

List of Publications171

List of Figures

Figure 2.1	The ideal software defined radio receiver	9
Figure 2.2	A feasible software radio receiver	10
Figure 2.3	Digital Front End	12
Figure 2.4	Bandpass ADC followed by digital down conversion	13
Figure 2.5	Per-channel approach	16
Figure 2.6	Channel-of-interest at baseband	17
Figure 2.7 (a)	Down conversion to baseband after band-pass filtering	20
Figure 2.7 (b)	Modified down conversion to baseband after band-pass filtering	20
Figure 2.8	Schematic of k^{th} filter bank branch containing M polyphase branches	21
Figure 2.9	Modified k^{th} filter bank branch containing M polyphase branches	21
Figure 2.10	DFT filter bank	22
Figure 2.11	Frequency domain filtering approach	24
Figure 3.1	Architecture of PFT approach	29
Figure 3.2	Direct form FIR filter structure	33
Figure 3.3	Transposed Direct form FIR filter structure	33
Figure 3.4	Filter tap implementation of (3.9) and (3.10)	44
Figure 4.1	Average values of N_{up} for binary and CSD representation of filter coefficients	51
Figure 4.2	Average values of DoS for binary and CSD representation of filter coefficients for different filter lengths	52
Figure 4.3	Average values of DoS for binary and CSD representation of filter coefficients for different coefficient wordlengths	53
Figure 4.4	Average values of N_{cs} for binary and CSD representation of filter coefficients	54
Figure 4.5	Average frequencies of occurrences of CSs in 16-bit coefficients of the example filters	55
Figure 4.6	Average frequencies of occurrences of CSs in the 120-tap filters for different wordlengths	55

Figure 4.7	BSE Realization of the filter with coefficients in Table 4.1	61
Figure 4.8	Subexpression sharing as a high-level synthesis transformation	62
Figure 4.9	Reduction of LOs in designing the filter with 200 taps using NR-SCSE [39], CRA [46], HCUB [47] and the proposed BSE method over direct method	66
Figure 4.10	Reduction of LOs in designing the filter with 16-bit coefficient word length using NR-SCSE [39], CRA [46], HCUB [47] and the proposed BSE method over direct method.....	67
Figure 4.11	Reduction of LOs in designing the D-AMPS channel filter with 610 taps using NR-SCSE [39], CRA [46], SS [48] and the proposed BSE method over direct method.....	69
Figure 4.12	Reduction of adders in designing the D-AMPS channel filter with 16-bit coefficient word length using NR-SCSE [39], CRA [46], SS [48] and the proposed BSE method over direct method.....	69
Figure 5.1	Transposed direct form of an FIR filter	72
Figure 5.2	Architecture of the processing element (PE)	73
Figure 5.3	Architecture of shift and add unit	73
Figure 5.4	Architecture of the PE for CSM	76
Figure 5.5	Architecture of the PE for PSM	81
Figure 5.6	Comparison of the number of addition operations for implementing filters with different filter lengths and coefficient wordlength of 16 bits	94
Figure 5.7	Implementation of the proposed CSM and PSM architecture on Virtex 2v3000ffl152-4 FPGA	95
Figure 6.1	FIR filter architecture based on FRM technique	100
Figure 6.2	Frequency Response illustration of FRM approach	101
Figure 6.3	Architecture of modal filter	106
Figure 6.4	Architecture of complementary delays	106
Figure 6.5	Proposed reconfigurable Filter architecture	114
Figure 6.6	Architecture of mode-10 proposed FB	116
Figure 6.7	Architecture of modal filter for mode-10 filter bank	117

Figure 6.8	Frequency edge specifications for expressions (6.10)-(6.17)	120
Figure 6.9	Architecture for extracting $(M+1)$ frequency bands	122
Figure 6.10	Cascaded connection of modal filters	123
Figure 6.11 (a)	Frequency response of modal filter	124
Figure 6.11 (b)	Frequency response of stage-I	124
Figure 6.11 (c)	Frequency response of stage-II	124
Figure 6.11 (d)	Frequency response of stage-III	124
Figure 6.12 (a)	Frequency response of stage-I	126
Figure 6.12 (b)	Frequency response of stage-II	126
Figure 6.12 (c)	Frequency response of stage-III	126
Figure 6.13 (a)	Frequency response of modal filter	134
Figure 6.13 (b)	Frequency response of case $M=10$	134
Figure 6.13 (c)	Frequency response of modal filter and FMA for case $M=12$	134
Figure 6.13 (d)	Frequency response of complementary delay output and FMC for case $M=12$	134
Figure 6.13 (e)	Overall frequency response of case $M = 12$ obtained by adding FMA2 (Fig. 6.13 (c)) and FMC2 (Fig. 6.13 (d)) as shown in the architecture of Figure 6.6	134
Figure 7.1 (a)	Frequency response of original modal filter	141
Figure 7.1 (b)	Frequency response of modal filter with $M=2$	141
Figure 7.1 (c)	Frequency response of modal filter with $M=3$	141
Figure 7.1 (d)	Frequency response of modal filter with $M=4$	141
Figure 7.1 (e)	Frequency response of decimated modal filter with $M=4$	141
Figure 7.2	Architecture of the proposed CD-based FIR filter	143
Figure 7.3	Architecture of the proposed filter bank	144
Figure 7.4 (a)	Frequency response at y_2-y_1	144
Figure 7.4 (b)	Frequency response at y_3-y_1	144
Figure 7.4 (c)	Frequency response at y_4-y_2	144
Figure 7.5 (a)	8-Channel DFTFB	146
Figure 7.5 (b)	Frequency response showing 5-channels	147

Figure 7.6 (a) Frequency response at y_1 147

Figure 7.6 (b) Frequency response at y_2 147

Figure 7.6 (c) Frequency response at y_4 147

Figure 7.6 (d) Frequency response at y_{4C} 147

Figure 7.7 (a) Frequency response with $M=7$ 153

Figure 7.7 (b) Frequency response with $M=3$153

Figure 7.7 (c) Frequency response with $M=5$153

Figure 7.8 (a) Spectrum of WCDMA output and input signal154

Figure 7.8 (b) Spectrum of CDMA output and input signal154

Figure 8.1 Hardwiring of partial product adders163

List of Tables

Table 2.1	Comparison of ADC technologies	11
Table 3.1	Comparison of Channelization Approaches	30
Table 3.2	VCSE in FIR filter Coefficients	40
Table 4.1	Binary representation of the filter coefficients	59
Table 4.2	Representation of the filter coefficients after BHSE	59
Table 4.3	Final Representation of the filter coefficients	60
Table 4.4	Simulation Results on benchmark filters	63
Table 4.5	No. of LOs for Example 2	65
Table 4.6	LD for Example 2	65
Table 4.7	No. of LOs for D-AMPS Filter example	68
Table 4.8	LD for D-AMPS Filter example	68
Table 5.1	Synthesis results for 3-bit BCSs and 4-bit BCSs based CSM architecture.....	79
Table 5.2	Synthesis results for an FIR Filter with 20 taps and coefficient wordlength of 16 bits	86
Table 5.3	Synthesis results of PSM architecture for different coefficient wordlengths.....	87
Table 5.4	Synopsys Synthesis results for 20-tap FIR Filter implementation of Section 5.3.2	89
Table 5.5	Synopsys Synthesis results for 32-tap FIR Filter implementation of Section 5.3.2	90
Table 5.6	Synopsys Synthesis results for 18-tap FIR Filter implementation of Section 5.3.2	91
Table 5.7	Synopsys Synthesis results for D-AMPS channel filter implementations	92
Table 5.8	Implementation results for proposed architectures with 20 taps and 16-bit coefficient wordlength	97
Table 6.1	Specifications of subfilters of Fig. 1 for different values of M	104

Table 6.2	Synopsys Synthesis results for CDMA/WCDMA reconfigurable channel filter implementations	108
Table 6.3	Implementation results	109
Table 6.4	Frequency specifications for the proposed filter bank	111
Table 6.5	Multiplication rate of channelizers	129
Table 6.6	Multiplication rate of a single channel channelizer	131
Table 6.7	Multiplication rate of a multiple channel channelizer	132
Table 6.8	Synopsys Synthesis Results	135
Table 6.9	Implementation results for Proposed FB Architecture	136
Table 7.1	Multiplication rate of Channelizers	151
Table 7.2	Implementation results	155

List of Abbreviations

AFE	Analog Front End
ADC	Analog-to-Digital Converter
BCS	Binary Common Subexpression
BHCS	Binary Horizontal Common Subexpression
BHSE	Binary Horizontal Subexpression Elimination
BSSE	Binary Super Subexpression Elimination
BVCS	Binary Vertical Common Subexpression
BVSE	Binary Vertical Subexpression Elimination
BH	Bull-Horrock's
CSD	Canonical Signed Digit
CPM	Coefficient-Partitioning Method
CSE	Common Subexpression Elimination
CS	Common Subexpression
CSM	Constant Shifts Method
CRA	Contention Resolution Algorithm
DoS	Degree of Sparseness
DFTFB	Discrete Fourier Transform Filter Bank
DPU	Digit Processing Unit
D-AMPS	Digital Advanced Mobile Phone Systems
DDC	Digital Down Conversion
DFE	Digital Front End
DSP	Digital Signal Processor
DF	Direct Form
DAG	Directed Acyclic Graph
FPGA	Field Programmable Gate Array
FIR	Finite Impulse Response
FDF	Frequency Domain Filtering
FA	Full Adder
GA	Genetic Algorithm

GD	Graph Dependence
HCSE	Horizontal Common Subexpression Elimination
HCS	Horizontal Common Subexpression
IIR	Infinite Impulse Response
IF	Intermediate Frequency
LD	Logic Depth
LO	Logical Operator
LUT	Look Up Table
MSD	Minimum Signed Digit
MILP	Mixed Integer Linear Programming
BHM	Modified Bull Horrock's
MSPS	Mega Samples Per Second
MSB	Most Significant Bit
MCM	Multiple Constant Multiplications
MMCM	Multiplexed Multiple Constant Multiplication
MB	Multiplier Block
MAC	Multiply-Accumulate
NRSCSE	Non-Recursive Signed Common Subexpression Elimination
PE	Processing Element
PSR	Peak Stopband Ripple
PC	Per-Channel
PDC	Personal Digital Cellular
PS	Programmable Shifter
PSM	Programmable Shifts Method
RF	Radio Frequency
ReMB	Reconfigurable Multiplier Block
RAG-n	Reduced Adder Graph-n-Dimensional
RCA	Ripple Carry Adder
SRC	Sample Rate Conversion
SA	Simulated Annealing
SDR	Software Defined Radio

SS	Super Subexpression
SSE	Super-Subexpression Elimination
TDF	Transposed Direct Form
VCSE	Vertical Common Subexpression Elimination
VCS	Vertical Common Subexpression

Abstract

Software Defined Radio (SDR) is a technology thought to build flexible radio systems, multi-service, multi-standard, multi-band, reconfigurable and reprogrammable by software. The fundamental idea of SDR is to replace most of the analog signal processing in the transceivers with digital signal processing in order to provide the advantage of flexibility through reconfiguration. This will enable different air-interfaces to be implemented on a single generic hardware platform. The most computationally intensive part in the digital front end of an SDR receiver is the channelizer as it operates at the highest sampling rate. The channelizer extracts multiple channels (frequency bands) from the wideband input signal using a digital filter bank. Reconfigurable low complexity channelizer is a vital part in SDR receivers. This thesis addresses the problem of incorporating low complexity and reconfigurability into the channelizer architecture.

Filter banks based on finite impulse response (FIR) filters are commonly employed in SDRs. Generally higher-order FIR filters are required in channelizers to meet the stringent adjacent channel attenuation specifications of wireless communication standards. Coefficient multiplications in such higher-order filters consume huge power and chip area. Apart from low complexity, reconfigurability of these filters and filter banks is important to support multi-standard operation of SDRs. In this thesis, five new methods are proposed for realizing reconfigurable low complexity channel filters and filter banks.

The first method makes use of common subexpression elimination (CSE) algorithm to reduce the complexity of coefficient multiplication in FIR filter architectures. The goal of CSE is to identify multiple occurrences of identical bit patterns, called common subexpressions, that are present in the coefficients, and eliminate redundant subexpression multiplications to minimize the number of adders needed to realize the coefficient multipliers. Conventional CSE methods use canonical signed digit (CSD)

representation of filter coefficients due to the inherent fewer number of nonzero bits in the CSD representation compared to binary representation. A new CSE technique called binary subexpression elimination (BSE) based on binary representation of filter coefficients has been proposed in this thesis, which resulted in better reduction in the number of adders than CSD-based CSE methods in literature. Design examples of FIR filters show that the proposed method offers an average adder reduction of 18% over the best known CSE method in literature, without any increase in delay of filtering operation.

As the second contribution, two new reconfigurable low complexity FIR channel filter architectures called the constant shifts method (CSM) and the programmable shifts method (PSM) have been proposed. In contrast to conventional shift and add units used in existing reconfigurable filter architectures, the binary common subexpressions-based shift and add unit have been employed in the proposed CSM and PSM architectures. The CSM produces high speed filters whereas the PSM produces filters that have less power consumption.

The implementation of reconfigurable higher-order filters based on conventional filter design techniques, such as Parks McClellan technique, is expensive. Frequency response masking (FRM) was a technique originally proposed for the design of sharp transition-band FIR filters with extremely low complexity. The basic idea was to compose the over-all sharp transition-band filter using three wide transition-band subfilters, which have lower orders and consequently low complexity. The third contribution in this thesis is the integration of reconfigurability into the FRM architecture. A reconfigurable filter bank and channel filter based on the FRM technique have been proposed, which have the advantages of extracting extremely narrowband and non-uniform bandwidth channels, and multimode operation, which were not achieved using conventional filter banks.

In SDR channelizers, filter banks need to extract multiple channels simultaneously and these channels can have bandwidths that are not related by integer factors. The simultaneous extraction of such channels using conventional filter banks is impossible. Based on the FRM approach, a new filter bank that can extract the channels whose

bandwidths are related by fractional factors has also been proposed. This forms the fourth contribution.

A new filter and filter bank based on a novel coefficient decimation technique that has absolute control over location of center frequencies of passbands forms the fifth work. An exact low complexity reconfigurable filter bank substitute for discrete Fourier transform based filter bank using the coefficient decimation approach has also been proposed.

All the proposed architectures have been implemented and tested on Xilinx Virtex 2v3000ff1152-4 FPGA. Implementation results show that the proposed architectures offer good trade-off between low complexity and reconfigurability.

Chapter 1

Introduction

The wireless communication industry has been experiencing an exponential growth with new radio access technologies and standards coming into the picture. Lack of harmony in spectrum allocation globally has also resulted in this growth. All these technologies have been optimized to obtain a good trade-off between data rate, range and mobility to suit specific application needs. But with the increase in trade relationship between different continents, researchers had to look for a common platform which can support all these radio technologies and standards. This has resulted in the birth of the software defined radio (SDR) concept [1-6]. SDR can be regarded as an ultimate solution which can cover any cellular communication standard in a wide frequency spectrum with any modulation and bandwidth.

The term SDR signifies that the same hardware architecture can be programmed or reconfigured to cope with any radio standard. The major application of SDR will be in mobile communication transceivers, generic cellular base stations and military radio systems. Some of the benefits that will result with the realization of SDR are:

- Easier international roaming, improved and more flexible services, increased personalization and choice for subscribers of mobile services.
- The potential to rapidly develop and introduce new value-added services and revenue streams with increased flexibility of spectrum management and usage for mobile network operators.
- The promise of increased production flexibility and improved, more rapid production evolution for handset and base station manufacturers.
- The prospect of increased spectrum efficiency and better use of scarce resources for regulators.

Currently, cellular base stations employ distinct receiver chain consisting of analog mixers, local oscillators, analog-to-digital converters (ADCs) and baseband processing units for each communication standard. Thus the complexity of these base

stations is dominated by analog components and it grows *linearly* with the number of received standards. But SDR based base stations employ a single analog and digital front end to receive all the communication standards. The digital front end, which does the entire signal processing tasks, is reconfigured to work with different communication standards. Hence the cost of the analog part is independent of the number of received standards.

The introduction of SDR concept has initiated the rapid transition of communications technology from analog to digital means. This has enabled different communication functionalities such as digital up/down conversion, modulation/demodulation, etc., to be performed by software on an appropriate digital hardware. The use of such programmable digital hardware has improved the flexibility of communication to a significant extent.

In order to become a reality, the SDR technology requires more investigation and research in the areas of wireless communications, digital signal processing and computer arithmetic. The major research in SDR technology can be grouped into the following areas:

- Analog front end (AFE): The AFE consists of the antenna and the analog filter to cope with the radio frequency (RF) signals. The research in AFE deals with tunable RF filters and advances in antenna technology to suit the versatility of SDR concept [66, 67].
- ADCs and digital-to-analog converters (DACs): ADCs and DACs in SDR are ideally placed as close to the antenna as possible. This requires fast ADCs and DACs satisfying the low power and high accuracy requirements of wireless communication systems. Currently, the highest speed of operation available for a commercial ADC is 130 mega samples per second (MSPS) for a wordlength of 12 bits, which is still not the best for SDRs [8].
- Digital Front End (DFE): The main function of DFE is to extract individual radio channels from the digitized wideband signal at the ADC output and to provide the extracted channels at desired sampling rate for baseband processing. Since the DFE comes directly after ADC in an SDR receiver, it needs to operate at very high sampling rate [2]. The DFE must be realized to

meet the low power requirements of wireless communication receivers. It should also be reconfigurable to adapt to different communication standards. The realization of a less complex and reconfigurable DFE is a challenging task.

- Advanced baseband processing: This includes challenges in areas such as smart antennas, multi-user detection, spread-spectrum processing, data coding and security [67].

The focus of the work presented in this thesis is on low complexity implementation of reconfigurable DFE of an SDR receiver. The basic idea of SDR is to replace the conventional analog signal processing in radio transceivers by digital signal processing. This is achieved by placing the ADC in receivers (DAC in transmitters) as close to the antenna as possible. The SDR should be able to use the same architecture for any number of channels by reconfiguring the DFE as compared to a conventional radio transceiver whose complexity grows linearly with the number of channels. In addition to these reconfigurability requirements, the hardware employed for SDR must also meet the stringent power and speed specifications of the wireless communication systems.

1.1 Motivation

The part of the SDR terminal where the analog signal processing is replaced by digital signal processing is referred to as the DFE [1, 2]. The main functionalities of the DFE include channelization, digital down conversion and sample rate conversion (down sampling). The channelizer, which extracts multiple radio channels from the wideband input signal, comes directly after the ADC and hence needs to operate at the highest sampling frequency. Thus the channelizer is the most computationally intensive block in the DFE. The channelizers in SDR receivers must be realized to meet the stringent specifications of low power consumption and high speed. The channelizer extracts individual radio channels with the help of a bank of digital filters commonly referred to as *channel filters*. Finite impulse response (FIR) filters are widely employed as channel filters due to their absolute stability and linear phase characteristics. The main challenges in designing an efficient filter bank channelizer are:

- The large bandwidth of input signal requires high speed digital filters.

- Stringent requirements of adjacent channel attenuation in channel filters necessitate the use of sharp transition-band filters, which would require large number of filter taps (coefficients).
- The channel filters must be implemented with low area and low power consumption to satisfy the resource-constrained battery operated mobile wireless communication systems.
- Flexibility and programmability of channel filters for multi-standard receiver operation.

Many architectures have been proposed in the literature for efficient implementation of the FIR filters and filter banks for communication receivers. But these architectures must be further optimized or new architectures need to be developed to meet the area, power and speed constraints of the channelizer in an SDR. Reconfigurability of the channelizer to work with multiple wireless communication standards is another key requirement in an SDR which has not been adequately addressed in literature. The conventional method of achieving reconfigurability by switching the operation among distinct receivers based on the current mode of operation is not an efficient approach from the perspective of resource utilization and power consumption. Integrating complexity reduction into a reconfigurable SDR is a challenging task as these two requirements often present optimization of contradicting design tasks.

The motivation behind the work in this thesis is to deal with the research issues related to the realization of high speed reconfigurable digital filter banks with low power consumption and area, which have been hardly addressed in literature.

1.2 Objectives and Contributions

This thesis addresses the design and realization issues of reconfigurable and hardware-efficient digital filter banks for SDR receivers. The main objective in this thesis is to realize low area, low power and high speed digital filter banks for SDR channelizers, which are capable of reconfiguring to different wireless communication standards with minimum overhead. Coefficient multiplication is the most expensive operation in digital filters and filter banks as it consumes most of the area, power and time of

filtering operations. The number of additions (subtractions) used to implement coefficient multiplication determines the hardware complexity of the filter. Hence the first objective is to minimize the number of adders in the coefficient multipliers of channel filters. The second objective is to incorporate reconfigurability into the digital filters that are realized using hardware optimized coefficient multipliers. Once a good trade-off between reconfigurability and low complexity is achieved at the filter level, the third objective is to extend the work to propose reconfigurable multimode filter bank channelizers. The final objective is the hardware implementation of the proposed filter and filter bank architectures on FPGAs and the comparison of synthesis results with existing methods.

The **contributions** of this thesis are:

1. A new common subexpression elimination (CSE) technique based on binary representation of filter coefficients which produces better reduction in the number of adders needed to implement the coefficient multipliers than other CSE methods in literature has been proposed. The critical path lengths (termed logic depth) of the coefficient multipliers realized using proposed binary representation based CSE method are almost identical to those realized using conventional techniques. Thus the proposed CSE method produces filters with low complexity coefficient multipliers without increasing the delay.
2. Two new reconfigurable, low complexity FIR filter architectures for SDRs have been proposed. The first method known as constant shifts method (CSM) focuses on direct implementation of FIR filters without using programmable shifters (PSs) and thus results in high speed operation. The second method known as programmable shifts method (PSM) employing PSs offers significant power reduction. In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, the binary common subexpressions-based shift and add unit has been employed in the proposed CSM and PSM architectures. To the best of author's knowledge, this is the first attempt towards integrating reconfigurability into CSE techniques for implementing higher-order FIR filters.
3. A reconfigurable filter bank and reconfigurable channel filter have been proposed based on a frequency response masking (FRM) technique. The basic idea is to interpolate a filter whose response has very wide transition-band and then to mask

out un-desired frequency bands, obtained while interpolating the filter, by employing wide transition-band masking filters. All the filters in FRM technique are having wide transition-band and hence they are less complex. The proposed reconfigurable filter bank overcomes three main deficiencies of conventional filter banks in SDR channelizers: Non-uniform bandwidth extraction, very narrowband channel extraction and multi-standard operation. The proposed channel filter and filter bank have the flexibility of dynamically changing of frequency responses with very low complexity.

4. In SDR channelizers, filter banks need to extract multiple channels simultaneously and these channels can have bandwidths which are not related by fractional factors. Based on the FRM approach, a new filter bank has also been proposed which can extract channels, whose bandwidths are related by fractional factors. The simultaneous extraction of such channels is impossible using conventional filter banks. The proposed filter bank offers reconfigurability at the architectural level and at the channel filter level.
5. A new filter and filter bank based on a novel coefficient decimation approach, which has absolute control over location of center frequencies of passbands form the next contribution. The basic idea is as follows: if the coefficients of an FIR filter are decimated by M , i. e., if every M^{th} coefficient of the filter is kept unchanged and remaining coefficients are changed to zeros, a multi-band frequency response will be obtained. The resulting frequency responses will have centre frequencies at $2\pi k/M$, where k is an integer ranging from 0 to $M-1$. If these multi-band frequency responses are selectively masked using inherently low complexity wide transition-band masking filters, different low-pass, high-pass, band-pass, and band-stop filters can be obtained. If every M^{th} coefficient is grouped together removing the zero coefficients in between, a decimated frequency response in comparison to the original frequency response is obtained. An exact low complexity reconfigurable filter bank substitute for discrete Fourier transform filter bank (DFTFB) has also been proposed using the above approach.
6. All the proposed architectures have been implemented on Xilinx Virtex 2v3000ff1152-4 FPGA and tested using real-time inputs.

1.3 Overview

The rest of this thesis is organized as follows. Chapter 2 presents the basics of SDR and various functionalities of the channelizer. In Chapter 3, the background knowledge and literature review pertaining to the design and implementation of different filter and filter bank architectures are presented. The proposed binary subexpression elimination (BSE) algorithm is presented in Chapter 4. Chapter 5 presents the proposed reconfigurable BSE architectures. In Chapter 6, filter bank realization based on frequency response masking technique is presented. Chapter 7 presents the proposed reconfigurable filters and filter banks based on a coefficient decimation approach. This thesis is concluded with an account of its contributions and possible future directions in Chapter 8.

Chapter 2

Introduction to Software Defined Radio Receivers

In this chapter, the basic concept of a software defined radio receiver (SDR) is presented and the functionalities of various blocks in the receiver architecture are studied. The emphasis in this chapter is to provide an insight of challenges in realizing the channelizer in the digital front end of the SDR receiver. A review of conventional channelizer architectures is also provided in this chapter.

2.1 Overview of SDR

A rigorous and exact definition of SDR is still difficult to formulate. In [4], a satisfactory definition was given as *“Software radio is an emerging technology, thought to build flexible radio systems, multi-service, multi-standard, multi-band, reconfigurable and reprogrammable by software”*. For realizing an SDR that satisfies this definition, many challenging research issues need to be addressed. The two main tasks that need to be achieved are [1-5]:

1. To replace most of the analog signal processing in conventional radio receivers by digital signal processing and achieving this aim by adopting the ADC and DAC conversions as close to the antenna as possible.
2. To replace application-specific integrated circuits (ASICs) with programmable digital signal processors (DSPs) or field programmable gate arrays (FPGAs) for intermediate frequency (IF) processing, in order to define as many radio functionalities as possible in software.

The replacement of ASIC technology with DSP/FPGA opens the way in two possible horizons [4]:

1. Software implementation of IF and baseband functions, such as coding, modulation, channelization, equalization, and pulse shaping.
2. Reprogrammability of the system to guarantee multi-standard operation.

2.2 SDR Architecture

The basic idea in an SDR is to employ digital signal processing as close to the antenna as possible. The complexity of the SDR receiver is typically four times that of the transmitter [6]. Thus the receiver implementation has a first-order impact on the hardware cost of SDR. Hence the focus in this work is on SDR receivers. The block diagram of such an ideal SDR receiver commonly known as ‘*software radio receiver*’ is shown in Fig. 2.1 [4]. In Fig. 2.1, the low noise amplifier (LNA) amplifies the input signal to a desired level. This is followed by an anti-aliasing filter to band-limit the input signal to prevent aliasing while sampling the signal using the ADC. Once the signal is converted to digital form using the ADC, the DSP will perform channelization and all necessary signal processing tasks which were earlier achieved using analog circuits. The main challenges for the realization of an ideal SDR are as follows:

- In an ideal SDR, since all the channelization tasks are performed digitally, ADC must process the total signal bandwidth. However according to Nyquist criterion, the sampling frequency must be twice the RF bandwidth of the signal which is practically infeasible. Although interleaved sampling using a parallel bank of ADCs can solve this problem, it is not an area-efficient and low power solution in resource constrained mobile SDR receivers.
- The dynamic range of RF signal is very high. Therefore digitization of RF signal is beyond the scope of the today’s ADCs. Currently, the highest speed of operation available for a commercial ADC is 130 MSPS for a wordlength of 12 bits [8].

Figure 2.1 The ideal software defined radio receiver.

Thus the ideal SDR receiver shown in Fig. 2.1 is not realizable in practice with today’s technology. A more feasible SDR receiver architecture is shown in Fig. 2.2 [4]. In Fig. 2.2, the LNA and anti-aliasing filter perform the same function as that of the receiver in Fig. 2.1. An RF image filter is used to remove the image frequencies that can affect the

output of the mixer in Fig. 2.2. The mixer is used to down convert the frequency of the input signal from RF to IF so that currently available ADCs can easily digitize the IF signal. The portion of the SDR which includes LNA, RF image filter, mixer and anti-aliasing filter is known as the AFE. The main function of the AFE is to reduce the frequency from RF to IF and to bandlimit the input signal to prevent aliasing. From the above discussion, it can be concluded that only partial band digitization is possible in a feasible SDR receiver in contrast to full band digitization envisaged for an ideal SDR. In order to cover all services to be supported by the SDR receiver, a limited band has to be selected out of the full band by means of analog conversion and IF filtering.

Figure 2.2 A feasible software radio receiver.

Once the input signal is band limited to IF using the AFE, the ADC will digitize the IF signal for further digital signal processing in the DFE of the receiver. The DFE performs digital down conversion (DDC), sample rate conversion (SRC) and channelization as shown in Fig. 2.2, which will be explained in more details in the next section. Thus the output of DFE will have individual radio channels at baseband extracted from the IF input signal. The DFE, which comes directly after the ADC, needs to operate at IF. Thus the DFE cannot be realized efficiently using DSPs because of speed constraints associated with DSPs. Also hardware optimization is very limited when DFE is implemented in DSPs [22]. Hence dedicated hardware architectures are required for implementing DFE [2]. Once all the channels are obtained at baseband at the output of DFE, they can be further processed using DSPs.

2.3 SDR Functionalities

The critical functionalities of an SDR receiver related to digital signal processing are discussed in this section.

2.3.1 Analog-to-Digital Conversion

The analog-to-digital conversion is employed to digitize the wideband input signal. Basically there are two approaches for ADCs [7]:

1. *Full Band Digitization*: In this case, the whole bandwidth containing all channels of all services need to be digitized. This bandwidth can be over 100 MHz which means the dynamic range of ADCs have to be larger than 100 dB. Even though full band digitization is the best solution, it is not efficiently realizable with today's ADC technology.
2. *Partial Band Digitization*: In this case, only the bandwidth equal to the widest channel bandwidth of all the supported services need to be digitized. This can be easily achieved using currently available ADCs.

Recently, different implementation technologies have been experimented to improve the performance of ADCs [8]. Though the semiconductor approach has been the forerunner for ADC implementation technology, the recent trend is to employ alternative technologies like optical sampling and super conductor technology. In semiconductor technology, the resolution degrades by 1-bit for every doubling of the sampling rate. This variation is known as aperture jitter and can be reduced to a significant extent by employing optical sampling. The other breakthrough technology that has opened the doors for high speed ADCs with improved resolutions is the rapid single flux quantum (RSFQ) [8, 80]. This technology is based on a fundamental quantum mechanical property of super conductors, stating the existence of magnetic flux in discrete quantized form. The speed and resolution comparison of different ADC technologies are shown in Table 2.1 [8].

Table 2.1
Comparison of ADC technologies

ADC Technology	Resolution	Speed	Status
Semiconductor Based	6 bits	3200 MS/s	Commercially Available
Optical Sampling	8.2 bits	505 MS/s	Experimental
Superconductor RSFQ	11 bits	175 MS/S	Experimental

From Table 2.1, it can be concluded that, it is basically a trade-off between technology, speed of operation and resolution to select the best ADC solution for SDR receivers.

2.3.2 Digital Front End

The DFE performs digital down conversion, sample rate conversion and channelization as shown in Figs 2.2 and 2.3. The DFE must deliver a digital signal which is ready for baseband processing, with a sample rate determined by the current air-interface. This digital signal represents the channel-of-interest of bandwidth, B_i centered at $f_c = 0$. In Fig. 2.3, the shaded frequency band represents the channel-of-interest. The DDC down-converts the wideband signal so that the channel-of-interest is at baseband. The channel filter isolates the channel-of-interest from the adjacent channels. The SRC performs downsampling so that, the sampling frequency of the signal is twice the bandwidth of the channel-of-interest. A brief analysis of the three stages of operation (DDC, Channelization and SRC) in the DFE is presented in the following sections.

Figure 2.3 Digital Front End.

2.3.2.1 Digital Down Conversion

A fundamental operation in many communication systems is DDC. SDR receivers often have fast ADCs delivering vast amounts of data; but in many cases, the channel-of-interest represents a small proportion of that bandwidth. The DDC allows the rest of that data to be discarded, allowing more intensive processing to be performed on the channel-of-interest. When partial band digitization is employed, the basic channel selection task is

performed in the analog domain by converting the signal-of-interest to a fixed IF. Hence DDC can be done using relatively simple hardware. The two parameters that influence the effort of implementing DDC are the frequency value of IF, f_{IF} , and the sample rate, f_s . If the ratio of frequency value of IF and sample rate obeys the rule,

$$f_{IF} = \frac{n}{4} f_s, n=1, 3, 5, \dots \quad (2.1)$$

i. e., the frequency value of IF is an odd multiple of a quarter of the sample rate, DDC can be performed by multiplying the signal with the sequences $[0 \ 1 \ 0 \ -1]$ and $[1 \ 0 \ -1 \ 0]$, representing the digital sine-signal and cosine-signal, respectively, at a quarter of the sample rate. Given the constraint of equation 2.1, digitization and down conversion of band-pass signals can be combined. Basically the task of band-pass analog-to-digital conversion is followed by an I-Q down conversion [5] as shown in Fig. 2.4.

Figure 2.4 Bandpass ADC followed by digital down conversion.

2.3.2.2 Channelization

Channelization comprises all tasks necessary to select the channel-of-interest. This includes channel filtering and de-spreading. For the realization of channelization, DSPs are not very efficient because of high sampling rate requirements. Hence dedicated hardware has to be designed. A detailed overview of channelizer, which is main area of work in this thesis, is given in the Section 2.4.

2.3.2.3 Sample Rate Conversion

An SDR should be able to process signals of different wireless communication standards. Thus it must process signals at different sample rates or symbol rates. The sample rate can be either made adaptive to different standards or fixed followed by a digital sample rate conversion. Basically two approaches are possible for sample rate conversion:

1. Both IF and f_s are made parameterizable by keeping the ratio between them according to the equation (2.1). In order to simplify the down conversion process, each signal can be digitized with the clock rate of the standard of current operation.
2. If the digitization is done with a clock rate fixed at IF, sample rate conversion can be performed by means of mathematical interpolation. This can be implemented in different ways:
 - a) The straight forward way of performing sample rate conversion is to interpolate and decimate the signal by integer factors. But if no proper image and alias rejection is done, this method results in improper results especially for signals with high dynamic ranges. Hence this approach results in expensive implementations.
 - b) Reduce the dynamic range of the signal by means of sharp cut-off channel selection filters, which would cut down the cost of approach in (a).
 - c) Assuming a block-wise processing, a certain number of samples per block can be dropped in order to reach the desired sample rate. The error thus introduced should be cancelled after the dropping or the signal has to be predistorted in a way that the dropping leads to a distortion-free signal. This approach is referred to as 'asynchronous decimation'.

The last approach seems to be the simplest one, avoiding parameterization of analog components as in the first approach. In the case where error cancellation can be performed efficiently, the dropping process is most elegant and straight-forward [5].

2.4 Channelization for SDR Receivers

Channelization involves the extraction of individual radio channels from wideband input signal by band-pass digital filters known as channel filters. Channelization is the first stage of digital signal processing where parameters such as bandwidth and sample-rate are changed according to the current mode of operation. For SDR, it is desirable to realize as much of the channelization functionalities as possible with digital signal processing. The basic assumptions of channelization are [3, 9]:

- The AFE employs complex I-Q down conversion.
- The base-station should be able to receive N independent channels in parallel.
- All channels can have the same bandwidth (i.e., stem from the same air-interface) or can have different bandwidth (i.e., simultaneous reception of channels of different air-interfaces).
- The bandwidth of the channels must be variable or tunable (i.e., defining the channel bandwidth by software).
- The bandwidth, B , of the digitized signal is as large as possible. Typically the number of channels of interest, N_I , is usually very small compared to the number of channels the bandwidth B comprises.

Two key issues that determine the technical requirements of the channelizer are the spectral content of the wideband channel-of-interest and the types of processing devices available for channelization [10]. The technical requirements for the channelization approach are driven by the frequency allocation plans supported by the SDR. This can range from fixed carrier spacing with a constant RF bandwidth per carrier channel as in cellular communications to variable carrier spacing and RF bandwidth as in multi-standard satellite gateways. The processing devices for channelization must be capable of operating at high speed. Hence, the front-end channelization is typically done using FPGAs, due to high speed requirements in dealing with the wideband input. The back-end (baseband) processing, which is performed on a per-channel basis, is done using digital signal processors or general purpose processors.

In SDR based mobile handsets, a reconfigurable channel filter is employed to perform the function of channelization with the filters being reconfigured to the current standard of operation. This is because only one-channel is of interest at a time in mobile handsets. However in SDR base-stations, several channels have to be extracted in parallel. An obvious approach to perform this task is to have a bank of channel filters for each channel. This approach is known as the per-channel approach. An alternative is to employ digital filter banks. The basics of these approaches are discussed in the following subsections.

2.4.1 Per-Channel Approach

The principle of per-channel (PC) approach is basically a parallel arrangement of many one-channel channelizers. Each one-channel channelizer performs the channelization process outlined in Fig. 2.3. Thus the PC approach based SDR channelizer with M received channels essentially consists of M narrowband digital filters, each extracting one-channel independently. The basic architecture of the PC approach is as shown in Fig. 2.5.

Figure 2.5 Per-channel approach.

In Fig. 2.5, the order of channelization is filtering ($H_0(z)$ to $H_{M-1}(z)$), digital down conversion (DDC), sample rate conversion (SRC) and finally baseband processing (BBP). Thus the filter, $H_0(z)$, is a low-pass filter and all other filters ($H_1(z)$ to $H_{M-1}(z)$) are band-pass filters. It is also possible to perform DDC followed by filtering and in this case, all the filters become low-pass filters (all filters are $H_0(z)$). Both these cases are discussed in the following sections.

2.4.1.1 Low-pass Filtering after Digital Down Conversion

In this case, first DDC is performed to the wideband signal at the output of ADC to bring the channel-of-interest to baseband. A typical channel arrangement in the frequency domain after DDC is shown in Fig. 2.6 [9]. There are many adjacent channels inside the received frequency band that have also been down converted. In order to select the channel-of-interest (shaded portion in Fig. 2.6), the adjacent channels have to be removed

by means of filtering. Since the channel-of-interest is located at baseband, a low-pass filter can be employed to perform channelization.

Figure 2.6 Channel-of-interest at baseband.

FIR filters are commonly employed for channel filtering because of their guaranteed stability and linear phase characteristics. Generally infinite impulse response (IIR) filters are seldom employed in wireless communication receivers because of their non-linear phase characteristics, which will cause distortion of the signal. Also coefficient quantization of IIR filters may affect the stability of the filters. FIR filters are absolutely stable because of its all-zero structure. However higher-order FIR filters are required for achieving sharp transition-band frequency responses due to stringent adjacent channel attenuation specification of wireless communication receivers. Hence FIR filters are generally more expensive to implement than their IIR counterparts. A more detailed analysis of design challenges of FIR filters and filter banks is presented in Chapter 3.

The FIR filters employed in PC approach need to operate at high speed because the channel filtering is done at the sampling rate of ADC. It is possible to reduce the complexity of the PC approach by employing polyphase decomposition of each of the filters and then shifting the SRC to the left of filtering operation. In this case, the channelization and SRC stages are combined into one stage which is termed as *multirate filtering* [11, 13]. The aliasing that occurs by SRC can be avoided by employing an anti-aliasing filter. In this case, the channel filter will act as the anti-aliasing filter, ensuring that the channel-of-interest is not aliased.

The Oversampling ratio (OSR) of a signal plays an important role in determining the filter order of these multirate filters. The OSR is given by (2.2).

$$OSR = \frac{f_s}{B} \quad (2.2)$$

where f_s is the sampling rate and B is the bandwidth of the channel-of-interest (i. e., the region to be kept free from aliasing). It should be noted that there is no restriction to how the frequencies are occupied outside the bandwidth B . This reflects a general view on oversampling known as generalized OSR (GOSR). The GOSR after SRC directly determines the relative bandwidth (compared to the f_s) of potential aliasing components, that have to be attenuated by the multirate filter. The higher the GOSR, the larger the transition-band width requirement of the multirate filter and the lower the complexity of the filter. Generally order of the filter is inversely proportional to the transition-band width of the filter.

The savings due to multirate filtering can be further improved by employing different filters for different stages in multirate processing. The GOSR is typically high in the first few stages, resulting in relatively large transition bands and narrow stop-bands. Thus comb filters such as cascaded-integrator comb (CIC) filters [14] can be employed for these first few stages. The CIC filters are multiplierless filters which can be implemented by employing only adders, subtractors and registers. These filters can be employed to downsample to as much as four times the Nyquist rate. By employing polyphase decomposition and thus multirate filtering, it is possible to relax the speed of filtering operation. But there are many constraints associated with polyphase decomposition while reconfiguring the filter which will be explained in Section 2.4.2.

2.4.1.2 Band-pass Filtering before Digital Down Conversion

The result of DDC followed by low-pass filtering can also be achieved by employing complex band-pass filtering followed by DDC. Though both solutions are equivalent in terms of their input-output behaviour, there are differences in implementations [9]. In the case of low-pass filtering after DDC, the impulse response is real-valued. Therefore, an FIR filter with N coefficients requires only $2N$ multiplications per output sample. The

multiplication rate is $4N$ multiplications per output sample when band-pass filtering is done before DDC. Therefore for the PC approach, low-pass filtering after DDC is preferred. On the other hand, complex band-pass filtering followed by DDC forms the basis of filter bank channelizers discussed in the next section.

The PC approach is a straightforward approach and hence relatively simple. But the main drawback is that, the number of branches of filtering-DDC-SRC is directly proportional to the number of channels need to be extracted. Hence the PC approach is not efficient when the number of channels is large and especially when the channels are of uniform bandwidth. The redundancy in filtering operation is not properly exploited in PC approach which led to the development of widely employed discrete Fourier transform filter banks (DFTFBs).

2.4.2 Filter Bank Approach

An alternative to the PC approach is the filter bank channelizer. In the filter bank approach, which is based on multirate signal processing theory, the channelizer can extract every channel between $[(-f_s/2), (f_s/2)]$, where f_s is the sampling frequency of the wideband ADC. In this approach, the complexity of the channelizer is independent of the number of received channels. The DFTFB, a classic example of filter bank approach, is developed as an efficient substitute for PC approach when the number of channels need to be extracted is large and these channels are of uniform bandwidth. The best example of such a scenario is extraction of multiple channels of a single communication standard. The main advantage of DFTFB is that, it can efficiently utilize the polyphase decomposition of filters. The derivation of DFTFB from PC approach can be explained based on Fig. 2.5 [15]. Consider the k^{th} channelization branch as shown in Fig. 2.7. Fig. 2.7(a) shows the whole process of band-pass filtering, $H_k(z)$, followed by DDC and SRC or downsampling by M . Notice that the only modulator (DDC) outputs not discarded by the SRC are those with time index $n=m.M$. For these outputs, the modulator has the value $e^{-j2\pi kn/M}=1$, and hence the modulator is not required. The resulting scheme eliminating DDC is shown in Fig. 2.7(b). Now it is possible to expand $H_k(z)$ in terms of M polyphase

branches as shown in Fig. 2.8 and it is possible to move the down sampling by M to the left of filtering operation. This can be explained with the help of following equations:

$$H_k(z) = \sum_{m=-\infty}^{\infty} h_k(n)z^{-n} = \sum_{l=0}^{M-1} z^{-l} \sum_{m=-\infty}^{\infty} h_k(mM+l)z^{-mM} \quad (2.3)$$

where $h_k(mM+l)$ represents the polyphase components of $H_k(z)$. Now expanding h_k in terms of the lowpass filter coefficient, h , i. e. substituting

$$h_k(mM+l) = h(mM+l)e^{j\frac{2\pi k(mM+l)}{M}} \quad (2.3) \text{ becomes,}$$

$$\begin{aligned} H_k(z) &= \sum_{l=0}^{M-1} z^{-l} \sum_{m=-\infty}^{\infty} \left(h(mM+l)e^{j\frac{2\pi k(mM+l)}{M}} \right) z^{-mM} \\ H_k(z) &= \sum_{l=0}^{M-1} z^{-l} \sum_{m=-\infty}^{\infty} \left(h(mM+l)e^{j\frac{2\pi kl}{M}} \right) z^{-mM} \\ H_k(z) &= \sum_{l=0}^{M-1} z^{-l} \left(\sum_{m=-\infty}^{\infty} h(mM+l)z^{-mM} \right) e^{j\frac{2\pi kl}{M}} \end{aligned} \quad (2.4)$$

Now replacing, $\sum_{m=-\infty}^{\infty} h(mM+l)z^{-mM}$ by $P_l(z^M)$, (2.4) becomes,

$$H_k(z) = \sum_{l=0}^{M-1} z^{-l} P_l(z^M) e^{j\frac{2\pi kl}{M}} \quad (2.5)$$

Figure 2.7 (a) Down conversion to baseband after band-pass filtering.

Figure 2.7 (b) Modified down conversion to baseband after band-pass filtering.

The expression (2.5) is shown in Fig. 2.8. In Fig. 2.8, $P_0(z)$ to $P_{M-1}(z)$ represent the polyphase components of a lowpass filter. The modified version of Fig. 2.8 by making use of the noble identity is shown in Fig. 2.9. It can be seen that, the dotted portion in Fig. 2.9 represents the inverse DFT (IDFT) operation for the k^{th} branch and hence can be replaced by IDFT. By employing M-point IDFT, all the channels (frequency bands) are obtained simultaneously in a DFT filter bank as shown in Fig. 2.10. Fig. 2.10 represents the DFT filter bank.

Figure 2.8 Schematic of k^{th} filter bank branch containing M polyphase branches.

Figure 2.9 Modified k^{th} filter bank branch containing M polyphase branches.

Figure 2.10 DFT filter bank.

Efficient implementations of a channelizer using DFTFBs are available in literature [9]. It can be seen from Fig. 2.10 that DFTFB can be realized by implementing one low-pass filter and a corresponding modulator such as IDFT. Thus instead of implementing N separate channel filters as in the case of PC approach, a single low-pass filter followed by DFT is only required (complexity of IDFT is same as that of DFT), provided all the channels have equal bandwidth. However, DFTFBs have following limitations for multi-standard SDR receiver applications [9, 10]:

1. DFTFBs cannot extract channels with different bandwidths. This is because DFTFBs are modulated filter banks with equal bandwidth of all band-pass filters. Therefore, for multi-standard receivers, distinct DFTFBs are required for each standard. Hence the complexity of a DFTFB increases linearly with the number of received standards.
2. Due to fixed channel stacking, the channels must be properly located for selecting them with the DFTFB. The channel stacking of a particular standard depends on the sample rate and the DFT size. To use the same DFTFB for another standard, the sample rate at the input of the DFTFB must be adapted accordingly. This requires additional SRCs, which would increase the complexity and cost of DFTFBs.
3. If the channel bandwidth is very small compared to wideband input signal (extremely narrowband channels), the prototype filter must be highly selective resulting in very high-order filter. As the order of the filter increases, the complexity increases linearly. Also the DFT size needs to be increased.

Reconfigurability is another key requirement in SDR receivers as discussed earlier. Ideally, the reconfigurability of the filter bank must be accomplished by reconfiguring the same prototype filter in the filter bank to process the signals of the new communication standard with the least possible overhead, instead of employing separate filter banks for each standard. However reconfiguration of DFTFB suffers from following overheads:

1. The prototype filter needs to be reconfigured. Generally DFTFB employs the polyphase decomposition. Hence reconfiguration can involve changing the number of polyphase branches and the number of coefficients in each polyphase branch which is a tedious and expensive task.
2. Downsampling factor needs to be changed. As a result of this, it is not appropriate to do down sampling before filtering. Hence for SDR receivers, the advantage of incorporating the digital down sampling to the left of filtering is not always feasible. Thus the prototype filter needs to operate at the same speed of ADCs.
3. The DFT needs to be reformulated according to the number of polyphase branches, which is also expensive.

For example, if switching from a 8-channel filter bank to 16-channel filter bank is considered, the number of polyphase branches need to be changed from 8 to 16 (first limitation of DFTFB), the downsampling factor needs to be adjusted from 8 to 16 (second limitation of DFTFB) and the 8-point DFT needs to be expanded to 16-point DFT.

2.4.3 Frequency Domain Filtering Approach

The frequency domain filtering (FDF) approach makes use of the properties of the fast Fourier transform (FFT) to simplify the DDC, channel filtering, and SRC functions [10]. The architecture for FDF approach is shown in Fig. 2.11.

Figure 2.11 Frequency domain filtering approach.

In this approach, the input data is first buffered into overlapping blocks, with an FFT performed on these blocks. The FFT bins extract frequency components for each channel-of-interest. These frequency components are then multiplied with FFT values of filter coefficients. This is because multiplication in frequency domain is equivalent to convolution in time domain. The inverse FFT is applied to the output of filter to retrieve the time domain values.

The FDF approach is very flexible when compared to the PC and the filter bank approaches because the parameters in frequency domain can be easily varied. The main drawback of FDF approach is the real time buffering of data and application of FFT on data coming at very high sampling rate. Also the delay imposed by FFT and inverse FFT can significantly delay the whole channelization process. Therefore the FDF approach is not an ideal approach for SDR receivers.

2.5 Summary

In this chapter, the basic concept of software defined radio (SDR) is introduced and the functionalities of different blocks in the SDR receiver architecture are discussed. The digital front end (DFE), especially different channelization approaches for the DFE, is given more emphasis in this chapter, as it is the main area of focus in this thesis. Three channelization approaches namely per-channel approach (PC), filter bank approach and frequency domain filtering (FDF) approach are discussed. The PC approach is the

simplest approach with a distinct channelizer used for each channel, thus having a complexity directly proportional to the number of channels. Filter bank approach makes use of the polyphase decomposition of filters and its complexity is independent of the number of received channels. But all the channels extracted by employing filter bank approach are of uniform bandwidth. Hence for the extraction of non-uniform bandwidth channels (channels of different wireless communication standards), distinct filter banks are required, which is an expensive task. FDF approach is the third channelization approach discussed where all channelization approaches are done in the frequency domain. However the frequency domain filtering is limited to low speed applications and hence not feasible for SDR applications. In the case of multiple channel extraction of single standard signal i. e., extraction of many channels of identical bandwidth, the complexity of PC approach is given by $N.L.f_s$, where N is the number of channels extracted, L is the length of filter employed or total length of filters employed in all the branches for PC approach and f_s is the sampling frequency. The complexity of DFTFB is only $L.f_s$ which is N times lower than PC approach. But in the case of SDR, multiple channels of multiple standards need to be extracted (extraction of multiple non-uniform bandwidths). In that case, the complexity of PC approach and DFTFB are $N_C.N_S.L.f_s$ and $N_S.L.f_s$ respectively, where N_C and N_S are the number of channels and number of standards respectively. Thus the complexity of these channelizers can be reduced further if (1) the length of filter, L , can be reduced and (2) the same filter bank can be reconfigured to the new standard (which will remove the term N_S from complexity equation). Thus there is a need for developing new filter bank architectures for SDR receivers. In the next chapter, the approaches presented in literature to minimize the hardware complexity of filters and filter banks in the channelizer of an SDR are discussed.

Chapter 3

Hardware-Efficient Implementation Approaches for Filters and Filter Banks

Channelization is a process where single, few or all radio channels from a wideband input signal are separated for further processing. The separation of single channel is usually done by down conversion followed by filtering and optional sample rate conversion. In the previous chapter, three basic channelization approaches called per-channel approach, filter bank approach and frequency domain filtering approach, have been discussed. Filtering is the most computationally intensive operation in channelizers. Therefore low complexity implementation of filters and filter banks is vital for reducing the over-all complexity of the channelizer. In this chapter, a review of different structural and hardware modifications of the three basic channelization approaches for low complexity implementation have been reviewed. Flexibility and reconfigurability are two other important considerations in software radio channelizers. This chapter also presents an account of different reconfigurable filter and filter bank architectures in literature.

3.1 Digital Filter Banks

Digital filter banks are the key components in any SDR base stations, when more than one channel need to be simultaneously extracted from the wideband input signal. The simplest and the easiest way of channelization is to use a separate filter for each channel to be extracted. This forms the basis for the PC approach discussed in the previous chapter. The PC approach is best suited when the number of channels to be extracted is less and the channels are of non-uniform bandwidth, i. e., extraction of multiple channels corresponding to multiple wireless communication standards. The complexity of PC approach increases linearly with the number of received channels. Therefore when the number of channels is more and these channels are of uniform bandwidth, uniform modulated filter banks are employed, where a single prototype filter is followed by a DFT. But the DFT filter bank (DFTFB) is most appropriate when all the channels to be

extracted are of uniform bandwidth, i. e., extraction of multiple channels corresponding to a single wireless communication standard. This is because for extraction of non-uniform bandwidth channels, distinct DFTFBs are required, which is expensive. The limitations of DFTFB have been discussed in Section 2.4.2 of Chapter 2. Several improvised approaches for the PC and DFTFB approaches are available in literature [9, 10, 16-21].

A filter bank based on modified Goertzel algorithm was proposed in [9] as a substitute to DFTFB. In the sequel, architecture in [9] is referred as Goertzel filter bank (GFB). In GFB, the DFT is replaced by a modified Goertzel algorithm which performs the modulation of the prototype low-pass frequency response to any centre frequency which is not possible using DFT. This will eliminate the limitation of fixed channel stacking associated with DFTFBs. But the GFB is also a type of modulated filter bank; hence it cannot extract channels with different bandwidths, as in the case of DFTFB. Also, extraction of narrow-band channels using GFB requires a very narrow passband prototype filter, which is expensive. The GFB approach requires IIR filter for the implementation of Goertzel algorithm and hence has stability constraints while reconfiguring the filter bank from one communication standard to another. Even though a theoretical introduction of GFB as a solution to some of the problems of DFTFB is given in [9], there is no consideration on the actual implementation complexities of GFB.

A channelizer based on a combination of polyphase filter bank and modified DFT (MDFT) modules have been proposed in [16]. The MDFT module performs real signal calculations instead of complex signal calculations and thus reduces computational complexity associated with the DFT operation. This is achieved by taking the real part of the DFT for the complex values. The MDFT module consists of one adder and two K -tap FIR filters, where K represents the number of polyphase branches of the prototype filter. Thus the over-all computational complexity of the filter bank is reduced when compared to conventional DFTFBs. However the channelizer in [16] is less flexible when compared to DFTFB. This is because the coefficients of the FIR filters in the MDFT module are dependent on the polyphase prototype filter. The reconfigurability of same filter bank for a new communication standard is also not achieved by the method in [16].

A multi-standard channelizer that has two stages of DFTFBs and efficient sample rate converters has been proposed in [17]. The front-end DFTFB has fixed number of channels, but the passband supports overlap with each other considerably resulting in easier isolation of channels with center frequencies of successive band-pass filters. The outputs of the front-end DFTFB are then fractionally decimated using SRCs. These decimated outputs are fed to the back-end DFTFB. Since the sample rate is considerably lowered in the back-end, the DFTFB at the back-end needs to operate only at low speed. Due to the reduced speed requirements, the back-end DFT can be repeatedly used to extract variable bandwidth channels. The drawback of the architecture in [17] is that, since the back-end DFTFB is employed for varying bandwidth channels, hardware optimization can be done only for the fixed front-end DFTFB. The back-end needs to be changed according to the new communication standard.

In [18], a channelizer based on modulated perfect reconstruction bank (MPRB) has been proposed. The MPRB approach in [18] consists of an analysis section and a synthesis section. By adding up the subband signals generated by the analysis section, wideband signals can be generated at the synthesis section. Thus the approach in [18] can be used for the channelization of signals of unequal bandwidths. However the bandwidths of the wideband signals generated by the synthesis section are integer multiples of the bandwidths of the subband signals generated by the analysis section. Thus the approach in [18] is not always appropriate for SDR signals, where the multiple communication standards have bandwidths which are not integer multiples of each other. A new method for the efficient design of the MPRB is also proposed in [18]. Also the approach in [18] consists of a polyphase prototype filter, IDFT analysis section and DFT synthesis section. Thus the computational complexity of the MPRB is double that of DFTFB. Also the implementation complexities associated with MPRB have not been considered in [18].

A pipelined frequency transform (PFT) based on the PC approach has been proposed in [19]. The basic PFT architecture consists of a binary tree of DDCs and SRCs, which splits the input signal frequency into a low and high frequency subbands, and then splits each half-band again until the last tree level extracts the desired channels. The PFT

architecture consisting of binary tree of DDCs and SRCs is shown in Fig. 3.1, where DDCs followed by SRCs are employed for dividing the input signal into a low-pass and high-pass bands with half sampling rate at the output. The main advantage of PFT approach over PC approach is that, the complexity of filtering can be reduced substantially taking advantage of half-band symmetry and reduced sampling rate at each output stage.

Figure 3.1 Architecture of PFT approach.

A reconfigurable channelizer using tree-structured quadrature mirror filter bank (TQMFB) has been proposed in [20]. The TQMFB consists of a tree of quadrature mirror filter banks, splitting the frequency band of input signal into high and low frequencies at quadrature frequency in each stage. Thus the TQMF approach in [20] is very similar to the PFT approach in [19]. The desired channel is obtained at an appropriate stage corresponding to the bandwidth of the channel-of-interest. The main drawback of the TQMFB is its delay in obtaining the desired output due to multistage filtering and decimation processes. The channelizers in [19] and [20] suffer from the drawback that they can only extract signals whose channel spacing are related by a factor of two. This constraint is imposed by the power-of-two subband stacking adopted in these architectures. Another problem of the methods in [19, 20] is that, as the subband decomposition tree extends, the wordlength of the output increases linearly and finite

wordlength multiplication would introduce truncation error, which propagates along the tree.

In the PFT approach, the problem with power-of-two subband stacking can be overcome by a tunable PFT (TPFT) architecture [21]. In the TPFT architecture, interleavers are introduced between different stages of PFT, which will enable the usage of intermediate outputs from different stages along the binary tree. These interleavers will help in fine tuning of channelization process and thus add more flexibility to the PFT architecture. Thus in TPFT, two levels of tuning are done, a coarse tuning at the PFT level and a fine tuning using another complex up/down converter assisted by a numerical controlled oscillator. However the implementation complexity of TPFT approach is much more than that of the PFT approach and thus not a very good candidate for wideband channelization.

Table 3.1
Comparison of Channelization Approaches

Parameter		PC Approach	DFTFB	PFT
Computational Complexity	For multiple uniform bandwidth channels	Poor	Excellent	Good
	For multiple non-uniform bandwidth channels	Poor	Poor	Poor
Silicon Cost		Poor	Excellent	Good
Initial Design Flexibility	Independent Channels	Yes	No	No
	Number of Channels	Selectable	2^N	2^N
Reconfigurability		Poor	Very Poor	Poor

A qualitative comparison of different channelization approaches is given in Table 3.1. The PC approach is compared to DFTFBs and PFT approaches based on four parameters. The approaches like frequency domain filtering discussed in Chapter 2 and TPFT approaches are not suitable for SDR, whereas the approaches in [9, 16-18] are modifications of DFTFB. In Table 3.1, the parameter 'computational complexity' means the number of multiplications associated with each method, which includes the multiplication in filtering, modulation (DFT) and digital down conversion. Previous works [9, 10, 22, 23] showed that when the number of uniform bandwidth channels to be extracted is more than two, the DFTFBs outperform the PC approach. It is also shown in [22] that an improvement in the filters of the PC approach can make it more efficient up to extraction

of 20 channels in some scenarios. The computational complexity of PFT method is less than the PC approach, but not lower than DFTFB. The parameter ‘silicon cost’ shows the actual implementation cost in FPGA. A drawback of this parameter is that it is platform dependent. It is shown in [19] that up to 256 channels, the silicon cost of PFT approach is comparable to DFTFB, but beyond 256 channels, DFTFB outperforms PFT approach. The third parameter is the ‘initial design flexibility’ which involves a combination of two factors: 1) ability to extract non-uniform bandwidth channels, and 2) the number of channels extracted. When ‘initial design flexibility’ is considered, the PC approach is obviously the best as all the extracted channels are independent, can have different bandwidths and can be non-uniformly and discontinuously distributed over the input frequency band. Neither PFT nor DFTFB is able to extract independent channels. The PFT has the limitation of power-of-two subband stacking and hence the number of extracted channels will be in powers of two i. e., 2^N , where N is an integer. Even though DFTFB has more flexibility, the most economical implementation of DFT has integer power-of-two bins and hence the number of extracted channels can be very similar to the PFT approach. On the other hand, the PC approach has no such problems. The parameter ‘reconfigurability’ represents the adaptation of channelization architecture to satisfy the new requirements with minimum overhead. As discussed in the previous sections, none of the existing approaches satisfy the reconfigurability requirement. It can be noted from Table 3.1 that the existing channelization approaches do not offer an efficient trade-off between complexity and reconfigurability.

The discussions so far have been on realization of reconfigurable low complexity filter banks for SDR receivers. But in order to achieve this, each channel filter needs to be implemented with low complexity and reconfigurability. In the following sections, different low complexity and reconfigurable implementations of filters are discussed and the problems with existing implementations in literature are highlighted.

3.2 Digital Filters for SDR Receivers

Digital filters in channelizers, commonly referred to as channel filters, extract frequency-divided channels from the digitized wideband input signal. Thus, the filters have to

attenuate adjacent-channel interferers and must meet the blocking characteristics specified by the relevant communication standard. Due to these stringent adjacent-channel attenuation specifications, higher-order channel filters are required in the receiver. As the order of the filter increases, the complexity of the filter also increases. Although this increased complexity may not be a major concern in base station receivers, when SDR migrates from base stations to resource constrained handsets, the high complexity would become a stumbling block. As the channel filters come directly after the ADC, they have to operate at high sampling rates and hence high speed. Reconfigurability is another important consideration in SDR receivers. Thus a channel filter having low power, high speed and reconfigurable characteristics needs to be investigated, which is a challenging task.

Digital filters can be either IIR or FIR filters. An IIR filter has a transfer function having both poles and zeros whereas for FIR filters, only zeros are present. Hence, IIR filters have a much sharper transition characteristics compared with the FIR filters for a given filter order. But IIR filters are difficult to be reconfigured due to the presence of poles. This is because IIR filters can become unstable because of the new pole values after reconfiguration [24]. But no such problems exist for FIR filters due to the absence of poles. FIR filters also have the advantage of linear phase property, which is an essential requirement in wireless communication receivers. For a given transition-band specification, FIR filter has higher-order compared to its IIR counterpart. The need of higher-order filters increases the cost of FIR filters.

The two most common implementation structures of FIR filters are direct form (DF) and transposed direct form (TDF) [25]. The DF structure shown in Fig. 3.2 is the most intuitive implementation of the time domain filter transfer function:

$$y[n] = \sum_{i=0}^{N-1} h[i].x[n-i] \quad (3.1)$$

where $x[i]$ is the input signal after i -th sampling delay, $y[n]$ is the output for the n th sample, N is the total number of taps of the filter, which is referred to as filter length and h_i is the coefficient value for the i -th tap.

Figure 3.2 Direct form FIR filter structure.

For the direct form N -tap FIR filter as shown in Fig. 3.2, N multiplications and $(N-1)$ additions are required. The critical path delay, t_{direct} , of the DF structure is given by:

$$t_{direct} = t_{mult} + (N-1)t_{RCA} \quad (3.2)$$

where t_{mult} is the critical path delay of the filter coefficient multiplier and t_{RCA} is the delay of a ripple carry adder. Thus the critical path length for DF FIR filter structure is directly proportional to the filter length, N .

As a result, the speed of operation of the direct form FIR filter is low especially when higher-order channel filters are required in SDRs. The critical path delay of DF structure can be further reduced by applying transposition theorem to obtain the TDF FIR filter as shown in Fig. 3.3 and equation (3.3).

$$y[n] = \sum_{i=0}^{N-1} x[i]h[n-i] \quad (3.3)$$

Figure 3.3 Transposed Direct form FIR filter structure.

The critical path delay of the TDF structure is given by:

$$t_{transpose} = t_{mult} + t_{RCA} \quad (3.4)$$

Since the critical path delay is independent of the number of taps, TDF has been the preferred structure for high speed, higher-order FIR filter implementations. Another advantage of the TDF structure is that the number of multiplications is reduced to $N/2$ as it exploits the symmetric property of FIR filter coefficients. Thus the number of multiplications in TDF structure is only half of that in DF structure, resulting in low power consumption. In conclusion, the TDF structure offers high speed operation at low power consumption compared to the DF structure. Therefore, this thesis focuses on the TDF structure of FIR filters. In the following sections, the complexity and reconfigurability issues in literature pertaining to FIR filter implementations are reviewed.

3.2.1 Complexity Analysis

Digital filters employed in mobile systems must be realized to consume less power and operate at high speed. Although programmable filters based on digital signal processor cores are available, they are not very efficient as they consume more power and operate at low speed. Hence dedicated FIR filter architectures have received great deal of attention in the last decade. The complexity of FIR filters is mainly dominated by coefficient multiplication operation. The methods that minimize the complexity of multipliers focus on decomposing multiplication operation into shift and add (subtract) operations. The two key metrics that determine the complexity of coefficient multiplications in FIR filters are the number of logical operators (LOs) and the logic depth (LD). LOs represent the adders required for computing the sum of partial products in the multiplier, which plays a major role in determining the area and power requirements of the filter circuit. LD is the number of adder steps in a maximal path of decomposed multiplications, which determines the speed of filtering operation. Therefore, the focus of low complexity FIR filter implementation algorithms is on reducing the number of LOs and LD in coefficient multipliers.

When the multipliers in filters are implemented using shift and add operations, the number of adders (subtractors) is directly proportional to the number of nonzero digits

present in the filter coefficients. Thus number systems with less number of nonzero digits are widely employed for the representation of filter coefficients compared to conventional binary representation [26-29]. Among such number representations, the canonical signed digit (CSD) representation is one of the most popular ones [26-28]. A number $b_0b_1b_2\dots b_{n-1}$ is said to be in CSD representation if each $b_i = -1, 0$ or $+1$ and no two consecutive b_i are nonzero. In [27], it was shown that any arbitrary n -bit 2's complement number (binary representation) can be represented in CSD form with no more than $(n+1)/2$ nonzero digits. Thus on an average, CSD representation offers a reduction of 33% of nonzero digits compared to the binary representation. Thus it is possible to implement a coefficient multiplier with 33% less number of adders in CSD form compared to binary representation [28]. In [29], a minimum signed digit (MSD) representation was proposed which resulted in the least number of nonzero digits. But the main drawback of the MSD representation is that, it can have more than one representation format for the same decimal or binary representation. The main problem with alternative number systems such as CSD and MSD compared to binary representation is that, multipliers realized using them will have subtraction operation due to negative bits, which will have constraints if the filters realized using these number representations need to be reconfigured. A more detailed analysis of the constraints due to negative bits while reconfiguring the filter is done in Chapter 5. Also if dynamic reconfigurability is required, on-the-fly conversion of filter coefficient binary values into CSD or MSD may not be possible in real time. In addition to this, there is additional memory requirement for storing the signed bits in case of CSD or MSD representations.

Once the filter coefficient is represented in a chosen number system, the next option will be to look for further hardware optimization by minimizing the number of nonzero bits while satisfying the filter characteristics such as passband ripple and stopband attenuations [30-32]. In [30], the filter coefficients are first scaled according to a local search before quantization process is performed resulting in a filter with much better frequency response characteristics compared to the original sum of power of two terms. In [31], the coefficient optimization is done as a polynomial optimization approach. The filter coefficients are scaled and normalized and then multiplied by a variable scale factor.

The algorithm in [31] continues in an iterative way and takes up lot of computation time. In [32], a combination of genetic algorithm (GA) and simulated annealing (SA) based scheme for coefficient optimization was proposed. The method in [32] uses the GA to search a population of the quantized filter coefficients of a digital filter for the optimal quantized filter. It retains the most accurate frequency characteristic of the original filter, which is either FIR filter or an IIR filter. The initial population in the GA is generated by binomial distributions, which are not used for the other GAs. An SA is also embedded in the GA search, which can support the GA to converge to the optimum in the early generations. But the method is only appropriate for short wordlength coefficients. In general, the coefficient quantization or optimization approaches [30-32] try to modify the filter coefficients in view of minimizing the number of nonzero bits for reducing the number of adders required for partial product addition. But the modification of coefficient values alter the frequency response characteristics and hence are generally lossy approaches.

Multiple constant multiplications (MCM) is a concept formulated in [33, 34] where, multiplication of one variable (input signal) with multiple constants (filter coefficients) is exploited to eliminate redundant multiplication operations. Since the filter coefficients are considered as constants, they can be analyzed for repetitive additions (subtractions) using identical bit patterns present in the CSD representation of coefficients called common subexpressions (CSs). The method in [34] was applicable to both DF and TDF structures of FIR filters. In [34], it was shown that when the filter coefficients are represented in CSD, certain 3-bit patterns like [1 0 1] and [1 0 -1] and their negated versions are repeated many times. These repetitive bit patterns (CSs) can result in repetitive or redundant additions and hence they need to be implemented only once. Thus in [34], an iterative algorithm was proposed to eliminate these redundant patterns by forming CSs only once. The coefficient $h_k = 0.10-1010101010010-1$ is used as an example to illustrate the method in [34]. In direct implementation (i.e. implementation of the multiplier using shifts and adds without using MCM concept), the filter tap output is

$$y_k = 2^{-1}x_1 - 2^{-3}x_1 + 2^{-5}x_1 + 2^{-7}x_1 + 2^{-9}x_1 + 2^{-11}x_1 + 2^{-14}x_1 - 2^{-16}x_1 \quad (3.5)$$

where x_1 is the input signal. It requires 7 LOs (adders and/or subtractors) to implement (3.5). The bit patterns $[1\ 0\ 1]$ and $[1\ 0\ -1]$ are repeated twice in h_k , which can be expressed as CSs, $x_2 = x_1 + 2^{-2}x_1$ and $x_3 = x_1 - 2^{-2}x_1$ respectively. Using CSs, the output (3.5) can be expressed as

$$y_k = 2^{-1}x_3 + 2^{-5}x_2 + 2^{-9}x_2 + 2^{-14}x_3 \quad (3.6)$$

Note that only 5 LOs (2 LOs for CSs, x_2 and x_3 , and 3 LOs for equation (3.6)) are needed for MCM implementation, which is a saving of 2 LOs when compared to direct implementation. Thus by employing the CSE technique, it has been shown that 38% reduction in the number of adders is achieved in MCM-based FIR filter structures [34].

In [35], Hartley proposed a method to eliminate the most commonly occurring 3-bit CSs such as $[1\ 0\ 1]$ and $[1\ 0\ -1]$ in the CSD representation of filter coefficients. As an additional criterion in the subexpression identification process, an estimation of a latch count improvement was also considered in [35]. The author investigated the timing and routing delay when more and more CSs are employed for adder reduction. In [35], CSs across the adjacent coefficients are also considered. Horizontal CSs (HCSs) are those CSs that occur within a coefficient and vertical CSs (VCSs) are those CSs that occur between adjacent coefficients. But as the HCSs $[1\ 0\ 1]$ and $[1\ 0\ -1]$ occurred more frequently, they were used for implementing the coefficient multiplier. The simulation results in [35] showed that the number of LOs can be reduced by 50% over conventional CSD implementations.

The method in [35] was modified in [36] using a mixed integer linear programming (MILP) design of FIR filters. The main aim in the method was to reduce the critical path length of the filter to obtain an LD almost identical to conventional CSD method. In [36], the CSD coefficient space is determined by the number of LOs used in the filters instead of the number of nonzero bits. Since the method in [36] employed coefficient optimization using MILP, it resulted in better frequency response characteristics than conventional methods. But this method consumed long optimization

time which makes it not suitable for channel filters, which would require on-the-fly reconfigurability.

A modification of the 3-bit CSE technique in [34] for identifying the “proper” CSs and to maximize the optimization impact was proposed in [37]. The method is a combination of an exhaustive search algorithm and steepest descent or greedy approach in selecting the “proper” CSs. This was done by formulating a matrix for the coefficients and then analyzing the coefficients for the most frequently occurring CSs. However the computation time of the method is directly proportional to the wordlength and hence takes longer time for large wordlengths.

In [38], the CSE techniques were applied to MSD representation of filter coefficients. The MSD representation is used because it provides a number of forms that have the minimal number of non-zero digits for a coefficient. This redundancy can lead to efficient filters if a proper MSD representation is selected for each coefficient. But there is no general rule for the proper selection of representation, which would impose constraints in decision making. In [38], there was no consideration for logic depth as the whole emphasis was for reduction of LOs. Also in [38], MSD representation is obtained from CSD representation of filter coefficients by a re-ordering mechanism. Hence there is a possibility of alteration of frequency response characteristics from that of the original filter.

The method in [35] was modified in [39] by developing a non-recursive signed CSE (NR-SCSE) algorithm that minimizes the LD. NR-SCSE allows the designer to overcome the problem of high LD by using each subexpression only once. NR-SCSE algorithm [39] gives two options to select the order of subexpressions. The first option allows to give a higher-order for the subexpressions, which will result in fewer numbers of LOs at the cost of an increased LD. The second option provides the user to select a lower-order for subexpressions, which will give the best reduction of LD at the cost of a slight increase in the number of LOs. But NR-SCSE algorithm uses subexpressions such as $[1\ 0\ 0\ 0\ 1]$ or higher-order in a non-recursive manner. As the order of the

subexpressions increase, the number of full adders for realizing each subexpression will also increase.

A novel CSE method was proposed in [40] for the implementation of FIR filters with a good trade-off between LOs and LDs. The method in [40] can be regarded as the first CSE approach resulting in minimum LOs with a reasonable shorter LD. This method can compute the results for a higher-order filter without consuming much time. A term known as C_n is defined in [40] which represents the absolute values of the CSD numbers that are waiting to be decomposed as a sum of some other CSD numbers. Instead of searching for matched bit patterns among coefficients as in conventional CSE techniques, the algorithm in [40] looks for pairs of available CSD numbers to synthesize the CSD numbers in C_n . The authors reported that this type of searching helps to find some unobvious relationship between CSD numbers which resulted in good reduction of LOs. But the rules and steps of the algorithm in [40] are too tedious to implement for higher-order channel filters in SDR receivers.

In general, the methods in [34-40] utilize the most HCSs and so, those methods can be referred to as horizontal common subexpression elimination (HCSE). In [41], a vertical common subexpression elimination (VCSE) technique has been proposed in which the authors exploit the fact that many VCSs exist, since adjacent filter coefficients of the FIR filters have similar patterns in the most significant bits. But the VCSE method is not very efficient when compared to HCSE for filters whose coefficients are of larger wordlengths. This is because VCSE depends on the statistical distribution of bit-patterns among the filter coefficients and is independent of bit patterns within the filter coefficients. The larger the wordlength of filter coefficients, the better the possibility of HCSs than VCSs. Hence the HCSE will be more efficient than the VCSE.

A comparison of the horizontal and vertical CSE techniques has been done in [42]. In [42], it has been shown that the HCSE technique offered better reduction of LOs and LD than VCSE in practical FIR filter implementations. The CSD-based CSE methods in [24-41] suffer from the drawback that the symmetry of FIR filter coefficients cannot be

completely exploited when the bits in VCSs are of opposite sign. As a result, additional LOs are required to obtain the symmetric part of the coefficients when more than one VCSs with bits of opposite sign exist [42]. This can be illustrated using the 4-tap symmetrical filter coefficient set given in Table 3.2. The numbers in the first row of Table 3.2 represent the number of bitwise right shifts. The coefficient multipliers are realized by employing vertical CSs of [1 1] and [1 -1] that occurs between h_0 and h_1 indicated inside the circles in Table 3.2. The VCS of [1 1] can be expressed as $x_4 = x_1 + x_1[-1]$ and [1 -1] as $x_5 = x_1 - x_1[-1]$, where $x_1[-1]$ represents the input x_1 delayed by one unit. Using VCSs x_4 and x_5 , the output y_0 corresponding to coefficients h_0 and h_1 can be expressed as

$$y_0 = 2^{-1}x_4 + 2^{-3}x_5 - 2^{-6}x_4 + 2^{-8}x_1 \quad (3.7)$$

The symmetric part output y_0 requires 3 LOs for (3.7) and 2 LOs for x_4 and x_5 , and thus a total of 5 LOs. Direct implementation (i. e., implementation without any CSE) requires (N_b-1) LOs, where N_b is the number of non-zero bits in the symmetric half coefficients. Thus for the example in Table 3.2, 6 LOs are needed in direct implementation as N_b is 7.

Table3.2
VCSE in FIR filter Coefficients

	1	2	3	4	5	6	7	8
h_0	1	0	1	0	0	-1	0	1
h_1	1	0	-1	0	0	-1	0	0
$h_2=h_1$	1	0	-1	0	0	-1	0	0
$h_3=h_0$	1	0	1	0	0	-1	0	1

In general, VCSE offers reduction of LOs due to the occurrence of many VCSs in the coefficient set. However, the CSD-based VCSE fails to completely exploit the symmetry of coefficients in implementing the filter. For the symmetric part of (3.7), using the same VCSs, the expression for output of coefficients h_2 and h_3 is given by

$$y_2 = 2^{-1}x_4[-2] - 2^{-3}x_5[-2] - 2^{-6}x_4[-2] + 2^{-8}x_1[-3] \quad (3.8)$$

There are two constraints related to symmetry exploitation in VCSE in this case. First, the differences in signs of the second terms in (3.7) and (3.8) and second, the delay

differences for the fourth terms in (3.7) and (3.8). Hence, the expression (3.8) cannot be directly obtained from its symmetric part (3.7) by a simple delay operation; instead extra LOs are needed for compensating the sign and delay differences. This requirement of extra LOs poses constraints in reducing the number of LOs in CSD-based VCSE method [42].

A different approach in which the whole multiplier block (MB) is synthesized using graph synthesis algorithms has been adopted in [43-46]. The Bull-Horrock's (BH) algorithm [43] used decimal representation of coefficients to optimize the logic complexity. In this method, the synthesis of filter coefficients is represented by a graph whereby partial sums called the fundamentals are symbolically encapsulated in the vertices of the graph and the shift amounts of the partial sums are annotated on the edges. Graph dependence (GD) algorithms involve the synthesis of a set of minimal cardinality connected graphs from the unity source to the sinks, which are coefficients to be synthesized. The fundamentals are generated one at a time depending on the previously generated fundamentals. But the BH algorithm used only additions for the generation of fundamentals, which resulted in high values of new fundamentals skipping the intermediate values. Also full exploitation of redundancy in the filter coefficients is not done in [43].

In order to overcome the limitations of the BH method, two new algorithms namely modified BH (BHM) and reduced adder graph-n-dimensional (RAG-n), have been proposed in [44]. In RAG-n algorithm, the coefficient that requires least number of adders is synthesized first. But in BHM, the synthesis is done in a previously defined manner. Both BHM and RAG-n made use of the subtractors which resulted in low valued fundamentals. The GD algorithms produced better results in terms of number of LOs required to implement the coefficient multipliers. But the RAG-n algorithm makes use of look up tables (LUTs), and hence there is an upper limit for the coefficient wordlength. The BHM algorithm consumes longer computational time for the optimization process and hence it is not attractive for higher-order filters. Generally GD algorithms synthesize coefficients sequentially, and thus greedily reduce the hardware complexity with little or

no regard to the adjacent coefficients. This will result in inadequate exploitation of redundancies among the coefficients. Also the GD algorithms [43, 44] resulted in increased LD which will increase the filter delay.

In [45], a modification of GD algorithm proposed in [44] for a good trade-off between LOs and LD has been proposed. Three new delay reduction methods such as tree reduction, limited selection method and minimum adder-step method are also reviewed in [45]. These three reduction methods are combined with BHM and RAG-n to form two new algorithms which offered better trade-off between LOs and LDs. But the inherent drawbacks of BHM and RAG-n have not been overcome in [45]. The authors show that the proposed algorithms can reduce the delay of the MB by slight increase of complexity.

The contention resolution algorithm (CRA) proposed in [46] also focused on the reduction of LD as the primary aim. CRA provides a leeway to break away from the local minimum and the flexibility of varying optimization options through a new admissibility graph. It manages three-bit CSs and aims at achieving the minimal LD as the primary goal. But the CRA method resulted in an increase in the number of LOs in many cases.

A new GD algorithm was proposed in [47] to minimize LOs. The main idea behind the method in [47] is to use a better heuristic approach to synthesize intermediate fundamentals to jointly optimize for all target coefficients. But the method in [47] is computationally more expensive than [44] as it explores a very large space of possible intermediate fundamentals. Also the method in [47] resulted in longer LD. Also [47] has not addressed the problem of realizing higher-order filters as the maximum number of coefficients was restricted to 100.

A general problem with GD algorithms [43-47] is their large computation time for optimization and increased LD which make them less attractive for high speed filters. From the review studies, it was found that, the GD algorithms [43-47] offer average LO reduction of 5-10% over CSE methods [33-42], but the LDs of coefficient multipliers realized using the GD algorithms are 40-60% more than CSE techniques. Since the

concern is more on higher-order high speed channel filters in SDRs, the work in this thesis has been concentrated more on CSE techniques.

3.2.2 Adder Complexity Analysis

Even though minimizing the number of LOs reduces the multiplier complexity, the actual cost of the multiplier is determined by the number of full adders (FAs) required for the realization of each adder in the multiplier, called adder-width. Methods for realizing filters with minimum FAs were proposed in [48-50]. In [48], a super-subexpression elimination (SSE) technique was used to reduce the number of FAs. The SSE technique combined the horizontal CSs and the vertical CSs to form super subexpressions (SSs) and reduced the filter complexity by eliminating redundant SSs. But some of the SSs discussed in [48] do not occur frequently in practical FIR filters and hence the use of such SSs would increase the routing complexity of the filter circuit. Moreover the algorithm in [48] produced good FA reductions only when the coefficient wordlength is larger. Also, the number of FAs needed in [48] is dependent on the shift amount and wordlength of the coefficients.

A method which reduces the adder complexity by removing the dependency on the shift amount has been proposed in [49]. The technique in [49] involved modification of the bit patterns of the original coefficients to reduce the FAs compared to [48]. This can be achieved by converting all the subtraction operations if possible into addition operations and then hardwiring the portion which doesn't necessarily require any adder. But the method in [49] has a strong dependency on the distribution of coefficient bits especially signed bits. Moreover [49] has a constraint in reducing the number of FAs due to the presence of the signed (negative) bits in CSD representation. For example, modifying the bit patterns such as (0.10-100-101) to (-0.-1010010-1) would increase the complexity of the filter.

An efficient coefficient-partitioning method (CPM) was proposed in [50], which offered better FA reductions for any wordlengths and hence it is a more general solution than the method in [48]. The basic idea of CPM [50] is to reduce the range of each

operand and thus to minimize the adder width. Let x_1 is an 8-bit quantized input signal. The output expression y for the 20-bit coefficient $h = 0.00010000100000100010$ is:

$$y = 2^{-4}x_1 + 2^{-9}x_1 + 2^{-15}x_1 + 2^{-19}x_1 \quad (3.9)$$

After applying CPM [53], expression (3.9) becomes

$$y = 2^{-4}(x_1 + 2^{-5}x_1 + 2^{-11}(x_1 + 2^{-4}x_1)) \quad (3.10)$$

Figure 3.4 Filter tap implementation of (3.9) and (3.10).

The implementations of (3.9) and (3.10) are shown in Fig. 3.4. The numerals adjacent to the datapath in Fig. 3.4 represent the number of bitwise right shifts. The adder width is shown inside brackets alongside each adder in Fig. 3.4. As shown in Fig. 3.4 (a), for direct implementation of (3.9), 75 FAs are required. For implementing the CPM optimized expression (3.10), only 52 FAs are required. The reduction of FAs in CPM is achieved by performing addition operations prior to shift operations. However it must be noted that the number of FAs obtained by CPM [50] is dependent on the shift amount and wordlength of the coefficients.

In this section, many approaches for the implementation of low complexity FIR filters have been discussed. In addition to low complexity, reconfigurability is also an important requirement of channel filters in SDR receivers. In the next section, some of the existing reconfigurable FIR filter architectures are reviewed.

3.2.3 Filter Reconfigurability

Several implementation approaches for reconfigurable FIR filters have been proposed in literature [51-60]. These designs include either a fully programmable Multiply-Accumulate (MAC) based filter processor or dedicated architectures where the filter coefficients can be stored in registers. The architecture of a filter processor consists of a datapath with a single MAC unit, data and program memories, and a control unit [51, 52]. The datapath includes a 16-bit adder (subtractor), a multiplier and a 32-bit accumulator. The performance of the processor is mainly restricted by the delay of this datapath, more specifically that of the multiplier. The main disadvantage of the filter processors is that the area and power requirements are significantly large.

In [53], a comparison of power consumption was done for the performance of speech based algorithms on dedicated architectures and general purpose processors. It was shown that the power consumption for a general-purpose processor can be a factor of four times more than dedicated architectures for a complex algorithm [53].

The works in [54-60] present reconfigurable FIR filter architectures. In [54], a CSD-based digit reconfigurable FIR filter architecture was proposed. This architecture was independent of the number of taps because the number of taps and non-zero digits in each tap were arbitrarily assigned. The intention of the authors was to reduce the wordlength of the coefficients and thus the filter complexity without affecting the filter performance. But the architecture in [54] demanded huge hardware resources and this makes the method infeasible for resource constrained SDRs. Also the architecture in [54] is digit based and hence the speed of operation is low.

In [55], a high speed and programmable CSD-based FIR filter was proposed. The filter architecture consisted of a programmable CSD-based Booth encoding scheme and partial product Wallace adder tree. The final adder was a carry look ahead adder. This method resulted in high speed but at the cost of high power consumption. The aim in [55] was solely to design a high speed reconfigurable filter and no consideration was given for reducing power consumption.

A high speed and programmable FIR filter based on polyphase decomposition was proposed in [56]. However this method used the built-in block multipliers of Virtex II FPGA and there was no consideration for the circuit level complexity reduction of the FIR filter. The method was just an implementation of polyphase structure based FIR filter employing reconfigurable features of Virtex II FPGA.

In [57], the concept of reconfigurable multiplier block (ReMB) was used. The ReMB consisted of a coefficient store and a general-purpose multiplier. The ReMB will generate all the coefficient products and a multiplexer will select the required ones depending on the input. It was shown that, by pushing the multiplexer deep into the multiplier block architecture, the redundancy can be reduced. The resulting specialized multiplier design is more efficient in terms of area and computational complexity compared to the general-purpose multiplier plus the coefficient store [57]. But the ReMB proposed in [57] has its area, power and speed dependent on the filter-length making them inappropriate for higher-order FIR filters.

In [58], a multiplexed multiple constant multiplication (MMCM) approach was proposed. This method considers the coefficient set as constant and uses the GD algorithms for reducing redundancy. But this method follows a directed acyclic graph (DAG) structure which will result in long LD and thus lower speed of operation. Also the area of the architecture linearly increases with the filter length as in [57] and filters with filter-length above 40 are infeasible.

In [59], the common DSP operations such as filtering and matrix multiplication were identified and expressed as vector scaling operations. In order to apply vector scaling, simple number decomposition strategies were identified. The idea was to precompute the values such as x , $3x$, $5x$, $7x$, $9x$, $11x$, $13x$, and $15x$, where x is the input signal and then reuse these precomputations efficiently using multiplexers. The presence of multiplexers gave the option of adaptive computing for the method in [59]. In [60], the method in [59] was modified and efficient circuit-level techniques that use a new carry-

select adder and conditional capture flip-flop (CCFF) to further reduce power and improve performance.

The architectures in [54-60] are appropriate only for lower-order filters and hence not suitable for channel filters in SDR receivers, which are normally higher-order filters due to stringent adjacent channel attenuation specifications. From the review studies, it was found that, integration of low complexity and reconfigurability into a hardware architecture for FIR filters and filter banks have been hardly addressed in literature.

3.3 Summary

In this chapter, various low complexity and reconfigurable implementations of filters and filter banks have been reviewed. Various basic channelization approaches such as per-channel approach and filter bank approaches have been studied and analyzed. From a qualitative comparison of channelization approaches, it was found that none of the existing approaches offer an efficient trade-off between low complexity and reconfigurability. Hence new efficient reconfigurable filter bank architectures need to be investigated. In order to reduce the complexity of filter banks, the complexity associated with each filter need to be reduced. Therefore efficient and reconfigurable implementations of finite impulse response filters used in filter banks have been discussed. The complexity of filtering operation is dominated by the coefficient multiplication operation. It was found that common subexpression elimination (CSE) techniques [33-42] and graph dependence (GD) algorithms [43-47] have been best suited for reducing coefficient multiplier complexity since they consider coefficient multiplication problem as a multiple constant multiplication (MCM) problem with input signal as variable and coefficients as constants. GD algorithms resulted in better reduction of adder complexity in coefficient multipliers with significant increase in delay compared to CSE techniques. The channel filters in software defined radio (SDR) receivers come directly after analog-to-digital converter (ADC) and hence need to operate at high speed. Therefore CSE techniques are appropriate for high speed channel filters in SDR receivers. But the CSE techniques in literature are based on signed digit representation of coefficients like canonical signed digit or minimal signed digit which have inherent

disadvantages when it comes to reconfiguration of filters in SDR. Also none of these CSE techniques have been applied to higher-order channel filters in SDR receivers. Hence a new CSE technique needs to be investigated which will be more efficient for higher-order reconfigurable channel filters in SDR receivers. In this thesis, the focus is on implementation of reconfigurable low complexity filters and filter banks for SDR channelizers, which will be presented in the subsequent chapters.

Chapter 4

Low Complexity Channel Filters using Binary Subexpression Elimination Algorithm

The channel filters employed in software defined radio channelizers must be of higher-order and realized to consume less power and operate at high speed. Design of dedicated finite impulse response (FIR) filter architectures have received great deal of attention in the last decade because of significant hardware optimization possibilities. The number of adders (subtractors) used to implement the coefficient multipliers determines the complexity of the FIR filters. It is well known that common subexpression elimination (CSE) methods based on canonical signed digit (CSD) coefficients reduce the number of adders required in the multipliers of FIR filters. A new CSE algorithm using binary representation of coefficients known as binary subexpression elimination (BSE) is presented in this chapter for implementing higher-order FIR filters with fewer number of adders than conventional CSD-based CSE methods. It is also shown that the CSE method is more efficient in reducing the number of adders needed to realize the multipliers when the filter coefficients are represented in the binary form. The proposed BSE algorithm consists of binary horizontal subexpression elimination, binary super subexpression elimination and binary vertical subexpression elimination. The proposed BSE algorithm offers a good trade-off between the number of adders and logic depth (critical path length) in comparison with methods in [34]-[42] especially for higher-order filters.

4.1 Binary Subexpression Elimination Algorithm

In this section, a CSE method based on binary representation of the filter coefficients called binary subexpression elimination is presented. The basic idea of BSE is to search and eliminate the redundant horizontal, vertical and super subexpressions that exist in the binary representation of filter coefficients. To the best of author's knowledge, most of the CSE methods in literature make use of the CSD representation of the filter coefficients. This is because the number of non-zero bits in the CSD representation is fewer than that

in corresponding binary representation. In [28], it is shown that the number of non-zero digits is reduced by 33% for CSD representation compared to normal 2's complement form. As the number of non-zero bits in CSD is less, only fewer adders are needed to realize the coefficient multiplier compared to binary representation. However it should also be noted that as the number of non-zero bits are minimum, the potential of the CSD-based CSE technique to reduce the number of adders by forming CSs is less than that of binary.

In this section, the aim is to relate the dependence of the cost of CSE method on three factors – the total number of non-zero bits in the coefficient set, the number of CSs that can be formed from the non-zero bits and the number of unpaired bits (bits that do not form CSs). The impact of these factors on the number of LOs needed to implement the coefficient multipliers have been analyzed statistically. Let N_{nz} represents the number of non-zero bits before the application of CSE technique, N_{cs} represents the number of CSs and N_{up} represents the number of unpaired bits (1s and -1s) after the application of CSE technique. Let the number of LOs required in CSE technique, N_{LO} ,

$$N_{LO} = (\pm\alpha \times N_{nz} \pm \beta \times N_{cs} \pm \gamma \times N_{up}) \quad (4.1)$$

where α , β and γ are the weights of N_{nz} , N_{cs} and N_{up} respectively which shows the amount of dependencies of N_{nz} , N_{cs} and N_{up} on the total cost N_{LO} . A statistical analysis has been done on coefficients for FIR filters of different lengths (20, 50, 80, 120, 200, 400 and 800 taps), and wordlengths of 12, 16, 20 and 24 bits for obtaining the weights α , β and γ in (4.1). The analysis was made for filters with different passband (ω_p) and stopband (ω_s) frequency specifications given by (a) $\omega_p = 0.1\pi$, $\omega_s = 0.12\pi$, (b) $\omega_p = 0.15\pi$, $\omega_s = 0.25\pi$, (c) $\omega_p = 0.2\pi$, $\omega_s = 0.22\pi$ and (d) $\omega_p = 0.2\pi$, $\omega_s = 0.3\pi$ respectively. In this analysis, narrow transition-band ($\omega_p = 0.1\pi$, $\omega_s = 0.12\pi$ and $\omega_p = 0.2\pi$, $\omega_s = 0.22\pi$) and moderately wider transition-band ($\omega_p = 0.15\pi$, $\omega_s = 0.25\pi$ and $\omega_p = 0.2\pi$, $\omega_s = 0.3\pi$) filters are considered. i. e., the first coefficient set has the specifications of $\omega_p = 0.1\pi$, $\omega_s = 0.12\pi$ with 20 taps and wordlength of 12 bits and so on. This is because the higher-order filters designed using the proposed BSE technique are mainly intended for SDR receivers which have stringent adjacent channel attenuation specifications and correspondingly

narrow transition bands. All the filters were designed using Park-McClellan algorithm. In this analysis, the CSs [1 0 1], [1 0 -1], [1 0 0 1] and [1 0 0 -1] and their negated versions were considered. The CSE technique employed in [35] was employed for finding CSs. Expression (4.1) consists of three unknown weights and hence by using three set of equations for each coefficient set (specification), the weights α , β and γ are obtained. Thus based on the different α , β and γ obtained for different coefficient sets, the average values of α , β and γ are obtained as 0.2345, -0.6643 and 4.0487 respectively.

Thus (4.1) can be written as

$$N_{LO} \approx (0.2345 \times N_{nz} - 0.6643 \times N_{cs} + 4.0487 \times N_{up}) \quad (4.2)$$

It can be noted from (4.2) that, the weight of N_{up} is substantially larger compared to the weights of N_{nz} and N_{cs} . Therefore the number of LOs is largely dependent on N_{up} . Fig. 4.1 shows the comparison of average N_{up} values for the binary and CSD filter coefficient representations for the same specifications mentioned above for different filter lengths and for 16-bit coefficient wordlength. From Fig. 4.1, in the case of CSD representation, the values of N_{up} are found on the higher side compared to binary representation of filter coefficients especially for higher-order filters and the average reduction of N_{up} for binary filter coefficients over CSD values is 67%. In order to clearly show the impact of N_{up} on the number of LOs, a factor known as the degree of sparseness (DoS) is introduced.

Figure 4.1 Average values of N_{up} for binary and CSD representation of filter coefficients.

Definition [Degree of Sparseness (DoS)]: The distance between non-zero bits (digits) in

the representation of filter coefficients which is greater than the maximum distance between the non-zero bits of the used CSs is known as DoS.

The DoS is the measure of those bits which will not come under the coverage of CSE techniques. Thus N_{up} is directly proportional to the DoS. DoS can be illustrated with the coefficient, $h_k=0.100010100001$. In this case considering the CS $[1\ 0\ 1]$, the maximum distance between the non-zero bits (maximum number of zeros between two non-zero bits) of the used CSs is 1. There are two non-zero bits (the 1s at the LSB and the MSB) which are not under the coverage of the CS and the DoS of h_k is 10, which is equal to the number of bits between the first and last '1's of h_k . To determine the DoS values, a statistical analysis of coefficients has been done for the same filter specifications mentioned earlier. The average of the DoS values for different filter lengths and 16-bit coefficient wordlength for binary and CSD representations are shown in Fig. 4.2. The binary CSs $[0\ 1\ 1]$, $[1\ 0\ 1]$, $[1\ 1\ 0]$, $[1\ 1\ 1]$ and $[1\ 0\ 0\ 1]$ and that of CSD-CSs $[1\ 0\ 1]$, $[1\ 0\ -1]$, $[1\ 0\ 0\ 1]$, $[1\ 0\ 0\ -1]$ and their negated versions have been analyzed. The CSE algorithm in [35] has been employed for the analysis purpose so that the comparison is fair. It can be noted that the trend of DoS shown in Fig. 4.2 has a close resemblance to the occurrence of unpaired bits (N_{up}) shown in Fig. 4.1. Also note that DoS is larger for CSD representation of coefficients compared to binary representation.

Figure 4.2 Average values of DoS for binary and CSD representation of filter coefficients for different filter lengths.

Figure 4.3 Average values of DoS for binary and CSD representation of filter coefficients for different coefficient wordlengths.

From Fig. 4.2, on an average, the DoS for CSD representation is 35.6% more over binary representation. Fig. 4.3 shows the average of the DoS values for different wordlengths of filter coefficients for a 120-tap FIR filter. In this case also, the DoS is more for CSD representation and the average reduction of DoS for binary over CSD is 35.6%.

A statistical analysis for different FIR filters whose specifications are given earlier has been again done to determine N_{cs} for binary and CSD representations of filter coefficients. Fig. 4.4 shows the average values of N_{cs} for different filter lengths. From the Fig. 4.4, it is evident that the number of CSs is almost same for both binary and CSD. The average difference in the number of common subexpressions, N_{cs} , is found to be 6.5% which is not a significantly high value. From the analysis of N_{nz} , N_{cs} and N_{up} , it is evident that the number of LOs is heavily dependent on the values of N_{up} . In order to provide a better insight about the relationship between the values of N_{cs} and N_{up} , another factor known as the frequency of occurrence of the CSs is introduced.

Figure 4.4 Average values of N_{cs} for binary and CSD representation of filter coefficients.

Definition [Frequency of Occurrence]: Frequency of occurrence of a CS is defined as the number of times the same CS is being reused or repeated in the filter coefficients.

Thus high frequency of occurrences of CSs means, most of the bits in the filter coefficients will be grouped as CSs which will result in fewer numbers of unpaired bits (N_{up}). It must be noted that frequency of occurrences of CSs is not the same as the number of CSs. Frequency of occurrence is a measure of how effectively the CSE technique will work on the given set of coefficients by grouping the 1s and -1s into CSs resulting in fewer numbers of unpaired bits. This can be illustrated using the example of a coefficient $h_k=0.01111000$. The frequency of occurrences of [1 1] is 2, [1 0 1] is 2 (h_k can be obtained by combining [1 0 1] and its shifted version), [1 1 1] is 1 and [1001] is 1 (h_k can be obtained by combining [1 0 0 1] and [1 1]). If the CSs that has highest frequency of occurrences (which is [1 1] or [1 0 1] in this case) are selected, then N_{up} would become zero. On the other hand if the CSs with low frequency of occurrences ([1 1 1] or [1 0 0 1] in this example) is chosen, N_{up} would be one. Therefore, CSs with higher frequency of occurrences would result in smaller N_{up} and correspondingly fewer numbers of adders (LOs as explained in Chapter 3) to implement the coefficient multiplier.

The frequency of occurrences of binary CSs (BCSs) [0 1 1], [1 0 1], [1 1 0], [1 1 1] and [1 0 0 1] and that of CSD-CSs [1 0 1], [1 0 -1], [1 0 0 1], [1 0 0 -1] and their

negated versions have been analyzed for a large number of FIR filters. The frequency of occurrences of CSs in the CSD and binary representations of FIR filters for different specifications, employed earlier for analyzing expression (4.1), have been compared. The average values of the frequency of occurrences of the CSs for the above filter specifications are shown in Figures 4.5 and 4.6.

Figure 4.5 Average frequencies of occurrences of CSs in 16-bit coefficients of the example filters.

Figure 4.6 Average frequencies of occurrences of CSs in the 120-tap filters for different wordlengths.

Fig. 4.5 shows the comparison of the frequency of CSs that occur in the CSD and binary

coefficients of 16-bit wordlength, for the filters having different lengths (20 to 800 taps). Note that the frequency of occurrences of BCSs are much more than the CSD-based CSs for the same filter specifications. Among the CSD-based CSs, the occurrences of $[1\ 0\ 0\ 1]$ and $[1\ 0\ 0\ -1]$ are comparatively much less than $[1\ 0\ 1]$ and $[1\ 0\ -1]$ which show that with limited number of non-zero bits in CSD representation, the scope of higher-order CSs such as $[1\ 0\ 0\ 1]$, $[1\ 0\ 0\ 0\ 1]$, etc., and their negated versions is less. For BCSs, the occurrence of $[1\ 1\ 1]$ is relatively less than other BCSs. From Fig. 4.5, it can be concluded that the possibility of finding a binary common subexpression is 2.95 times more than a CSD common subexpression for 16-bit filter coefficients. Fig. 4.6 shows the comparison of the number of CSs that occur in CSD and binary coefficients of a 120-tap filter for different coefficient wordlengths of 12, 16, 20 and 24 bits. Note that the frequency of BCSs is considerably more compared to CSD-CSs. It can be concluded from Fig. 4.6 that, the possibility of finding a binary common subexpression is 2.73 times more than a CSD common subexpression for same wordlengths.

The number of non-zero bits (N_{nz}) in binary representation in the worst-case is only 2 times more than that in CSD representation. As the frequency of occurrences of BCSs is more than CSD-CSs in filter coefficients by 2.84 (reducing the number of non-zero operands in the filter coefficients by grouping the '1's into CSs), the BSE technique will compensate for the increased number of '1's compared to CSD-based CSE techniques. Thus the number of LOs is mainly dependent on N_{up} . From Figures 4.1, 4.2 and 4.3, it is evident that N_{up} is more for CSD than binary representation of filter coefficients. Based on the above statistical analysis, it can be concluded that the proposed BSE method is efficient for reducing the LOs compared to CSD-based CSE techniques.

The proposed BSE algorithm consists of elimination of horizontal, vertical and super subexpressions as explained below.

4.1.1 Binary Horizontal Subexpression Elimination (BHSE)

BHSE deals with the elimination of redundant binary horizontal common subexpressions (BHCSs) that occur within a coefficient. The number of BHCSs that can be formed in an

n -bit binary number is

$$2^n - (n+1) \quad (4.3)$$

For example, a 3-bit binary number ($n=3$) can form 4 BHCSs, which are [0 1 1], [1 0 1], [1 1 0] and [1 1 1]. Note that other BHCSs such as [0 0 1], [0 1 0] and [1 0 0] do not require any adder for implementation since they have only one nonzero bit. These BHCSs can be expressed as:

$$[0 \ 1 \ 1] = x_6 = 2^{-1} x_1 + 2^{-2} x_1 \quad (4.4)$$

$$[1 \ 0 \ 1] = x_7 = x_1 + 2^{-2} x_1 \quad (4.5)$$

$$[1 \ 1 \ 0] = x_8 = x_1 + 2^{-1} x_1 \quad (4.6)$$

$$[1 \ 1 \ 1] = x_9 = x_1 + 2^{-1} x_1 + 2^{-2} x_1 \quad (4.7)$$

where x_1 is the quantized input signal. A straightforward realization of the BHCSs (4.4) to (4.7) would require 5 LOs. However, x_6 can be obtained from x_8 by a simple right shift operation (without using any LOs) as follows.

$$x_6 = 2^{-1} x_1 + 2^{-2} x_1 = 2^{-1} (x_1 + 2^{-1} x_1) = 2^{-1} x_8 \quad (4.8)$$

Also, x_9 can be obtained from x_8 using an LO:

$$x_9 = x_1 + 2^{-1} x_1 + 2^{-2} x_1 = x_8 + 2^{-2} x_1 \quad (4.9)$$

Thus 2 LOs can be saved and only a total of 3 LOs are needed to realize the BHCSs (4.4)-(4.7). In general, the number of LOs, N_a , required for realizing all the possible n -bit binary subexpressions is

$$N_a = 2^{n-1} - 1 \quad (4.10)$$

In the proposed BHSE method, the 3-bit BHCSs (4.5)-(4.7) and a 4-bit BHCS [1 0 0 1] = $x_{10} = x_1 + 2^{-3} x_1$ have been chosen as it produced the best LO reductions. Statistically, these 5 BHCSs were found most commonly occurring in filter coefficients as shown in Figures 4.5 and 4.6. It was observed that the LO reductions are not significant when other higher-order 4-bit, 5-bit and 6-bit BHCSs are used. Note that as the bit length of the BCS increases, the number of LOs required to realize all the possible subexpressions will also increase. The frequency of higher-order BHCSs, i.e., 4-bit, 5-bit and 6-bit BHCSs have

also been analyzed with the same filter specifications used for the Figures 4.1 and 4.2. Note that higher-order BHCSs are basically a combination of the 3-bit BHCSs (Super sets of 3-bit BHCSs). It was observed that the frequency of the higher-order BHCSs in filter coefficients is very less. Therefore, higher-order BHCSs do not offer considerable reduction of LOs. Moreover, the use of higher-order BHCSs will have adverse effect on the LD. For a binary tree-structured addition scheme, the LD of an n -bit BHCS is given by $\log_2 n$ [48]. Thus as n increases the LD will also increase and correspondingly the delay of the filter will also increase. Due to the above reasons, higher-order BHCS are omitted in the proposed BSE method.

4.1.2 Binary Super Subexpression Elimination (BSSE)

If two or more common BCSs occur among different coefficients and if these BCSs are having identical shifts between them, then they are known as super subexpressions (SSs). BSSE involves the grouping of SSs terms shared among coefficients. Each coefficient is compared with the remaining coefficients for SSs. If more than one common BHCSs occur between a coefficient pair, the SSs can be grouped together to eliminate redundant computations. The steps for BSSE are as follows:

Step 1: Set $i=0$, where $h(i)$ represents the i -th coefficient.

Step 2: Identify different SSs in $h(i)$.

Step 3: Set $j=i+1$. Check whether $j \leq L$, where $L = \left\lceil \frac{N}{2} \right\rceil$ and N is the number of filter

taps. Since FIR filters are symmetric, only $\left\lceil \frac{N}{2} \right\rceil$ coefficients need to be considered. The filter implementation for the symmetric second-half can be done using structural (inter-tap) adders and without any multiplier block adders. If $j \leq L$, go to Step 4, else go to Step 6.

Step 4: Eliminate the SSs in $h(i)$ from $h(j)$.

Step 5: Increment j and check whether $j \leq L$. If $j \leq L$, go to Step 4, else go to Step 6.

Step 6: Increment i , check whether $i \leq L$. If $i \leq L$, go to Step 2, else END.

4.1.3 Binary Vertical Subexpression Elimination (BVSE)

BVSE deals with the elimination of redundant binary vertical common subexpressions (BVCSs) among the coefficients after the elimination of BHCSs. The coefficient set is scanned for multiple occurrences of the BVCS [1 1], given by $x_4 = x_1 + x_1[-1]$. The problem of extra adders for forming the symmetric part due to delay difference as explained in Chapter 3 with respect to VCSE in CSE techniques of filter coefficients is taken into account while considering BVSE.

4.2 Illustrative Example

In this section, an example to illustrate the proposed BSE method is presented. Consider the symmetrical filter coefficients of a 7-tap FIR filter expressed in binary form as shown in Table 4.1. The BHCSs [1 1], [1 0 1], [1 1 1] and [1 0 0 1] are indicated inside circles in Table 4.1. Table 4.2 is obtained from Table 4.1 by employing the BHSE on the BHCSs in Table 4.1. In Table 4.2, BHCSs $3 = x_7 = [1\ 0\ 1]$, $4 = x_9 = [1\ 1\ 1]$ and $5 = x_{10} = [1\ 0\ 0\ 1]$. The BCSs of 3 and 1 are repeated for h_0 , h_1 and h_2 in Table 4.2 and [3 0 0 0 0 1] forms a SS.

Table 4.1
Binary representation of the filter coefficients

	1	2	3	4	5	6	7	8	9	10	11	12
h_0	0	0	0	1	0	1	0	0	0	1	0	0
h_1	0	0	0	0	1	0	1	0	0	0	1	0
h_2	1	1	1	0	1	0	1	0	0	0	1	0
h_3	1	0	0	1	0	0	0	1	0	0	0	0

Table 4.2
Representation of the filter coefficients after BHSE

	1	2	3	4	5	6	7	8	9	10	11	12
h_0	0	0	0	3	0	0	0	0	0	1	0	0
h_1	0	0	0	0	3	0	0	0	0	0	1	0
h_2	4	0	0	0	3	0	0	0	0	0	1	0
h_3	5	0	0	0	0	0	0	1	0	0	0	0

Using the SS $[3\ 0\ 0\ 0\ 0\ 0\ 1] = x_{11} = 6$ of h_0 , h_1 and h_2 , Table 4.2 is simplified to Table 4.3. From Table 4.3, the expression for output for the symmetric portion:

$$y_k = 2^{-4} x_{11} + 2^{-5} x_{11}[-1] + 2^{-1} x_9[-2] + 2^{-5} x_{11}[-2] + 2^{-1} x_{10}[-3] + 2^{-8} x_1[-3] \tag{4.11}$$

Table 4.3
Final Representation of the filter coefficients

	1	2	3	4	5	6	7	8	9	10	11	12
h_0	0	0	0	6	0	0	0	0	0	0	0	0
h_1	0	0	0	0	6	0	0	0	0	0	0	0
h_2	4	0	0	0	6	0	0	0	0	0	0	0
h_3	5	0	0	0	0	0	0	1	0	0	0	0

The BSE realization of (4.11) and the 7-tap FIR filter is shown in Fig. 4.7. The numerals adjacent to the datapath represent the right shift values. As shown in Fig. 4.7, the LD is 3 adder steps and a total of 7 LOs are required for implementing the MB (LOs needed are only 6 as x_8 is not used). (Note that the MB is shown as dotted box in the Fig. 4.7). For the direct implementation, i. e., implementation without employing any CSE algorithm, considering the representation in Table 4.1, 11 LOs are required and the LD is 3 adder steps. Thus BSE offers 36% reduction of LOs compared to the direct implementation without any increase in LD.

The BSE procedure is as follows:

1. Obtain the filter coefficients in binary-coded format.
2. Eliminate the repetition of BCSs $[1\ 1]$, $[1\ 0\ 1]$, $[1\ 1\ 1]$, and $[1\ 0\ 0\ 1]$ with importance given to minimize LD. Thus $[1\ 1\ 1]$ whose, LD=2, will be given least priority.
3. Eliminate the SSs.
4. Eliminate vertical BCS $[1\ 1]$ while considering the extra adders needed for symmetric part.

Figure 4.7 BSE Realization of the filter with coefficients in Table 4.1.

4.3 Extension of BSE to High-Level Synthesis

In high-level synthesis, the primary goal of transformations has been to optimize the dedicated architectures to obtain a good trade-off between area, power and speed of operation of the circuit [61]. CSE techniques have been extensively used in literature for obtaining these high-level synthesis goals [33, 34, 48, 62, 63]. In [34], the filter coefficient set was considered as a matrix and CSE technique was applied iteratively to obtain a good trade-off between area, power and speed. In [62], CSE technique was employed as an algebraic procedure for improving throughput of the circuit. This was achieved by reducing the LD. In [33], the concept of MCM was employed and the complexity was reduced by iterative pairwise matching. The super subexpression elimination method proposed in [48] combined CSs and formed new CSs and thus reduced additional requirement of adders. But the method was based on CSD

representation of filter coefficients and had some constraints. This can be illustrated by using Fig. 4.8 and employing the same approach in [48].

Figure 4.8 Subexpression sharing as a high-level synthesis transformation.
(a) Conventional 2-bit CSE (b) 3-bit/4-bit Super Subexpression Elimination

The operands a , b , c , and d in Fig. 4.8 represent the input signal of the filter and its shifted versions. The sums e and f are the CSs that are shared for minimizing adders and s_1 , s_2 , s_3 and s_4 represent shifts. Note that 4 adders are required to obtain the final expressions, h and i . Fig. 4.8(b) represents the super subexpression method of [48]. The subexpression g is shared for further reduction of adders to obtain h and i using appropriate shifts, s_2 and s_4 respectively. Thus the number of adders required reduces to 3. But it must be remembered that, CSD consists of negative bits, and hence the combination of A_3 and A_4 to form A_{34} is not always possible due to the change of signs. As a result of this, there are constraints for the technique in [48] for reducing the number of adders. But this signed digit issue will not occur in binary representation and hence the BSSE method can take full advantage of SSs as it has no constraints of negative numbers. In CMOS technology, there are three sources of power dissipation arising from switching (dynamic) currents, short circuit currents, and leakage currents. Among these parameters, the switching component, which is a function of the effective capacitance, plays the most significant role [63]. It is possible to reduce the power by employing transformations such as reductions in LD, number of operations and average transition activity. In [63], it was shown that a binary tree

structured adder always ensure lowest LD and least number of transitions. Hence the binary tree structured approach is employed in this chapter. It must be noted that none of the GD algorithms employ a binary tree structured addition and hence will always result in an increased LD. The number of operations can be reduced through the reduction of adders by employing the BSE. Thus the proposed BSE method improves the efficiency of CSE in high-level synthesis and offers a power efficient solution by reducing the number of operations (additions).

4.4 Design Examples

In this section, results of several design examples of FIR filters are presented.

Example 1: In this example, the number of LOs and LD generated by the proposed algorithm are compared with other algorithms for five benchmark filters FIR1 to FIR5. FIR1 and FIR2 are the example filters presented in [30]. FIR1 has a passband frequency of 0.15π and stopband frequency of 0.25π . For FIR2, the passband and stopband frequencies are 0.021π and 0.07π respectively. FIR3 is the high-pass filter L1 from [64]. FIR3 has a stopband frequency of 0.37π and passband frequency of 0.5π . FIR4 is an FIR filter employed in the filter bank channelizer of digital advanced mobile phone systems (D-AMPS) with passband and stopband frequencies of 0.6173π and 0.6276π respectively. FIR5 is the filter employed in the receivers for the personal digital cellular (PDC) standard. The passband and stopband frequencies of FIR5 are 0.6836π and 0.6973π respectively. The LOs and LDs obtained using these specifications for the proposed BSE method is compared with Pasko [37], NR-SCSE [39], Chia-Yao [40], RAG-n [44], CRA [46] and H-CUB [47]. Table 4.4 shows the comparison of the number of LOs and LDs. In Table 4.4, T represents the filter length and n represents the coefficient wordlength.

Table 4.4
Simulation Results on benchmark filters

			Pasko [37]		NR-SCSE [39]		RAG-n [44]		Chia [40]		CRA [46]		HCUB [47]		Proposed BSE	
			LO	LD	LO	LD	LO	LD	LO	LD	LO	LD	LO	LD	LO	LD
FIR1	25	9	6	2	6	2	6	2	6	2	6	2	5	2	6	2
FIR2	59	14	32	2	30	2	26	2	26	2	30	2	24	6	29	3
FIR3	120	17	58	3	58	3	51	4	51	3	55	3	48	8	50	4
FIR4	200	13	68	4	68	3	64	5	70	3	70	3	63	6	60	3
FIR5	230	16	139	4	137	3	127	5	127	4	138	3	NA	NA	120	4

Note that HCUB algorithm [47] cannot be applied beyond 200 taps and those non-available results are mentioned as *NA*. From Table 4.4, the GD algorithm H-CUB [47], results in minimum LOs compared to other algorithms for shorter filter lengths. But the LOs for H-CUB [47] and all the compared algorithms are more than the proposed BSE method for higher-order filters of 200 and 230 taps. It can be seen from Table 4.4 that, the LD for HCUB is on the higher side. The other GD algorithm RAG-n [44] also provides good LO reduction for lower-order filters. But for higher-order filters the performance of RAG-n [44] is not good for both LOs and LD. Also RAG-n [44] has upper limit for coefficient wordlength due to the use of LUTs. Note that the proposed BSE method offers the best LO reduction compared to the CSE and GD algorithms with minimal increase in the LD for filter lengths greater than 50. For shorter filter lengths ($T < 120$), the factor of N_{up} of (4.2) is almost the same for CSD and binary representations and hence CSD-CSE methods will result in comparable or better adder reduction. Also for higher-order filters, the VCSE and super subexpression elimination become significant and hence the proposed BSE algorithm is advantageous in those cases. As a result of fewer VCSs for lower-order filters, the LD for the proposed method is slightly larger than other methods in literature.

In the following examples, the proposed BSE method is compared in a more comprehensive manner with methods such as NR-SCSE [39] and CRA [46] as these algorithms offered good trade-off between LOs and LDs and HCUB [47] as it offered good reduction of LOs.

Example 2: In this example, the FIR filter with the passband and stopband frequencies of 0.2π and 0.22π respectively are considered. The comparisons have been done for different filter lengths of 20, 50, 80, 120, 200, 400 and 800 and for different wordlengths of 12, 16, 20 and 24 bits. Table 4.5 and Table 4.6 show the comparison of the LOs and LDs needed to implement the filter using the proposed BSE method and the methods in [39], [46] and [47]. Table 4.5 shows that HCUB and CSD-based CSE methods offer better reduction of LOs compared to the proposed BSE, for filters with fewer number of taps. But the increase in LD for HCUB can be clearly seen from Table 4.6. Also HCUB is

applicable only up to 200 taps.

Table 4.5
No. of LOs for Example 2

Filter Length (N)	NR-SCSE [39]				CRA [46]				HCUB [47]				Proposed BSE			
	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit
20	18	24	37	49	19	23	34	41	13	17	24	29	17	24	31	40
50	25	48	67	87	27	45	66	86	20	37	45	65	26	48	58	72
80	50	72	91	126	46	67	90	121	33	59	80	119	34	61	78	112
120	50	79	134	179	46	78	123	165	34	72	110	159	31	69	107	157
200	50	126	179	250	46	121	176	241	38	100	164	225	34	91	159	213
400	55	186	305	450	51	174	294	421	NA	NA	NA	NA	40	126	226	348
800	59	240	470	683	50	228	445	656	NA	NA	NA	NA	38	144	289	473

Table 4.6
LD for Example 2

Filter Length (N)	NR-SCSE [39]				CRA [46]				HCUB [47]				Proposed BSE			
	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit
20	2	3	4	4	2	3	4	4	6	6	7	8	3	3	4	4
50	2	3	4	4	2	3	4	4	6	6	7	8	3	3	4	4
80	2	3	4	4	2	3	4	4	6	6	7	8	3	3	4	4
120	2	3	4	4	2	3	4	4	7	7	8	9	3	3	4	4
200	2	3	3	4	2	3	3	4	8	8	8	9	3	3	4	4
400	2	3	4	4	2	3	4	4	NA	NA	NA	NA	3	3	4	4
800	2	3	4	4	2	3	4	4	NA	NA	NA	NA	3	3	3	4

The proposed BSE method fails to produce the best reduction of LOs for lower-order filters because of three reasons. First, the number of SSs will be less for lower-order filters (In general, the number of SSs is proportional to filter order). Second, fewer number of unpaired bits occur for lower-order filters compared to higher-order filters and third, VCSE becomes significant when the number of filter taps is more. As a result, the advantages of the proposed BSE method cannot be completely exploited to realize lower-order filters. Consequently, the fewer number of non-zero bits in the CSD representation gives the CSE methods slight advantage over the proposed BSE method. However for higher-order filters (50 taps and more), the proposed BSE method offers considerable reduction in LOs compared to methods in [39], [46] and [47]. From Tables 4.5 and 4.6, it can be noted that the proposed BSE method offers the LO reduction without any increase in LD for higher-order filters. Thus the proposed BSE method offers the best trade-off between the LOs and LD especially for higher-order filters. Fig. 4.9 shows the reduction

of LOs achieved using the proposed BSE method and the CSE methods [39] and [46] and HCUB, over the direct method when the filter tap is 200 for wordlengths of 12, 16, 20 and 24 bits.

Figure 4.9 Reduction of LOs in designing the filter with 200 taps using NR-SCSE [39], CRA [46], HCUB [47] and the proposed BSE method over direct method.

The proposed BSE method offers an average LO reduction of 11% over the NR-SCSE, 9% over CRA and 4% over HCUB. But considering LD, HCUB algorithm is inferior to the proposed BSE method. For a 24-bit coefficient, the proposed BSE method offers up to 55.6% improvement in speed over HCUB. Fig. 4.10 shows the LO reductions achieved when the wordlength is 16-bit, for filter lengths 20, 50, 80, 120, and 200. The filter taps of 400 and 800 have not been considered in this example, as HCUB is not applicable in these cases. Note that the proposed BSE method offers better reduction of LOs for filters with more than 50 taps. This is because for higher-order filters, the BVSE and BSSE becomes significant and the proposed method works leaving behind very few unpaired bits. Hence the proposed BSE method becomes more effective for such higher-order filters. For example 2, the average reduction of LOs achieved using the proposed BSE method over the NR-SCSE [39] is 18%, over the CRA [46] is 14%, and over the HCUB [47] is 7%.

Figure 4.10 Reduction of LOs in designing the filter with 16-bit coefficient word length using NR-SCSE [39], CRA [46], HCUB [47] and the proposed BSE method over direct method.

Example 3: In this example, the FIR filters employed in the filter bank channelizer of D-AMPS in [22] are examined. These filters must have a large number of taps due to the stringent adjacent channel attenuation specifications of wireless communications standards. The sampling rate chosen is 34.02 MHz as in [22]. The channel filters extract 30 kHz D-AMPS channels from the input signal after down sampling by a factor of 350. The passband and stopband edges are 30 kHz and 30.5 kHz respectively. The peak passband ripple is chosen as 0.1 dB. The filter stop-band specifications are chosen as in the D-AMPS standard [65]. The length of the FIR filter N is determined using (4.12) [25].

$$N = \frac{-10 \log_{10} \frac{\partial_1 \partial_2 - 13}{14.6 \Delta f}}{1} + 1 \quad (4.12)$$

where ∂_1 and ∂_2 are the peak passband and stopband ripples respectively, and Δf is the normalized width of the transition-band. The comparison of LOs and LDs needed to implement the filter using the proposed BSE method with the methods in [39], [46] and super subexpression technique in [48] are shown in Tables 4.7 and 4.8 respectively. As the number of taps is well above 200, HCUB is not applicable in this example. Filters of lengths 200, 460, 610, 940 and 1180 are chosen corresponding to peak stopband ripple (PSR) specifications of -24 dB, -48 dB, -65 dB, -85 dB and -96 dB respectively.

Table 4.7
No. of LOs for D-AMPS Filter example

PSR (dB)	Filter Length (N)	NR-SCSE [39]				CRA [46]				SS [48]				Proposed BSE			
		12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit
-24	200	55	130	200	267	54	125	193	262	55	121	190	260	41	108	146	233
-48	460	55	250	370	500	55	246	361	497	52	203	350	500	41	149	274	408
-65	610	59	287	480	620	53	268	457	613	50	253	467	620	42	164	316	473
-85	940	55	325	640	870	55	322	623	859	50	300	635	855	42	181	384	636
-96	1180	67	374	723	1040	65	358	700	1032	55	373	706	987	46	203	425	707

Table 4.8
LD for D-AMPS Filter example

PSR (dB)	Filter Length (N)	NR-SCSE [39]				CRA [46]				SS [48]				Proposed BSE			
		12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 Bit	12 bit	16 bit	20 bit	24 bit	12 bit	16 bit	20 bit	24 bit
-24	200	2	3	4	4	2	3	4	4	3	3	4	5	3	3	4	4
-48	460	2	3	4	4	2	3	4	4	3	3	4	5	3	3	3	4
-65	610	2	3	4	4	2	3	4	4	3	3	4	5	3	3	4	4
-85	940	2	3	4	4	2	3	4	4	3	3	4	5	3	3	4	4
-96	1180	2	3	4	4	2	3	4	4	3	4	4	5	3	4	4	4

From Table 4.7, it can be seen that the number of LOs required for the proposed BSE method is considerably less compared to the methods in [39], [46], and [48]. Table 4.8 shows that the LDs of filters realized using the proposed BSE method are almost same as that obtained using NR-SCSE and shorter than CRA and SS [48]. The percentage reductions of LOs achieved using the proposed BSE method and NR-SCSE and CRA over direct implementation for the 610-tap filter in Table 4.7 for different wordlengths are shown in Fig. 4.11. The proposed BSE method offers an average LO reduction of 17% over the NR-SCSE, 12% over CRA and 11% over SS [48].

Fig. 4.12 shows the percentage reductions of LOs for different filters of lengths 200, 460, 610, 940 and 1180 and 16-bit coefficient wordlength. The average reductions of LOs in Fig. 4.12 using the proposed BSE method over the NR-SCSE, CRA and SS [48] are 23%, 18% and 15% respectively. Overall, for the FIR filters in example 3, the proposed BSE method offers an average LO reduction of 20% over the NR-SCSE 16% over CRA, and 13% over SS [48] with negligible increase in the LD.

Figure 4.11 Reduction of LOs in designing the D-AMPS channel filter with 610 taps using NR-SCSE [39], CRA [46], SS [48] and the proposed BSE method over direct method.

Figure 4.12 Reduction of adders in designing the D-AMPS channel filter with 16-bit coefficient word length using NR-SCSE [39], CRA [46], SS [48] and the proposed BSE method over direct method.

For all the example filters considered (shown in Tables 4.4, 4.5 and 4.7), the proposed BSE method offers an overall average LO reduction of 24% over the NR-SCSE method [39] and 18% over CRA method [46] without any increase in the LD.

4.5 Summary

In this chapter, a new common subexpression elimination (CSE) method using binary representation of coefficients has been proposed to implement low-complexity finite

impulse response (FIR) filters. It is shown that the CSE technique can be efficiently employed for reducing the number of adders using binary representation of filter coefficients compared to the canonical signed digit (CSD) representation. The reduction of adders achieved using the proposed binary subexpression elimination (BSE) method is slightly inferior for short filters. However, the BSE method offers better reduction of adders without any increase in the critical path length for higher-order filters. Therefore, the proposed BSE method is best suited for implementing higher-order FIR filters in software defined radio (SDR) channelizers. Design examples showed that the proposed BSE method offers an average LO reduction of 24% over the NR-SCSE method [39] and 18% over CRA method [46] without any increase in the logic depth. However the proposed BSE method and all the CSE techniques [34]-[42] in literature have been applied only to fixed-coefficient FIR filters. Hence they cannot be employed for reconfigurable filters in SDR receivers. In the next chapter, the proposed BSE technique has been extended to reconfigurable FIR filters and proposed two new low complexity and reconfigurable FIR filter architectures.

Chapter 5

Reconfigurable Low Complexity Channel Filters

Integration of reconfigurability and low complexity in a single finite impulse response (FIR) filter architecture is an important requirement in software defined radio receivers, which is the focus of the work presented in this chapter. Two new FIR filter architectures called constant shifts method (CSM) and programmable shifts method (PSM) are proposed in this chapter. The CSM and PSM architectures consider coefficients, which are stored in look up tables (LUTs), as constants and input signal as a variable. The coefficient multiplication in such a case becomes a multiple constant multiplication (MCM) task. The MCM is then optimized to eliminate redundant multiplications using the binary subexpression elimination (BSE) algorithm proposed in Chapter 4 for reducing the filter complexity. The proposed CSM focuses on the implementation of FIR filters employing partitioning of filter coefficients into fixed groups. The proposed PSM has a pre-analysis part which eliminates the redundancy in coefficient multiplications using the BSE algorithm. The advantage of CSM is that it produces high speed filters at the cost of a slight increase in area and power consumption. On the other hand, the PSM produces filters with low area and low power consumption at the cost of a slight increase in delay. Another advantage of PSM is that, the wordlength of the filter coefficients can be changed dynamically without any modification in the hardware. A detailed analysis of the proposed architectures is given the following sections.

5.1 Proposed Reconfigurable FIR Filter Architectures

In this section, the architecture of the proposed reconfigurable FIR filter is presented. The proposed architecture is based on the transposed direct form FIR filter structure as shown in Fig. 5.1. In Fig. 5.1, $PE-i$ represents the processing element corresponding to the i th coefficient. The PE performs the coefficient multiplication operation with the help of a shift and add unit which will be explained in the latter part of this section. The architecture of PE is different for the proposed CSM and PSM. In the CSM, the filter

coefficients are partitioned into fixed groups and hence the PE architecture involves constant shifters. But in the PSM, the PE consists of programmable shifters (PSs). The proposed FIR filter architecture can be realized in a serial way in which the same PE is used for generation of all partial products by convolving the coefficients with the input signal ($h*x[n]$) or in a parallel way, where parallel PE architectures are employed. The first option is used when power consumption and area are of prime concern. The basic architecture of the PE is shown in Fig. 5.2. The functions of different blocks of the PE are explained below:

a) **Shift and Add Unit:** It is well known that, one of the efficient ways to reduce the complexity of multiplication operation is to realize it using shift and add operations. In contrast to shift and add units used in conventional reconfigurable filter architectures, the BCSs-based shift and add unit has been used in the proposed CSM and PSM architectures. The architecture of shift and add unit is shown in Fig. 5.3. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1].

Figure 5.1 Transposed direct form of an FIR filter.

In Fig. 5.3, ' $x \gg k$ ' represents the input x shifted right by k units. All the 3-bit BCSs [0 1 1], [1 0 1], [1 1 0] and [1 1 1] of a 3-bit number are generated using only 3 adders, whereas a conventional shift and add unit would require 5 adders. Since the shifts to obtain the BCSs are known beforehand, PSs are not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit. In both the CSM and PSM architectures, the same shift and add unit has been used. Thus the use of 3-bit BCSs

reduces the number of adders needed to implement the shift and add unit compared to conventional shift and add units.

Figure 5.2 Architecture of the processing element (PE).

Figure 5.3 Architecture of shift and add unit.

b) Multiplexer Unit: The multiplexer units are used to select the appropriate output from the shift and add unit. All the multiplexers will share the outputs of the shift and add unit. The inputs to the multiplexers are the 8/4 inputs from the shift and add unit and hence 8:1/4:1 multiplexer units are employed in the architecture. The select signals of the multiplexers are the filter coefficients which are previously stored in an LUT. The CSM and PSM architectures differ in the way filter coefficients are stored in the LUT. In the CSM, the coefficients are directly stored in LUTs without any modification whereas in

PSM, the coefficients are stored in a coded format. The multiplexer requirement will also be different for PSM and CSM. In CSM, the number of multiplexers depends on the number of bit groups after the partitioning of the filter coefficient into fixed bit groups. The number of multiplexers in the PSM depends on the number of non-zero operands in the coefficient for the worst-case after the application of BSE algorithm which is explained in details in 5.1.2.

c) Final Shifter Unit: The final shifter unit will perform the shifting operation after all the intermediate additions (i.e. intra-coefficient additions) are done. This can be illustrated using the output expression (5.1), where x is the input and the powers-of-two terms represent filter coefficients.

$$y = 2^{-4}x + 2^{-6}x + 2^{-15}x + 2^{-16}x \quad (5.1)$$

By partitioning (5.1) into groups of two bits:

$$y = 2^{-4}(x + 2^{-2}x) + 2^{-15}(x + 2^{-1}x) \quad (5.2)$$

After obtaining the intermediate sums $(x + 2^{-2}x)$ and $(x + 2^{-1}x)$ from the shift and add unit with the help of multiplexer unit, the final shifter unit will perform the shift operations 2^{-4} and 2^{-15} in (5.2). In the CSM, the final shifts are constants and hence no PSs are required whereas the PSM employs PSs.

d) Final Adder Unit: This unit will compute the sum of all the intermediate additions $2^{-4}(x + 2^{-2}x)$ and $2^{-15}(x + 2^{-1}x)$ as shown in (5.2). As the filter specifications of different communication standards are different, the coefficients change with the standards. In conventional reconfigurable filters, the new coefficient set corresponding to the filter specification of the new communication standard is loaded in the LUT. Subsequently the shift and add unit performs a bitwise addition after appropriate shifts. On the contrary, the proposed CSM and PSM architectures perform a BCS-wise addition (instead of bitwise addition). Thus the same hardware architecture can be used for different filter specifications to achieve the necessary reconfigurability. Moreover the proposed BCS-based shift and add unit reduces the number of addition operations, which will in turn reduce the hardware complexity. In the next section, the CSM is explained in a detailed manner.

5.1.1 Architecture of Constant Shifts Method (CSM)

In the CSM architecture, the coefficients are stored directly in the LUT. These coefficients are partitioned into groups of 3-bits and are used as the select signal for the multiplexers. The number of multiplexer units required is $\lceil n/3 \rceil$, where n is the wordlength of the filter coefficients. The CSM can be explained with the help of an 8-bit coefficient $h = '0.11111111'$. This coefficient h is the worst-case 8-bit coefficient since all the bits are nonzero and hence needs maximum number of additions and shifts. The number of multiplexers required is 3 as n is 8. The output $y = h * x$ is expressed as

$$y = 2^{-1}x + 2^{-2}x + 2^{-3}x + 2^{-4}x + 2^{-5}x + 2^{-6}x + 2^{-7}x + 2^{-8}x \quad (5.3)$$

By partitioning (5.3) into groups of 3 bits from most significant bit (MSB),

$$h = 2^{-1}(x + 2^{-1}x + 2^{-2}x + 2^{-3}(x + 2^{-1}x + 2^{-2}x) + 2^{-6}(x + 2^{-1}x)) \quad (5.4)$$

Note that the terms $x + 2^{-1}x + 2^{-2}x$ and $x + 2^{-1}x$ can be obtained from the shift and add unit in Fig. 5.3. Then by using the 3 multiplexers (Mux), i. e., two 8:1 Mux for the first two 3-bit groups and one 4:1 Mux for the last two bits of the filter coefficients, the intermediate sums shown inside the brackets of (5.4) can be obtained. The final shifter unit will perform the shift operations 2^{-1} , 2^{-3} and 2^{-6} . Since these shifts are always constants irrespective of the coefficients, they can be hardwired and PSs are not required. The final adder unit will compute the sum of all the intermediate sums to obtain $h * x[n]$.

The architecture of the PE for CSM is shown in Fig. 5.4. The coefficient wordlength considered is 16 bits. The filter coefficients are stored in the LUT in sign-magnitude form with the MSB reserved for the sign bit. The first bit after the sign bit is reserved for the integer bit and the rest 16 bits will form the magnitude of the coefficients. Thus each 16-bit coefficient is stored as an 18-bit value in LUTs. Each row in LUT corresponds to one coefficient. Note that only half the number of coefficients need to be stored as FIR filter coefficients are symmetric. The coefficient values corresponding to 2^0 to 2^{-14} are partitioned into groups of 3 bits and are used as select signals to multiplexers Mux1 to Mux5. i. e., the set $(2^0, 2^{-1}, 2^{-2})$ forms the select signal to Mux1 and so on. Since there are 3-bits, 8 combinations are possible and hence Mux1 to Mux5 are 8:1 multiplexers. The value corresponding to 2^{-15} forms the select to a 2:1 multiplexer, Mux6.

Figure 5.4 Architecture of the PE for CSM.

The output from the i th multiplexer is denoted as r_i . Note that even though coefficient with values up to a wordlength of 16 bits are taken, the shifting of 2^{-1} is done finally as shown in (5.4) and hence the maximum shift will be 2^{-15} . Mux7 determines whether the output needs to be complemented based on the sign bit of the filter coefficient and hence it is a 2:1 multiplexer. In FIR filters, coefficient values are always less than one (In the all design examples, Parks-McClellan algorithm has been used to design filters using “firpm” command in Matlab). Hence the integer bit has not been used. However if coefficients have values greater than one, appropriate scaling can be applied to obtain fractional values. In Fig. 5.4, the shifts are obtained as follows: Let r_1 to r_6 denotes the outputs of Mux1 to Mux6 respectively. Then

$$y = 2^{-1}r_1 + 2^{-4}r_2 + 2^{-7}r_3 + 2^{-10}r_4 + 2^{-13}r_5 + 2^{-16}r_6 \quad (5.5)$$

The shifts are obtained by partitioning the 16-bit coefficient into groups of 3-bits. By partitioning (5.5),

$$y = 2^{-1}[(r_1 + 2^{-3}r_2) + 2^{-6}[(r_3 + 2^{-3}r_4) + 2^{-6}(r_5 + 2^{-3}r_6)]] \quad (5.6)$$

Substituting $(r_1 + 2^{-3}r_2)$, $(r_3 + 2^{-3}r_4)$ and $(r_5 + 2^{-3}r_6)$ respectively by r_7 , r_8 and r_9 respectively,

$$y = 2^{-1}[r_7 + 2^{-6}((r_8 + 2^{-6}r_9))] \quad (5.7)$$

By substituting $(r_8 + 2^{-6}r_9)$ by r_{10} ,

$$y = 2^{-1}(r_7 + 2^{-6}r_{10}) \quad (5.8)$$

By substituting $(r_7 + 2^{-6}r_{10})$ by r_{11} ,

$$y = 2^{-1}(r_{11}) \quad (5.9)$$

The expressions from (5.6)-(5.9) are represented in the Fig. 5.4. The main advantage of the CSM architecture is that all the shifts are constants irrespective of the coefficients. The coefficients are always partitioned into fixed groups of three bits and the shifts are constants as shown in Fig. 5.4. Hence the shifts in the CSM architecture can be hardwired resulting in high speed operation of the filter.

The shift and add unit employed in the proposed approach can generate all the 3-bit BCSs using only 3 adders. The impact of using higher-order BCSs (4-bit, 5-bit BCSs etc.) has also been investigated. The choice of the best shift and add unit will depend on the complexities of (1) shift and add unit, (2) multiplexer unit and (3) final adder unit. The number of adders needed to implement n -bit CSs is $2^{n-1} - 1$ as explained in Chapter 4. Thus shift and add units capable of generating 4-bit, 5-bit and 6-bit BCSs would require 7, 15 and 31 adders respectively. The LD is 2 adder-steps for both the 3-bit and the 4-bit BCSs-based shift and add units, and hence they have the same speed. The LD of 5-bit and 6-bit BCSs-based shift and add units are same, i.e., 3 adder-steps, which is 1 adder-step more than that of the 3-bit and 4-bit BCSs. Thus the 3-bit BCSs-based shift and add unit results in fewer number of adders than the 4-bit BCSs-based shift and adder unit (reduction of 4 adders) with the same LD. The requirement of additional 4 adders would

increase the complexity of the 4-bit BCSs-based shift and add unit. Note that the cost of shift and add unit is independent of the number of coefficients (filter length) as the same shift and add unit is shared by all the coefficients. In the proposed CSM architecture, $\lceil W/3 \rceil$ number of 8:1 multiplexers ($\lfloor W/3 \rfloor$ 8:1 multiplexers and remaining 2:1 or 4:1 multiplexers in some cases) of bit-width $(l+2)$ are required, where W is the coefficient wordlength and l is the input data wordlength. For example, if $W = 16$, the proposed 3-bit BCSs-based approach requires five 8:1 multiplexers and one 2:1 multiplexer. On the other hand, if 4-bit BCSs were used instead of 3-bit BCSs, four 16:1 multiplexers are required. Assuming that the complexity of one 8:1 multiplexer is equivalent to that of four 2:1 multiplexers, and that of a 16:1 multiplexer is equivalent to that of eight 2:1 multiplexers, the 3-bit BCSs-based PE requires twenty one 2:1 multiplexers and 4-bit BCSs-based PE requires thirty two 2:1 multiplexers respectively. Thus the multiplexer complexity would increase when 4-bit BCSs are used. To be more precise, for each PE with 16-bit filter coefficients, the multiplexer complexity of 4-bit BCSs-based PE is increased by eleven 2:1 multiplexers when compared to 3-bit BCSs based shift and add unit. But it can be noted that, the total number of adders required for 3-bit BCS based filter with n coefficients is $3+5n$ (3 adders for shift and add unit and 5 adders for each PE) and that for 4-bit BCS based PE is $7+3n$ (7 adders for shift and unit and 3 adders for each PE). Hence $2n$ adders ($5n-3n$) are saved for 4-bit BCSs-based filter for n PEs. It must be noted that while considering the complexity of PE, the complexity of the shift and add unit is not considered. From the above discussion, it can be concluded that, if 4-bit BCSs were used instead of 3-bit BCSs, the complexity of the shift and add unit and multiplexer unit of PE would have increased, whereas the complexity of the final adder unit would decrease.

To provide a quantitative comparison, consider a 16-bit (i. e., $W = 16$) coefficient with an 8-bit quantized (i. e., $l = 8$) input signal. The number of filter taps is fixed as 100. The proposed 3-bit BCSs-based and 4-bit BCSs based CSM architectures have been implemented on Virtex 4XC4VSX35-10ff668 FPGA and the complexities of the shift and add unit, multiplexer unit, final adder unit and over-all complexities have been compared. The 5-bit and higher-bit BCSs based CSM architecture have higher LD as discussed in the

previous paragraph and hence have not been considered. The implementation results are presented in Table 5.1.

Table 5.1
Synthesis results for 3-bit BCSs and 4-bit BCSs based CSM architecture

	3-bit BCS				4-bit BCS			
	Shift and Add Unit	Multiplexer Unit	Final Adder Unit	Over-all	Shift and Add Unit	Multiplexer Unit	Final Adder Unit	Over-all
LUTs	2058	3026	2976	8976	2987	5434	2342	10875
Slices	1420	1768	1678	5023	1976	2345	1245	5800
Flip-flops	1125	1450	1345	4198	1597	2467	1056	5170

From Table 5.1, it is clear that, the complexity of shift and add unit and the multiplexer unit is more for 4-bit BCSs based approach and the complexity of final adder unit is more for 3-bit BCSs based approach. Over-all, the 3-bit BCSs-based shift and add unit results in low complexity implementation when compared to the 4-bit BCSs-based implementation. Nevertheless, it must be noted that the CSM architecture can be easily modified to incorporate 4-bit or 5-bit shift and add unit based CSM architectures, if an application demands such a requirement.

In the CSM approach, as the coefficients are directly stored in the LUT, it is not possible to eliminate entire redundancy in coefficient multiplication. Also in case of the outputs of any of the multiplexers becoming zero, the adder corresponding to that Mux will be used, which is not required if the output is zero. But it can be seen that, the adders at the output of the multiplexers can be combined in many ways and hence the best power saving solution can be utilized. Also carry save adders can be employed if much faster operation is required. The above drawbacks in CSM are resolved by employing the BSE algorithm proposed in Chapter 4. This forms the PSM architecture which is explained in the next section.

5.1.2 Architecture of Programmable Shifts Method (PSM)

In Chapter 4, it was shown that the BSE method results in minimum number of adders for fixed coefficient filters compared to previously proposed CSE methods [34-40]. However the BSE architecture presented in Chapter 4 is not reconfigurable. The PSM architecture

presented in this section incorporates reconfigurability into the BSE. The PSM has a pre-analysis part in which, the filter coefficients are analyzed using the BSE algorithm. Thus the redundant computations (partial product additions) are eliminated using the BCSs and the resulting coefficients are stored in the LUT in a coded format. The coding format is explained in the latter part of this section. The shift and add unit is identical for both PSM and CSM. The number of multiplexer units required can be obtained from the filter coefficients after the application of BSE. The multiplexer requirement is fixed after considering the number of non-zero operands (BCSs and unpaired bits) in each of the coefficient after employing the BSE algorithm. The number of multiplexers will correspond to the number of non-zero operands for the worst-case coefficient (worst-case coefficient being defined as coefficient that has the maximum number of non-zero operands).

The architecture of the PE for PSM is shown in Fig. 5.5. The coefficient wordlength is fixed as 16 bits. A statistical analysis for various filters with coefficient wordlength of 16 bits and different filter lengths (20, 50, 80, 120, 200, 400 and 800 taps) has been done and it was found that the maximum number of non-zero operands (CSs + unpaired bits) is 5 for any coefficient. The analysis was done for filters with different passband (ω_p) and stopband (ω_s) frequency specifications given by (a) $\omega_p = 0.1\pi$, $\omega_s = 0.12\pi$, (b) $\omega_p = 0.15\pi$, $\omega_s = 0.25\pi$, (c) $\omega_p = 0.2\pi$, $\omega_s = 0.22\pi$ and (d) $\omega_p = 0.2\pi$, $\omega_s = 0.3\pi$ respectively. Based on the statistical analysis, the number of multiplexers has been fixed as 5 (same as the number of non-zero operands). The LUT consists of two rows of 18 bits for each coefficient of the form SDDDDXXDDDDXXMMMML and DDDDXDDDDXXDDDDXX, where 'S' represents the sign bit, 'DDDD' represents the shift values from 2^0 to 2^{-15} and 'XX' represents the input 'x' or the BCSs obtained from the shift and add unit. In the coded format, XX='01' represents 'x', '10' represents $x+2^{-1}x$, '11' represents $x+2^{-2}x$, '00' represents $x+2^{-1}x+2^{-2}x$ respectively. Thus the two rows can store up to 5 operands which is the worst-case number of operands for a 16-bit coefficient. In most of the practical coefficients, the number of operands is found to be less than the worst-case number of operands, 5. In such cases, the five MSB bits of the 18-bit format 'MMMML' can be used to avoid unnecessary additions. However if a situation

arises where the number of operands is more than five, the PSM architecture needs to be expanded with more multiplexers and adders.

Figure 5.5 Architecture of the PE for PSM.

The values 'MMMM' will be given as select signal to the Mux6 and 'L' to Mux8. 'MMMML' indicates the presence of five operands. A '1' in each position indicates the presence of each operand. Thus if all operands are present, 'MMMML' = '11111'. This means the Mux6 will select the output from the output of adder, A₄ and Mux8 will select the output of adder, A₂. If only first operand is present, 'MMMML' = '10000'. This means the Mux8 will select the output of PS, shr₄ and Mux6 will select the output of PS, shr₁. As

a result of this, none of the adders A_1 to A_4 will be loaded saving significant amount of dynamic power. The coding can be explained with a coefficient $h = [1010011001010011]$. By using the BSE approach, substituting $2=[1\ 1]$, $3=[1\ 0\ 1]$, $h = [3000020003000020]$. Then the coefficient h will be stored in the LUT as 000001101011011110 and 10011111010000000. It must be noted that as h has only 4 operands, the fifth operand values 'DDDDXX' are substituted as 000000 and 'MMMML' as '11110'. The XX values are given as select signals for Mux1 to Mux5. The values of DDDD are fed to corresponding PS. The multiplexer Mux6 and Mux8 will select the appropriate output in case the number of operands after BSE is less than 5. The use of Mux6 and Mux8 reduces the number of adders utilized by selecting the output from the appropriate adder as all the adders in the PE are not always needed. For example, only four operands occur in the coefficient, h , above and therefore the output can be taken from the output of PS, shr_4 without using adder, A_2 . Mux8 will perform this and hence the adder A_2 is not loaded and consumes zero current and power. The select signals of Mux6 and Mux8 has five bits and hence 2^5 different control signals are possible which adds considerable flexibility to the architecture. Mux7 is used to complement the output in case of a negative coefficient and its select signal is the sign bit 'S' of the coefficient.

The PSM architecture has two advantages. First it guarantees fewer number of additions compared to CSM. Second, it offers the flexibility of changing the wordlength of coefficients. The same PSM architecture designed for 16-bit coefficients is capable of operating for any coefficient wordlength less than 16 bits. This means, if the wordlength is reduced, the format of the LUT can be changed if required. Thus the coefficient wordlength of the proposed PSM architecture can be changed dynamically without any change in hardware. The advantage of reducing the wordlength is that, some of the adders in the PSM architecture will be unloaded resulting in zero dynamic power.

5.1.3 Comparison of CSM and PSM

The idea of CSM is to split the filter coefficients into groups of 3 bits and use these groups as selectors to multiplexer unit and obtain the product $h \cdot x[n]$. This does not guarantee the substantial reduction of additions to be performed. In PSM, since the BSE

algorithm is employed, the number of additions required will always be less compared to CSM. This can be illustrated as follows. Consider the coefficient $h = [010100001010]$. If CSM is employed, always 4 multiplexers are needed and this means the shift and add unit in Fig. 5.3 needs to be used 4 times. Thus 3 additions are required for CSM. But if PSM is used, first BSE is applied to obtain the BSE-coded form, $h_1 = [020000002000]$. The output computation corresponding to h requires only 2 additions, one for h_1 and one for obtaining $2 = [101]$. This reduction is significant for higher-order filters used in SDR channelizers. Therefore the proposed PSM architecture is best suited for the channel filters in SDRs. In the case of PSM, the final shift is done based on the values from LUT using PSs whereas in the case of CSM, the shifts are constants because the filter coefficients are partitioned into fixed groups of 3-bits. Thus the CSM architecture results in faster coefficient multiplication operation at the cost of few extra adders compared to PSM architecture whereas the PSM architecture results in fewer number of additions and thus low area and low power consumption compared to the CSM architecture, at the cost of a slight increase in delay.

Another advantage of PSM is that, it is capable of working for any wordlength of the filter coefficients less than the designed wordlength (i. e., 16 bits in this case). The number of multiplexers in the PSM architecture is fixed based on the number of BCSs present in a given coefficient set (worst-case-coefficient of the set). Thus even if the wordlength changes, it hardly affects the architecture of PSM. In [54], it was pointed out that, for many filters, the highest coefficient wordlength is not required. Valuable hardware resources will be wasted if all taps are implemented with the highest wordlength. The proposed PSM architecture can be implemented for dynamically varying coefficient wordlength. One of the limitations of the PSM architecture is that, it requires preanalysis of filter coefficients, which will impose some constraints on on-the-fly reconfigurability. But this restriction does not impose constraints on popular reconfigurable filter applications like wireless communications. This is because in such applications, a distinct filter is required for each communication standard and the coefficients of the filter are fixed for a specific standard. When the communication system is operating on a particular wireless standard, the filter coefficients do not change, i.e., the

filter is not required to be an adaptive filter. In the case of a multi-standard transceiver, when the system changes its mode of operation to a different wireless communication standard, the coefficient set corresponding to the specification of the new standard is loaded replacing the current filter coefficients. Note that the coefficients of the new standard are known beforehand (pre-stored) and therefore the pre-analysis can be done offline and the necessity of on-the-fly reconfigurability does not really exist in most wireless communication applications.

5.2 Extension of CSM and PSM to High Level Synthesis

In this section, an extension of proposed reconfigurable architectures to high level synthesis is presented. CSE techniques have been used in literature as a powerful transformation for eliminating hardware redundancies to reduce power consumption and area [33, 48]. However there is hardly any work that addressed the problem of designing reconfigurable architectures using CSE techniques. In [57], the concept of reconfigurable multiplier block (ReMB) was introduced, which utilized GD algorithms for eliminating coefficient redundancies and thus reducing the number of additions for the ReMB. The approach in [57] reduces the redundancies in multiplications by pushing multiplexers deep into the ReMB design, thus increasing the number of multiplexers. In other words, the reduction in complexity achieved by the approach in [57] is directly proportional to the number of multiplexers. But it is shown in [69] that the delay imposed by multiplexers in reconfigurable designs can heavily degrade the performance of the system, which will have adverse effects on the architecture in [57]. In this work, the BSE algorithm has been employed to reduce the redundancies in multiplications in the reconfigurable filter architecture. To the best of author's knowledge, this is the first approach that employs CSE technique to achieve high-level synthesis goals for reconfigurable systems. The proposed CSM and PSM methods make use of architectures with fixed number of multiplexers and the reduction in complexity is achieved by applying the BSE algorithm proposed in Chapter 4. Also, the shift and add unit, which significantly reduces the number of adders compared to direct implementation, has no multiplexer in contrary to the approach in [57].

The high level synthesis literature has an extensive coverage of employing partitioning techniques to integrate low power realization within the scheduling process [29-32]. These methods generally use some scheduling techniques or path analysis to identify regions that can be organized into partitions. Each partition will have an activation/deactivation mechanism, which can be controlled. The basic idea is that the partition can be switched off when it is not used and consequently power can be saved. The methods in [69-72] have not exploited hardware level redundancy in operations which can result in better performances of the system. In [73], an algorithm based on graphs was devised which reuses the hardware resulting in less power consumption. But reusing of hardware results in increased number of multiplexer logic being created which degrades the system performance as discussed in [69]. The partitioning of coefficients into 3-bit groups in the proposed CSM is a high level synthesis transformation targeted to reduce power consumption. In the CSM architecture, the partitioned coefficient bit groups are given as select signals to multiplexers. These multiplexers will load and unload different parts of the circuit and thus save significant amount of power as discussed in Section 5.2.2. In CMOS technology, there are three sources of power dissipation arising from switching (dynamic) currents, short circuit currents, and leakage currents. Among these parameters, the switching component, which is a function of the effective capacitance, plays the most significant role [63]. It is possible to reduce the power by employing transformations such as reductions in LD, number of operations and average transition activity. In [63], it was shown that a binary tree-structured adder always ensured lowest LD and consequently the least number of transitions. The proposed CSM and PSM architectures also employ the binary tree-structured approach so as to achieve low LD. It must be noted that as the coefficients are synthesized sequentially in GD algorithms, the resulting filter structures do not have a binary tree structure and hence will always result in an increased LD. Hence the reconfigurable approaches in [57] and [58] which employ GD algorithms will result in increased power consumption when applied to high-level synthesis. In addition to reducing the LD in the CSM and PSM architectures, the number of operations is also reduced by employing the BSE. Furthermore, the proposed PSM architecture can make use of dynamic change of coefficient wordlength which will save significant amount of dynamic power as explained in Section 5.2.2. In conclusion, the

proposed CSM and PSM approaches improve the efficiency of reconfigurable systems in high-level synthesis and offers a power efficient solution by reducing the LD as well as the number of operations (additions).

5.3 Experimental Results

In this Section, the synthesis and design results of the proposed CSM and PSM architectures are presented and compared with recently proposed reconfigurable FIR filter architectures in literature [54, 55, 57, 58].

5.3.1 Synthesis Results

For synthesis purpose, *Xilinx*[®] 8.1i integrated software environment (ISE) was employed. The synthesis has been done on *Xilinx's Virtex-II 2v3000ff1152-4* FPGA. Table 5.2 shows the synthesis results of the proposed CSM and PSM in realizing a 20-tap FIR filter that has a coefficient wordlength of 16 bits. The implementation of filters have been done with different passband edge (ω_p) and stopband edge (ω_s) specifications given by (a) $\omega_p = 0.1\pi$, $\omega_s = 0.12\pi$, (b) $\omega_p = 0.15\pi$, $\omega_s = 0.2\pi$, (c) $\omega_p = 0.2\pi$, $\omega_s = 0.22\pi$ and (d) $\omega_p = 0.2\pi$, $\omega_s = 0.3\pi$ respectively.

Table 5.2
Synthesis results for an FIR Filter with 20 taps and coefficient wordlength of 16 bits

	Proposed PSM	Proposed CSM
LUTs	1527	1693
Slices	795	888
Flip-flops	665	627
Data Arrival Time (ns)	33.64	26.824

Even though the proposed architectures are reconfigurable, the usage of adders and shifters are dependent on the filter coefficient values. Some of the adders may not be used by the multiplexers. As a result of this, they are unloaded and do not consume any dynamic power. Hence the power and speed values of the synthesis results are dependent on the filter coefficients. Therefore the average of the synthesis results obtained for all the filters whose specifications are given earlier, are presented in Tables 5.2, 5.3 and 5.4. From Table 5.2, it can be noted that, the CSM requires 93 slices more than that of PSM, whereas PSM requires 6.82ns more for the data to arrive at the output compared to CSM.

Thus the CSM results in higher speed whereas the PSM results in lower area. The reason for lower speed of PSM is due to the presence of PSs and that of less area is due to elimination of redundant additions by using BSE algorithm. The synthesis values for different filter coefficient wordlengths of 8, 12 and 16 bits for the PSM architecture are shown in Table 5.3. It must be noted that, the PSM architecture has the capability to adapt to dynamically changing coefficient wordlengths. Thus by choosing the appropriate filter coefficient wordlength, it is possible to obtain reduced area and power as well as increased speed for the PSM architecture.

Table 5.3
Synthesis results of PSM architecture for different coefficient wordlengths

Wordlength	8-bit	12-bit	16-bit
LUTs	803	1340	1527
Slices	400	600	795
Flip-flops	356	495	665
Data Arrival Time (ns)	19.96	29.76	33.64

5.3.2 CSD Based Reconfigurable FIR Filter Architecture

The CSD-based CSE algorithms are considered to be one of the best algorithms that can result in low complexity fixed-coefficient FIR filter implementations. However the implementation of CSD-CSE based reconfigurable filter architectures have been hardly addressed in literature. In this work, two FIR filter architectures have been implemented: a CSD-based FIR filter using the CSM architecture, which is referred to as CSD-CSM in the sequel, and a CSD-CSE based FIR filter employing the CSD-based CSE similar to the proposed PSM architecture. The latter architecture is referred to as CSD-PSM. For low complexity, the CSE algorithm in [35] has been employed on the coefficients before they are stored in LUT. A CSD-based shift and add unit to generate CSs such as $[1\ 0\ 1]$, $[1\ 0\ -1]$, $[1\ 0\ 0\ 1]$ and $[1\ 0\ 0\ -1]$ and their negated versions have also been implemented. In the previous works based on CSE algorithm [34-42], it was assumed that CSs such as $[-1\ 0\ -1]$ and $[-1\ 0\ 1]$ can be generated from their respective negated versions $[1\ 0\ 1]$ and $[1\ 0\ -1]$ by configuring the existing adder as a subtractor without using any extra adder. But this is applicable only for fixed coefficient filters. An n -bit adder circuit would require n additional XOR gates to reconfigure the adder to subtractor mode. These additional XOR gates would increase the critical path of the adder circuit (equivalent to the delay of

n half-adders) and impose overheads for CSD implementation of the FIR filter. Another drawback of CSD implementation is the storage overhead of coefficients in LUT. The CSD value like $[1\ 0\ -1\ 0\ -1\ 0\ 1\ 0\ -1]$ can be stored in an LUT like $[01\ 00\ 11\ 00\ 11\ 00\ 01\ 00\ 11]$ with '00' corresponding to 0, '01' corresponding to 1 and '11' corresponding to -1. Therefore, for the worst-case scenario, an 8-bit CSD coefficient requires 16 bits storage space in LUT. Although the bit-length requirement can be optimized as no adjacent bits in CSD are ones, the CSD still requires more number of bits than binary to store the same value in LUT. On the other hand, since all the bits in binary representation are positive, there is no storage overhead. Thus the additional half-adders required for implementing the subtractor circuit and the additional LUT storage space required for CSD will increase the area and reduce the speed of operation of the CSD-based reconfigurable FIR filters compared to binary representation-based FIR filter implementations. The area and speed savings achieved using the binary representation-based FIR filters over the CSD-based counterpart become significant as the filter order increases, which is the case in SDR channel filters.

The synthesis has been done using *Synopsys*[®] tool for all the FIR filter specifications mentioned in Section 5.1 on 0.18 μ m CMOS technology. The synthesis results for the 20-tap FIR filter with 16-bit coefficient wordlength, whose specifications as mentioned in Section 5.1, are summarized in Table 5.4. The proposed CSM and PSM architectures which employ binary representation of filter coefficients are denoted as BCSM and BPSM respectively. The CSD-based implementations of CSM and PSM are denoted as CSD-CSM and CSD-PSM respectively. Table 5.4 shows that the CSD-CSM and CSD-PSM architectures consume more area, power and has less speed compared to the binary representation based BPSM and BCSM architectures. The BCSM architecture has area reductions of 10% and 1% over CSD-CSM and CSD-PSM architectures respectively and the area reductions for BPSM architecture over CSD-CSM and CSD-PSM architectures are 15% and 7% respectively. The improvement in the speed of operation for the BCSM architecture over the CSD-CSM and CSD-PSM architectures are 10% and 22% respectively. The BPSM architecture offers an improvement in the speed of operation of 4% and 12% over the CSD-CSM and CSD-PSM architectures respectively.

Table 5.4
Synopsys Synthesis results for 20-tap FIR Filter implementation of Section 5.3.2

	Proposed BPSM	Proposed BCSM	CSD- CSM	CSD- PSM	FIR Filter [58]
Area (mm ²)	0.2594	0.275	0.304	0.2796	0.5467
Delay (ns)	8.2	7.67	8.5	9.34	15.6
Dynamic Power (mW)	5.98	7.8	10	13.97	16

The dynamic power reductions for the BCSM architecture are 22% and 44% over the CSD-CSM and CSD-PSM architectures respectively. The BPSM architecture offers the dynamic power reductions of 40% and 57% over the CSD-CSM and CSD-PSM architectures respectively. The BPSM architecture offers area and power reductions of 6% and 23% over the BCSM architecture respectively. The BCSM architecture offers an improvement in the speed of operation by 7% compared to the BPSM architecture. In Table 5.4, the proposed architectures are also compared with the multiplexed multiple constant multiplications (MMCM) architecture based FIR filter in [58]. The proposed BCSM architecture offers an area reduction of 49.7%, power reduction of 51.3% and a speed improvement of 50.8% over the MMCM [58]. The area and power reductions offered by the BPSM architecture over MMCM [58] are 52.7% and 62.5% respectively with an improvement in speed of 47.7%. It must be noted that the MMCM [58] architecture is limited to maximum filter length of 40 whereas no such restrictions exist for the proposed architectures.

In [54], a low power CSD-based digit reconfigurable FIR filter was proposed. The architecture in [54] performs multiplication using direct shift and add method and hence will result in low speed operation. If a coefficient of n -bit wordlength is considered, then the LD for method in [54] is n adder-steps as it is digit based. But for the CSM architecture, the LD is only $\lceil (\log_2(n/3 - 1)) + 2 \rceil$. Also for the PSM architecture, the LD is $\lceil (\log_2 b) + 2 \rceil$, where b is the number of non-zero operands in the worst-case coefficient after the application of BSE as explained in Section 5.2. In both CSM and PSM, '2' denotes the LD of the 3-bit BCS based shift and add unit. Note that b is considerably less than n in the PSM as common subexpressions are used. Consequently, the LD of the PSM is also less. It can be seen that, the digit processing unit (DPU) in [54] is used in series to

form the processing element. Thus for a higher-order filter, the number of DPUs will be significantly large, which would delay the filtering operation substantially. The proposed architectures employ CSE techniques and hence a much faster filtering operation for any filter-length is feasible. Also the method in [54] is CSD-based which has many inherent difficulties, as explained before. The Table 5.5 shows the comparison of a 32-tap FIR filter with coefficient wordlength of 8 bits using the BCSM and BPSM architectures and the FIR filter given in [54] and [57]. The passband and stopband specifications of the filter are chosen as $\omega_p = 0.2\pi$ and $\omega_s = 0.3\pi$ respectively. It can be seen from Table 5.5 that the BCSM architecture offers an area reduction of 81% over the architecture in [54] with a reduction in power consumption of 38.8% and increase in speed of 38.5%. Table 5.5 also shows that the BPSM results in area reduction of 83.7%, power consumption reduction of 45.9% and increase in speed of 29.7% over the architecture in [54]. Compared to the architecture in [57], the BCSM offers area reduction of 81%, power reduction of 20% and speed improvement of 48.8% whereas the BPSM offers area and power reductions of 82.8% and 29.2% respectively, and a speed improvement of 41.4%.

Table 5.5
Synopsys Synthesis results for 32-tap FIR Filter implementation of Section 5.3.2

	Proposed BPSM	Proposed BCSM	FIR Filter [54]	FIR Filter [57]
Area (mm ²)	0.245	0.27	1.47	1.394
Delay (ns)	4.2	3.67	5.97	7.17
Power (mW)	4.3	5.2	8.5	6.5

Table 5.6 shows the comparison of an 18-tap FIR filter with a coefficient wordlength of 10 bits using the BCSM and BPSM architectures and the FIR filter given in [55]. The passband and stopband specifications of the filter are chosen as $\omega_p = 0.1\pi$ and $\omega_s = 0.3\pi$ respectively. It can be seen from Table 5.6 that, the area of the filter architecture in [55] is almost 100% more than the BPSM and BCSM architectures. The reason for the increase in area and delay of [55] is due to the employment of Booth encoding schemes, Wallace adder trees, etc.

Table 5.6
Synopsys Synthesis results for 18-tap FIR Filter implementation of Section 5.3.2

	Proposed BPSM	Proposed BCSM	FIR Filter [55]
Area (mm ²)	0.137	0.14	13.872
Delay (ns)	5.7	5.47	7

Thus from all the above comparisons, it can be concluded that the proposed BCSM and BPSM architectures are equally suitable for higher and lower order filters and outperform existing reconfigurable filter implementations in terms of area, power consumption and speed.

5.3.3 Design Results

In this example, FIR filters employed in the filter bank channelizer of D-AMPS receiver in [22] are considered. The sampling rate chosen is 34.02 MHz as in [22]. The channel filters extract 30 kHz D-AMPS channels from the input signal after down sampling by a factor of 350. The passband and stopband edges are 30 kHz and 30.5 kHz respectively. The peak passband ripple is chosen as 0.1 dB. The filter stop-band specifications are chosen according to the D-AMPS standard [65]. The length of the FIR filter, N , is determined using (4.12). The peak stopband ripple is chosen as -24 dB and the transition bandwidth is fixed as 0.01. The optimum filter length is obtained as 350 using expression (4.12). The coefficient wordlength used in this implementation is 16 bits.

The synthesis of the above channel filter has been done on 0.18 μ m CMOS technology. The synthesis results for a 350-tap FIR filter with 16-bit coefficient wordlength are shown in Table 5.7. It can be seen that proposed CSM and PSM architectures offer considerably better results compared to the CSD and CSD-CSE architectures as well as the architectures in [54] and [57]. The BCSM architecture has area reductions of 17% and 6.6% over CSD-CSM and CSD-PSM architectures respectively and the area reductions for BPSM architecture over CSD-CSM and CSD-PSM architectures are 22.8% and 12.2% respectively. The speed improvement for the BCSM architecture over the CSD-CSM and CSD-PSM architectures are 32.8% and 40.8%

respectively.

Table 5.7
Synopsys Synthesis results for D-AMPS channel filter implementations

	Proposed BPSM	Proposed BCSM	CSD- CSM	CSD- PSM	FIR Filter [54]	FIR Filter [57]
Area (mm ²)	4.531	4.82	5.82	5.17	14.08	12.37
Delay (ns)	9.87	9.08	13.5	15.34	17.45	19.65
Power (mW)	48.3	84	105	114.9	120.7	102.4

The BPSM architecture offers speed improvement of 26.9% and 35.66% over the CSD-CSM and CSD-PSM architectures respectively. The dynamic power consumption reductions for the BCSM architecture are 20% and 24% over the CSD-CSM and CSD-PSM architectures respectively. The BPSM architecture offers dynamic power reductions of 53.3% and 57.4% over the CSD-CSM and CSD-PSM architectures respectively. From Table 5.7, it can be seen that the proposed BPSM offers area reduction of 67.8%, power reduction of 60% and speed improvement of 43.4% over the method in [54] and these reductions are 63.4%, 52.8% and 49.8% respectively over the method in [57]. The proposed BCSM offers area reduction of 65.8%, power reduction of 30.4% and speed improvement of 48% over the method in [54]. The respective reductions are 61%, 18% and 53.8% over the method in [57]. It can be noted that the proposed BPSM and BCSM architectures are significantly better than the architectures in [54] and [57] in terms of area, power and speed. The BPSM architecture offers area and power reductions of 6% and 42% over the BCSM architecture respectively. The BCSM architecture offers in speed improvement of operation by 8% compared to the BPSM architecture.

The main difference of the proposed CSM architecture from the architectures in [59, 60] is the use of the BCSs-based shift and add unit and hardwiring of shifts. In the architectures proposed in [59, 60], there are pre-computers which are used to generate x , $3x$, $5x$, $7x$, $9x$, $11x$, $13x$ and $15x$ using 9 adders employing a special carry select adder, where x is the input signal. The other products such as $2x$, $4x$, $6x$, $8x$, $10x$, $12x$ and $14x$ can be generated using simple shifts and no extra adders are required. This is in comparison with only 7 adders required by the proposed 4-bit BCSs-based shift and add unit. Thus the CSM architecture offers adder reduction over the architectures in [59, 60] and is different from the latter ones because it employs BSE-based shift and add unit for complexity

reduction. Another major difference is that [59, 60] employ two programmable shifters named *SHIFTER* and *ISHIFTER* with coefficient values as select values. The shifters were used to identify the most significant non-zero bit (digit) in each filter coefficient. These shifters should always be preceded by 8:1 multiplexers in [59] and hence the multiplexer complexity is also not reduced. These programmable shifters will reduce the overall speed of operation of the resulting filters especially for higher-order channel filter applications in wireless communication receivers. In the proposed CSM architecture, all the shifts are constants and hence can be hardwired using a constant propagation tool and hence results in better speed of operation compared to methods in [59, 60]. This can be clarified using an example. For a 16-bit coefficient, the proposed CSM (3-bit BCSs-based shift and add unit) architecture requires five 8:1 multiplexers and one 2:1 multiplexer (equivalent to twenty one 2:1 multiplexers), eight adders (3 adders for shift and add unit and 5 adders for the final adder unit). Note that programmable shifters are not required in CSM since all shifts are constants which can be hardwired. On the other hand, the approach in [59, 60] requires four 8:1 multiplexers (main multiplexers) + four 4:1 multiplexers (for programmable shifters) (equivalent to twenty four 2:1 multiplexers), twelve adders (9 adders for precomputers and 3 adders for final adder unit) and eight programmable shifters. From above example, it is evident that the CSM is less complex than the methods in [59, 60].

The comparison on the number of addition operations needed to implement the coefficient multipliers required for proposed BPSM and BCSM and those for CSD-based CSD-CSM and CSD-PSM implementations for different filter lengths for the D-AMPS channel filter of 16-bit coefficient wordlength has been done. The Fig. 5.7 shows the result of this comparison. The PSM and CSM offer an average reduction of 23% and 15% respectively in the number of addition operations compared to the CSD-CSE based methods.

Figure 5.6 Comparison of the number of addition operations for implementing filters with different filter lengths and coefficient wordlength of 16 bits.

5.4 Implementation Results

The proposed CSM and PSM architectures for a 20-tap FIR filter with 16-bit coefficient wordlength have been implemented on Xilinx's Virtex-II 2v3000ff1152-4 FPGA associated with the dual DSP-FPGA Signalmaster kit provided by Lyrtech[®] [74]. A model based design using Matlab's Simulink[®] and Xilinx's System generator[®] was employed for the implementation purpose as shown in Fig. 5.7. Fig. 5.7 consists of eight blocks whose details are given below:

1) Multi-tone Input Signal: A multi-tone input signal was generated by summing up sine waves of frequencies 300 Hz, 1000 Hz, 2500 Hz, 3500 Hz and 4200 Hz, each sampled at 10 MHz. Note that the signal frequencies and the sampling frequency in this example are only for illustration purpose. By dynamically changing the input frequencies using the function in Simulink, it was verified that the CSM and PSM architectures work well for frequencies of several tens of MHz.

2) Lyrtech Signal Master Controller: Lyrtech signal master controller consists of three components: 1) The board configuration for configuring the FPGA/DSP and for downloading the bitstream to FPGA, 2) Xilinx's system generator for generating the bit stream to be downloaded to the FPGA and 3) Log viewer which gives implementation information about the area and delay of the used architecture. Xilinx's XPower can be employed for the calculation of power dissipation.

Figure 5.7 Implementation of the proposed CSM and PSM architecture on Virtex 2v3000ffl152-4 FPGA.

3) Coefficient Controller: The CSM and PSM architectures have been implemented with the provision of dynamic changing of filter coefficients. This was made possible with the help of a multipoint switch. Based on the select signal values, the switch will select one of the inputs, h_2 to h_6 , which are the coefficients stored in LUTs, generated using Matlab program according to the specifications of the architecture as explained in Sections 5.1.1 and 5.1.2. As shown in Fig. 5.7, h_2 to h_6 are low-pass filters with cutoff frequencies 50 Hz, 1 kHz, 2 kHz, 3 kHz and 4 kHz respectively with sampling frequency fixed as 10 MHz. For example, if the constant value is 4 (as shown in Fig. 5.7), the filter with specification h_5 with cutoff 3 kHz will be chosen. Simulink[®] provides option to change the constant value dynamically. Hence if the constant value is changed to 1, the

coefficient h_2 with cutoff frequency of 500 Hz will be chosen and the output of the filter changes dynamically. This scheme was used to achieve dynamic reconfigurability. In this illustrative example, only five filter specifications from h_2 to h_6 have been employed, but it is possible to include filters for additional specifications.

4) Coefficient Extractor: The coefficient extractor is used to extract the coefficients individually and to provide the extracted coefficient to each processing element of the proposed CSM and PSM architectures.

5) Gateways: Gateways are employed as an interface between Xilinx's blocks which are used for developing the proposed CSM and PSM architectures. Thus gateways provide the connection of the bitstream file with input sources and output sinks of the Simulink environment.

6) Simulation architecture: This forms the CSM and PSM architectures developed as shown in Fig. 5.4 and Fig. 5.5 respectively using *Xilinx* block set in *Simulink* library. The simulation architecture is used as a reference for comparing with the hardware implementation on FPGA (bit stream running on FPGA).

7) Bit stream running on FPGA: The bit stream of the simulation architecture has been generated using the *Xilinx* system generator. The generated bitstream can be downloaded to FPGA and it appears as a 'Lyrtech Cosim Engine' block. The performances of the bitstream and the simulation architecture were checked to ensure that they are identical.

8) Fast Fourier transform (FFT) of outputs: The FFT block was employed for observing the outputs of the simulation and FPGA architectures. The FFT block plots the energy of the output in dB against the frequencies. The implementation results are shown in Table 5.8. The area and delay results are obtained using log viewer block shown in Fig. 5.7. The power dissipation is obtained using Xilinx XPower[®]. The results show that the proposed CSM results in low delay compared to proposed PSM whereas the latter results in low area and power implementations. From Table 5.8, it can be concluded that the CSM architecture results in an improvement in speed of 5.3% over the PSM architecture, whereas the PSM architecture results in area and power reductions of 8.2% and 8% over CSM architecture respectively.

Table 5.8
Implementation results for proposed architectures with 20 taps and 16-bit coefficient wordlength

	Proposed PSM	Proposed CSM
LUTs	1527	1693
Slices	896	1024
Flip-flops	790	756
Data Arrival Time (ns)	40.824	38.672
Power Dissipation (mW)	375.45	408.29

5.5 Summary

In this chapter, two new approaches namely, constant shifts method (CSM) and programmable shifts method (PSM), have been proposed for implementing reconfigurable higher-order filters for SDR receivers with low complexity. In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, the proposed CSM and PSM based reconfigurable FIR filters use BCSs-based shift and add unit in the proposed CSM and PSM architectures. Thus significant number of adders are saved in proposed filter architectures when compared to other reconfigurable architectures in literature. The CSM architecture results in high speed filters and PSM architecture results in low area and thus low power filter implementations. The PSM also provides the flexibility of changing the filter coefficient wordlengths dynamically. The proposed architectures have been implemented on Virtex-II 2v3000ff1152-4 FPGA and 0.18 μ m CMOS technology with a high coefficient wordlength of 16 bits and compared with other reconfigurable FIR filter architectures in literature. Design example of a channel filter shows that the PSM architecture offers an average reduction of 23% in the number of addition operations compared to other FIR filter implementations. The proposed reconfigurable architectures can be easily modified to employ any CSE method. Thus the proposed method is a general approach for low complexity reconfigurable channel filters. In this chapter, the main objective was to incorporate reconfigurability into the BSE algorithm proposed in Chapter 4, to implement reconfigurable FIR filters with low complexity. However the FIR filters used in this chapter have very high order as they are designed using conventional techniques such as Parks-McClellan technique. In the next chapter, a technique known as frequency response masking (FRM) [75] is employed to reduce the order of the filter. FRM was originally proposed to design sharp transition-

band fixed coefficient FIR filter, as required in SDR receivers, with low complexity. The main contribution is in the incorporation of reconfigurability into the inherently less complex FRM based architecture to implement extremely low complexity reconfigurable FIR filters and filter banks.

Chapter 6

Reconfigurable Low Complexity Filter Banks based on Frequency Response Masking Technique

In this chapter, reconfigurable low complexity filters and filter banks based on a frequency response masking (FRM) technique [75] are proposed. The filters based on the FRM technique have lower-order and thus low complexity when compared to filters designed using conventional finite impulse response (FIR) filter design techniques such as Parks-McClellan technique. The FRM technique was originally proposed for the design of sharp transition-band FIR filters with very low complexity [75]. The basic idea was to compose the over-all sharp transition-band filter using three wide transition-band subfilters. Since the subfilters have wide transition-band specifications, they can be of lower-order. Consequently, the over-all complexity of FRM-based filter will be much less than sharp transition-band FIR filters designed using conventional methods. The work in this chapter focuses on integrating reconfigurability into the FRM-based filter, a research problem not addressed in literature, but necessary for software defined radio (SDR) channelizers. The proposed reconfigurable filter bank overcomes certain shortcomings of existing filter banks such as simultaneous extraction of non-uniform bandwidth channels, extremely narrowband channels and multi-mode operation. The proposed filter and filter bank have the flexibility of changing frequency responses dynamically with very low hardware overhead. In SDR channelizers, the filter banks may have to extract channels, whose spacings are not related by integer factors. Based on the FRM approach, a new filter bank has also been proposed, which can extract channels whose spacings are related by fractional factors.

6.1 Review of Frequency Response Masking (FRM) Technique

In conventional FIR filter designs, higher-order filters are required to obtain sharp transition-band frequency response. The complexity of FIR filters increases with the

filter order. In [75], the FRM technique was employed for the synthesis of sharp transition-band FIR filters with low complexity. The basic idea behind the FRM technique is to compose the over-all sharp transition-band filter using several wide transition-band subfilters. The advantage of FRM technique is that, the bandwidths of the filters are not altered and the resulting filter will have many sparse coefficients resulting in less complex filters. In this work, architectural modifications to inherently less complex sharp transition-band FRM filter are proposed for integrating reconfigurability in channel filters in an SDR channelizer. Given a prototype symmetrical impulse response linear phase low-pass filter $H_a(z)$ of odd length N_a , its complementary filter $H_c(z)$ can be expressed as

$$H_c(z) = z^{-\frac{(N_a-1)}{2}} - H_a(z) \quad (6.1)$$

$H_a(z)$ is also known as ‘modal filter’. Replacing each delay elements of both filters by M delays, two filters with transfer functions $H_a(z^M)$ and $H_c(z^M)$ are formed. The transition-band widths of $H_a(z^M)$ and $H_c(z^M) = z^{-(M(N_a-1))/2} - H_a(z^M)$ are a factor of M narrower than that of $H_a(z)$. In the FRM technique, two filters $H_{Ma}(z)$ and $H_{Mc}(z)$, are cascaded to $H_a(z^M)$ and $H_c(z^M)$, respectively as shown in Fig. 6.1. The transfer function of the entire filter is given by

$$H(z) = H_a(z^M)H_{Ma}(z) + H_c(z^M)H_{Mc}(z) \quad (6.2)$$

Figure 6.1 FIR filter architecture based on FRM technique.

Note that the group delay of the filters $H_{Ma}(z)$ and $H_{Mc}(z)$ must be equal, and $M(N_a-1)$ in equation (6.2) must be an even number. The design steps for the subfilters in Fig. 6.1 involve the solution of the expressions [75]:

$$m = \lfloor f_p M \rfloor \quad (6.3 \text{ (a)})$$

$$f_{ap} = f_p M - m \quad (6.3 \text{ (b)})$$

$$f_{as} = f_s M - m \quad (6.3 \text{ (c)})$$

$$f_{map} = f_p \quad (6.3 \text{ (d)})$$

$$f_{mas} = \frac{m+1-f_{as}}{M} \quad (6.3 \text{ (e)})$$

$$f_{mcp} = \frac{m-f_{ap}}{M} \quad (6.3 \text{ (f)})$$

$$f_{mcs} = f_s \quad (6.3 \text{ (g)})$$

where $\lfloor x \rfloor$ denotes the largest integer less than x , M is the up-sampling rate for $H_a(z)$, f_p and f_s are the passband and stopband edges of the overall filter, f_{ap} and f_{as} are the passband and stopband edges of the modal filter $H_a(z)$, f_{map} and f_{mcp} are the passband edges and f_{mas} and f_{mcs} are the stopband edges of the two masking filters respectively. All the stopband and passband edges mentioned in this chapter including expressions (6.3) are normalized to unity. Thus by suitable selection of the passband and stopband edges of the modal and the masking filters, any sharp transition-band FIR filter can be implemented with low complexity [75].

Figure 6.2 Frequency Response illustration of FRM approach.

The FRM approach can be more clearly explained with the help of frequency response illustration shown in Fig. 6.2. Fig. 6.2 (a) represents the frequency response of a low-pass filter $H_a(z)$. The passband and stopband edges of the modal filter are f_{ap} and f_{as} respectively. The complementary filter of the modal filter, $H_c(z)$, is shown in Fig. 6.2 (b). Replacing each delay of $H_a(z)$ and $H_c(z)$ by M delays, two filters $H_a(z^M)$ and $H_c(z^M)$ are obtained, and their frequency responses are shown in Fig. 6.2 (c). Two masking filters $H_{Ma}(z)$ and $H_{Mc}(z)$ as shown in Fig. 6.2 (d), are used to mask $H_a(z^M)$ and $H_c(z^M)$ respectively. If the outputs of $H_{Ma}(z)$ and $H_{Mc}(z)$, are added, as shown in Fig. 6.1, the frequency response of the resulting filter, $H(z)$, is shown in Fig. 6.2 (e). Thus a sharp transition-band FIR filter is obtained using four subfilters. Since these subfilters have wide transition-band specifications, the overall complexity will be much less than conventional design of sharp transition-band FIR filters.

6.2 Proposed Reconfigurable Channel Filter

In this section, a method is proposed to realize reconfigurable FRM filters by modifying the conventional FRM technique which was originally intended for realizing fixed-coefficient filters. The proposed reconfigurable FRM technique is combined with the BSE technique proposed in Chapter 4 to further reduce the filter complexity. The proposed multi-mode channel filter offers reconfigurability at architecture and filter levels.

6.2.1 Architecture Level Reconfigurability

The architectural reconfigurability of the channel filter can be illustrated using the expressions (6.3). For ease of explanation, a dual-mode channelizer is considered, although the proposed architecture is not restricted to two modes of operation. Let f_{p1} and f_{s1} represent the passband and the stopband frequencies of the channel filter corresponding to one of the communication standards (modes), and f_{p2} and f_{s2} , the respective frequencies of the other standard. Reconfigurability can be achieved by using the same subfilters shown in Fig. 6.1 for both the standards. The parameters f_{ap} and f_{as} remain unchanged for both the standards, i.e., the same modal filter is employed for both the standards and the masking filters can be reconfigured by swapping the LUT values. Thus,

$$f_{p1}M_1 - \lfloor f_{p1}M_1 \rfloor = f_{p2}M_2 - \lfloor f_{p2}M_2 \rfloor \quad (6.4)$$

$$f_{s1}M_1 - \lfloor f_{s1}M_1 \rfloor = f_{s2}M_2 - \lfloor f_{s2}M_2 \rfloor \quad (6.5)$$

where M_1 and M_2 denote the up-sampling factor for the two standards which can be obtained by solving (6.4) and (6.5). Thus by changing the number of delay elements, it is possible to reconfigure the same modal filter to work for both the standards. The dual-mode channelizer can be extended to incorporate additional communication modes by choosing an appropriate number of delay elements. For example, if a third standard needs to be incorporated, then the number of delays or the up sampling factor (M) can be obtained by substituting the filter specification corresponding to the third standard in expressions (6.4) and (6.5). This computation of the number of delays can be done online or offline depending upon the necessity. Online computation would be required only when a new communication standard needs to be incorporated. When such a scenario arises, a software will compute the value of M required using (6.4) and (6.5) for the new standard and change the mode selector accordingly.

The architectural reconfigurability can be explained with the design example of a channelizer for $M = 2, 3, 4$ and 5 . Let the specifications of the modal filter be $f_{ap} = 0.2$ and $f_{as} = 0.25$. For $M = 2$ i.e., replacing each delay of the modal filter by 2 delays, expression (6.4) becomes $0.2 = 2f_p - \lfloor 2f_p \rfloor$ and expression (6.5) becomes $0.25 = 2f_s - \lfloor 2f_s \rfloor$. By solving the two expressions, $f_p = 0.1$ and $f_s = 0.125$ and let this channel be denoted by y_1 . In this case as $f_p < f_{ap}$ and $f_s < f_{as}$, the filter specifications of $f_p = 0.1$ and $f_s = 0.125$ can be obtained by masking the modal filter with a masking filter $H_{Mal}(z)$ with the wide transition-band specifications of f_{map} and f_{mas} of 0.1 and 0.375 respectively as obtained from expressions (6.3). In Fig. 6.1, the 2:1 multiplexer (Mux) is employed for obtaining the output either directly from the output of masking filter, $H_{Mal}(z)$, or from the output of the adder. In this case, the output y_1 is obtained from the output of $H_{Mal}(z)$. Since the transition-band is wide, the complexity of the masking filter is very low. The length of each of the filters, N , used in the architecture shown in Fig. 6.1, can be obtained from [77]:

$$N = -\frac{2 \log_{10}(10\delta_p\delta_s)}{3(f_s - f_p)} - 1 \quad (6.6)$$

where δ_p is the peak passband ripple, δ_s is the peak stopband ripple and $(f_s - f_p)$ is the transition width normalized to one. Let the values of $\delta_p = 0.1$ dB and $\delta_s = -40$ dB are fixed for all the subfilters in Fig. 6.1. By substituting the corresponding values in (6.6), the length of the modal filter for all the cases is obtained as $N_{modal} = 39$ and that of the masking filter $H_{Ma}(z)$ as $N_{HMa}(z) = 6$. The number of complementary delays is given by (6.7) [75]:

$$N_{delays} = \frac{((N_{modal} - 1)M)}{2} \quad (6.7)$$

Note that complementary delays are not required for obtaining y_1 .

The general procedure to obtain the specifications of $H_{Ma}(z)$, $H_{Mc}(z)$ and values of $N_{HMa}(z)$, $N_{HMc}(z)$ and N_{delays} are as follows.

Step 1) Solve the expressions (6.4) and (6.5) after substituting the values of $f_{ap} = 0.2$ and $f_{as} = 0.25$ and value of M to obtain the specifications f_p and f_s .

Step 2) Using (6.3), obtain the specifications of $H_{Ma}(z)$ and $H_{Mc}(z)$ substituting the values of f_p and f_s from step 1.

Step 3) Obtain the values of N_a , $N_{HMa}(z)$ and $N_{HMc}(z)$ using (6.6).

Step 4) Obtain the value of N_{delays} using (6.7).

Step 5) By employing the architecture in Fig. 6.1, obtain the overall filter with specifications of f_p and f_s from the output of the adder A_2 using the 2:1 Mux.

Table 6.1
Specifications of subfilters of Fig. 1 for different values of M

M	H(z)		H _{Ma} (z)			H _{Mc} (z)			N _{delays}
	f _p	f _s	f _{map}	f _{mas}	N _{HMa} (z)	f _{mcp}	f _{mcs}	N _{HMc} (z)	
2	0.1	0.125	0.1	0.375	6	NA	NA	NA	NA
3	0.4	0.415	0.4	0.584	12	0.267	0.415	12	57
4	0.3	0.312	0.3	0.438	16	0.2	0.312	16	76
5	0.24	0.25	0.24	0.35	21	0.16	0.25	21	95

Table 6.1 shows the specifications of the subfilters for different values of M , computed using above procedure. The specifications of the modal filter are $f_{ap} = 0.2$ and $f_{as} = 0.25$ and $N_{modal} = 39$ for all values of M . Thus it can be seen that, in all the cases, the modal filter is a fixed filter and only the number of delays, M , is changed. By masking using appropriate masking filters and employing the architecture in Fig. 6.1, filter specifications of $f_p = 0.1$ and $f_s = 0.125$, $f_p = 0.24$ and $f_s = 0.25$, $f_p = 0.3$ and $f_s = 0.312$ and $f_p = 0.4$ and $f_s = 0.415$ are obtained. The transition-band of the resulting filter specifications can be made narrower by choosing a much sharper transition-band modal filter. For example, if a modal filter with $f_{ap} = 0.2$ and $f_{as} = 0.22$ is chosen, the

resulting filter specification with $M = 2$ becomes $f_p = 0.1$ and $f_s = 0.11$. Obviously, the complexity of the modal filter will increase for very narrow transition-band specifications.

6.2.2 Filter Level Reconfigurability

The filter level reconfigurability has been achieved for FRM-based FIR filters by employing the PSM architecture proposed in Chapter 5. In Chapter 5, the CSM and PSM architectures for implementing channel filters have been proposed that are designed using conventional FIR filter design algorithm (Parks-McClellan). However, sharp transition-band FIR filters designed using conventional methods would require large number of taps (higher-order), and consequently increased hardware cost. On the contrary, the sharp transition-band FIR filters designed using FRM technique have substantially fewer number of taps, which would significantly reduce the cost and power consumption.

6.2.3 Proposed Filter Architecture

The basic architecture of the proposed channel filter is similar to the FRM filter architecture in Fig 6.1. The 2:1 Mux is used to obtain the output directly from the output of the masking filter (as in the case of y_1) or from the output of final adder (as in the case of y_2, y_3 and y_4). The mode selector input in Fig. 6.1 will select either of y_1, y_2, y_3 and y_4 at the output y . The architecture of modal filter is shown in Fig. 6.3. The length of the modal filter (N_{modal}) is chosen as 39 (computed in Section 6.2.1). The number of delays is fixed by the mode selector. For example, to obtain output channels y_1, y_2, y_3 and y_4 , the mode selector is fixed as '00', '01', '10' and '11' respectively so that number of delays, M , will be changed from 2 to 5 respectively. The same scheme is employed for obtaining the complementary outputs y_{C2}, y_{C3} and y_{C4} as shown in Fig. 6.4. Depending on the mode selector, the number of delays will be selected as '0' for y_{C1} (as no complementary delays are required for y_1), '57' for y_{C2} , '76' for y_{C3} and '95' for y_{C4} (as discussed in Section 6.3.2 and obtained from (6.7)).

Figure 6.3 Architecture of modal filter.

Figure 6.4 Architecture of complementary delays.

The masking filters $H_{Ma}(z)$ and $H_{Mc}(z)$ are implemented using the architecture shown in Chapter 5. The four-set coefficients of filters corresponding to four channel specifications from y_1 to y_4 are stored in LUT, and appropriate coefficient set is selected for the desired channel. Another flexibility of the architecture is that the number of taps of masking filters can be changed depending on the mode selector. For $M = 2$, the masking filter needs only 6 taps, whereas for $M = 3$, the number of taps required is 12 and so on (as discussed in Section 6.3.1). Depending on the mode selector, the number of taps is changed dynamically. For example, a 21-tap masking filter is required for obtaining y_4 . If the desired channel is y_1 , only a 6-tap masking filter is needed. The number of taps is changed dynamically to 6 in this case. This reconfiguration of filter-length offers savings in dynamic power consumption as unused taps are unloaded. Thus to obtain y_1 , the mode selector takes the value of '00' so that $M = 2$, no complementary delays are required as y_1 is obtained by directly masking the output of the modal filter y_{A1} with the masking filter having specifications as that of $H_{Ma1}(z)$. Hence the LUT will be switched to coefficients corresponding to

$H_{Ma1}(z)$. The output y_1 is taken directly from the output of $H_{Ma1}(z)$ by employing the 2:1 Mux as shown in Fig. 6.1. For obtaining y_2 , the mode selector takes the value of '01' so that $M = 3$, the number of complementary delays is chosen as $N_{delays} = 57$. The LUT corresponding to $H_{Ma}(z)$ is loaded with coefficients corresponding to $H_{Ma2}(z)$ and that of $H_{Mc}(z)$ with $H_{Mc2}(z)$. The outputs y_3 and y_4 can also be obtained in same way. Thus the same filter architecture can provide four set of channel specifications, $f_p = 0.1$ and $f_s = 0.125$, $f_p = 0.24$ and $f_s = 0.25$, $f_p = 0.3$ and $f_s = 0.312$ and $f_p = 0.4$ and $f_s = 0.415$, by changing the mode selector and the LUT values of $H_{Ma}(z)$ and $H_{Mc}(z)$. If filters for any new channel specifications are required, they can be easily implemented by changing the coefficient values in the LUT to that of the desired filter specifications. Thus it can be concluded that, by changing the value of mode-selector, different filter specifications such as y_1 to y_4 can be obtained at the output y as shown in Fig. 6.1.

6.2.4 Experimental Results

In this section, the design and synthesis results of the proposed channel filter architecture are presented.

6.2.4.1 Design Results

A dual-mode Code Division Multiple Access (CDMA)/Wideband CDMA (WCDMA) SDR channelizer is considered in this example. The sampling rate chosen is 12.5 MHz. The channel filters can extract 1250 kHz CDMA channels and 5 MHz WCDMA channels from the input signal. The passband and stopband edges are 1250 kHz and 1350 kHz respectively for CDMA and 5000 kHz and 5100 kHz for WCDMA. The peak passband ripple is chosen as 0.1 dB. The filter stop-band specification is chosen as -48 dB for both the standards. Although the passband and stopband ripples are chosen as same for both CDMA and WCDMA, the proposed architecture can handle different passband and stopband ripple specifications. This is achieved by designing the modal filter for the worst case ripples. The reconfigurability can be achieved as follows. Considering CDMA, a channel with specifications, $f_p = (1250/12500) = 0.1$ and $f_s = (1350/12500) = 0.108$ is required at the output. Now considering WCDMA, a channel with specifications $f_p = (5000/12500) = 0.4$ and $f_s = (5100/12500) = 0.408$ is required at the output. For WCDMA, mode selector of Figures 6.3 and 6.4 are chosen

as ‘11’ to select the output y_4 which has the specification of $f_p = 0.4$ and $f_s = 0.415$. The LUTs of masking filters FMA and FMC are loaded with coefficients corresponding to $FMA4$ and $FMC4$ respectively as discussed in Section 6.2.3 so that the WCDMA channels can pass through the channel filter. For CDMA, the mode selector is chosen as ‘00’ to obtain the output y_1 with specifications of $f_p = 0.1$ and $f_s = 0.125$. The LUTs of masking filters H_{Ma} and H_{Mc} are loaded with coefficients corresponding to H_{Ma1} and H_{Mc1} respectively, so that CDMA channels can pass through. Thus the proposed reconfigurable filter is capable of operating on CDMA which is a 2G standard and on WCDMA which is a 3G standard. In case none of the low-pass outputs y_1 to y_4 satisfy the requirements of a new communication standard, the modal filter specifications can be changed by changing the coefficients in the LUT as shown in Chapter 5.

6.2.4.2 Synthesis Results

The proposed filter has been synthesized on 0.18μm CMOS technology and implemented and tested on Virtex-II 2v3000ff1152-4 FPGA. The synthesis results of the proposed architecture is compared with the reconfigurable BSE method in Chapter 5 and a reconfigurable CSD-based CSE method designed by the authors using conventional Parks-McClellan algorithm. As realization of higher-order reconfigurable FIR filters is hardly addressed in literature, for comparison purpose, reconfigurable filter architectures (called CSD-CSM and CSD-PSM described in Chapter 5) are implemented by the authors using the CSE algorithm in [35] which was originally intended for fixed-coefficient filters. The synthesis results are shown in Table 6.2.

Table 6.2
Synopsys Synthesis results for CDMA/WCDMA reconfigurable channel filter implementations

	BPSM	BCSM	CSD-CSM	CSD-PSM	Proposed Filter
Taps	350	350	350	350	81
Area (mm ²)	4.53	4.82	5.82	5.16	2.34
Delay (ns)	9.87	9.08	13.5	15.34	5.98
Power (mW)	49	84	105	115	33.6

In Table 6.2, BCSM and BPSM are the CSM and PSM architectures in Chapter 5 whereas CSD-CSM and CSD-PSM are the architectures based on the CSD-CSE algorithm in [35] (Fixed-coefficients CSE algorithm in [35] modified using the CSM

and PSM in Chapter 5 to incorporate reconfigurability). The wordlengths of filter coefficients in all the architectures are fixed to 16 bits. The filter-length required for implementations of BPSM, BCSM, CSD-PSM and CSD-CSM, using conventional Parks-McClellan algorithm, are obtained from (6.6) and found to be 350. For the proposed FRM based channel filter, the total length of the filters in Fig. 6.1 for the worst-case (i. e., for obtaining the output for y_4) is 81. The proposed channel filter offers area and power reductions of 48.3% and 31.4% over BPSM, 51.5% and 60% over BCSM, 59.8% and 68% over CSD-CSM and 54.6% and 70.8% over CSD-PSM respectively. The proposed channel filter offers an improvement in speed of 39.4% over BPSM, 34.1% over BCSM, 55.7% over CSD-CSM and 61% over CSD-PSM. Clearly, the proposed filter offers considerable reductions of area, power and delay over the conventional filter design methods.

6.2.4.3 Implementation Results

The proposed channel filter has been implemented on Xilinx’s Virtex-II 2v3000ff1152-4 FPGA associated with the dual DSP-FPGA Signalmaster kit provided by Lyrtech [74]. A model based design using Matlab’s Simulink and Xilinx’s System generator was employed for the implementation purpose. The area and delay results are obtained using log viewer block provided by Lyrtech. The power consumption is obtained using Xilinx’s XPower®. The implementation results are shown in Table 6.3. The sampling frequency represents the maximum speed at which the corresponding architecture can be operated. It can be seen that the proposed channel filter offers average area reduction of 38% over BCSM and 40.5% over BPSM and power reduction of 61.4% over BCSM and 57.95% over BPSM respectively.

Table 6.3
Implementation results

	Proposed Channel Filter	BPSM	BCSM
LUTs	12228	21897	22345
Slices	8936	12678	12999
Flip-flops	3842	5678	6643
Data Arrival Time (ns)	15.78	42.824	38.672
Sampling Frequency (MHz)	12	12	12
Power Dissipation (mW)	157.89	375.45	408.9

In this section, the FRM technique was employed to implement low complexity reconfigurable FIR filters for SDR receivers. In the next section, the proposed methodology is extended to implement low complexity reconfigurable filter banks.

6.3 Proposed Reconfigurable Filter Bank

In an SDR receiver, the specifications of the channelizer changes as the mode of communication changes. In conventional multi-mode channelizers [78, 79] a separate channelizer is needed for each mode, and reconfigurability is achieved by switching among distinct channelizers. This is not an efficient approach due to its increased hardware complexity and poor resource utilization. In this section, a reconfigurable filter bank channelizer is presented, which uses two blocks: a common hardware block at the front-end (modal filter) for multiple communication standards and a reconfigurable masking filter at the back-end. The complexity of the channelizer is dominated by the block at the front-end (as the order of the modal filter is substantially higher than that of the masking filter). Since the front-end hardware in the proposed scheme is same (common) for all the communication standards, its complexity can be significantly reduced using the BSE method presented in Chapter 4. The proposed reconfigurable filter bank channelizer is based on the FRM technique [75]. The conventional FRM technique employs the DF structure and therefore the critical path delay is proportional to filter-length. But if TDF is employed, the critical path delay can be made independent of the filter-length. Hence the TDF has been used for the proposed approaches, which will result in filters with low delay compared to DF structure. In order to further reduce the complexity of FRM filters, the BSE algorithm presented in Chapter 4 has been employed. The proposed channelizer also offers reconfigurability at two levels namely 1) Architectural level and 2) Filter Level.

6.3.1 Architectural Reconfigurability

The architectural reconfigurability can be illustrated with the design of a FB using $M = 2, 3, 4$ and 6 , where M is the delay. Let the specifications of the modal filter be $f_{ap} = 0.2$ and $f_{as} = 0.3$. For $M = 2$, from expressions (6.4) and (6.5), $0.2 = 2f_p - \lfloor 2f_p \rfloor$ and $0.3 = 2f_s - \lfloor 2f_s \rfloor$. By solving these two expressions, the multiple values such as $f_p = 0.1$ and $f_s = 0.15$ or $f_p = 0.6$ and $f_s = 0.65$ etc are obtained. But according to Nyquist criterion, the frequency edges cannot be greater than 0.5 . Hence the only possible frequency band edge specification is $f_p = 0.1$ and $f_s = 0.15$. In this case, as $f_p < f_{ap}$ and $f_s < f_{as}$, the filter specifications of $f_p = 0.1$ and $f_s = 0.15$ can be obtained by masking the modal filter with a masking filter that has wide transition-band specifications of f_{map} and

f_{mas} of 0.1 and 0.35 respectively, as obtained from expressions (6.3). Since the transition-band of the masking filter is sufficiently wide, the filter order and consequently the complexity of the masking filter are very low. It should be noted that, if the complementary output is also considered (dotted response in Fig. 6.2), it is possible to obtain many other frequency edges. Thus expressions for obtaining the values of f_p and f_s can be generalized as follows

$$f_p = \frac{f_{ap} + n}{M}, f_s = \frac{f_{as} + n}{M} \tag{6.8}$$

$$f_p = \frac{(n+1) - f_{as}}{M}, f_s = \frac{(n+1) - f_{ap}}{M} \tag{6.9}$$

where $n = 0, 1, 2$ etc till $f_p \leq 0.5$. Expression (6.8) gives possible frequency edges at the modal filter output (replacing each delay of modal filter by M delays) and (6.9) represents possible frequency edges at the complementary output. Hence for $M = 2$, $f_p = 0.1$ and $f_s = 0.15$. For other cases, $M = 3, 4$ and 6 , the results are shown in Table 6.4. In Table 6.4, all the possible frequency edges are shown for each value of M . In this particular design, the aim is to obtain four basic low-pass bands in the range 0 to 0.1, 0 to 0.2, 0 to 0.3 and 0 to 0.4 using a single architecture. Hence the final specifications obtained according to the above specific requirement are also shown in Table 6.4. The specifications of masking filters obtained from expression (6.3) are also shown in Table 6.4.

Table 6.4
Frequency specifications for the proposed filter bank

M	Possible Frequency edges		Masking Filter Specifications H_{Ma}		Masking Filter Specifications H_{Mc}		Over-all Frequency Specifications	
			f_{map}	f_{mas}	f_{mcp}	f_{mcs}		
2	0.1	0.15	0.1	0.35	-	-	0.1	0.15
	0.3	0.4						
3	0.067	0.1	0.4	0.567	0.267	0.433	0.4	0.433
	0.233	0.267						
	0.4	0.433						
4	0.05	0.075	0.3	0.425	0.2	0.325	0.3	0.325
	0.175	0.2						
	0.3	0.325						
	0.425	0.45						
6	0.033	0.05	0.2	0.284	0.134	0.217	0.2	0.217

Thus it can be seen that, by using the same modal filter and changing the number of delays, M , and using appropriate masking filters, filter specifications of $f_p = 0.1$ and $f_s =$

0.15, $f_p = 0.2$ and $f_s = 0.217$, $f_p = 0.3$ and $f_s = 0.325$ and $f_p = 0.4$ and $f_s = 0.433$ are obtained. The transition-band of the resulting filter specifications can be narrowed by choosing a much sharper transition-band modal filter. For example, if a modal filter with $f_{ap} = 0.2$ and $f_{as} = 0.25$ is chosen, the resulting filter specification with $M = 2$ becomes $f_p = 0.1$ and $f_s = 0.125$. Note that the complexity of the modal filter will also increase when the transition-band becomes narrower. By careful examination of the filter specifications $f_p = 0.1$ and $f_s = 0.15$, $f_p = 0.2$ and $f_s = 0.217$, $f_p = 0.3$ and $f_s = 0.325$ and $f_p = 0.4$ and $f_s = 0.433$, it can be noted that subtracting the output of the filter with $f_p = 0.1$ and $f_s = 0.15$ from the filter with specifications $f_p = 0.2$ and $f_s = 0.217$ would produce a band-pass output. It must be noted that the total length of both the filters must be the same to make the group delay identical. A more detailed analysis on this is given in Section 6.3.3. When all the possibilities are considered, since there are 4 filter specifications available, $C_4^2 = \frac{4!}{2! \times (4-2)!} = 6$ additional filter specifications can be obtained. This means that by using $M = 2, 3, 4$ and 6 (i. e., 4 delays) and two masking filters for each delay, $4+6 = 10$ filter specifications are achieved. In other words, the proposed filter bank architecture illustrated in this example can work as 10-mode channelizer. *In general, with 'M' delays, a total of $M + C_M^2$ - mode channelizer is possible.*

It is possible to design the proposed FB to obtain desired frequency band edge specifications at the output. Assume that the requirement is to obtain n -low-pass channels with passband edges at $\Omega_{p1}, \Omega_{p2}, \dots, \Omega_{pn}$ and stopband edges at $\Omega_{s1}, \Omega_{s2}, \dots, \Omega_{sn}$. Let the passband and stopband ripples be $\delta_{p1}, \delta_{p2}, \dots, \delta_{pn}$ and $\delta_{s1}, \delta_{s2}, \dots, \delta_{sn}$ respectively. In order to achieve the desired stopband and passband ripples, the modal filter can be designed with most stringent of the ripple specifications. Expressions (6.8) and (6.9) give the possible passband and stopband edges for a specific delay, M , and specific modal filter specifications, f_{ap} and f_{as} . Thus the desired passband and stopband edges $\Omega_{p1}, \Omega_{p2}, \dots, \Omega_{pn}$ and $\Omega_{s1}, \Omega_{s2}, \dots, \Omega_{sn}$ can be obtained by solving (6.8) and (6.9). A more specific illustration can be done based on Table 6.4. In Table 6.4, let $f_{ap} = 0.2$ and $f_{as} = 0.3$. For $M = 2$, two low-pass channels are possible, and for $M = 3$, three channels are possible and so on. Thus by using appropriate masking filter

specifications, it is possible to extract the desired low-pass channel. The generalized design procedure is as follows:

1. Design the modal filter with specifications of f_{ap} and f_{as} and passband and stopband ripple satisfying the most stringent requirement from $\delta_{p1}, \delta_{p2}, \dots, \delta_{pn}$ and $\delta_{s1}, \delta_{s2}, \dots, \delta_{sn}$.
2. Solve $\Omega_{pi} = \frac{f_{ap} + n}{M}$ or $\Omega_{pi} = \frac{(n+1) - f_{as}}{M}$ and $\Omega_{si} = \frac{f_{as} + n}{M}$ or $\Omega_{si} = \frac{(n+1) - f_{ap}}{M}$ for the desired value of M . (If not found, then change the modal filter specifications of f_{ap} and f_{as} and recalculate M .)
3. Solve for the masking filter specifications using expressions:
$$m = \lfloor f_p M \rfloor, f_{map} = f_p, f_{mas} = \frac{m+1 - f_{as}}{M},$$

$$f_{mcp} = \frac{m - f_{ap}}{M}, f_{mcs} = f_s$$
4. Realize the architecture as shown in Fig. 6.1.

The main advantage of the proposed channelizer based on FRM architecture is that it can extract channels with non-uniform bandwidth which is not possible with DFTFB or its modifications. Also a channel with very small bandwidth (extremely narrowband channel) can be easily selected using the proposed FB by selecting an appropriate delay value of M . In the previous example, if the output of the modal filter with $M = 6$ is directly masked and obtained the output, the resulting filter will have a very narrow transition-band of 0.067 corresponding to specifications, $f_p=0.033$ and $f_s=0.1$. Conventional channelizers based on DFTFB would require very high-order prototype filter to meet such a stringent narrow transition-band specification. Furthermore, as non-uniform bandwidth extraction is possible using the proposed architecture, the limitation of fixed channel stacking in DFTFB based channelizers does not exist in the proposed channelizer. In addition to the architectural reconfigurability explained above, each of the subfilters of the proposed channelizer has been implemented such that they can be reconfigured. This enables the same FB architecture to operate for a new communication standard. The filter level reconfigurability is explained in the next section.

6.3.2 Filter level Reconfigurability

By filter reconfigurability, it means changing the coefficients of each of the filter shown in the architecture of Fig. 6.1 according to the specifications of the new standard. It is well known that one of the efficient ways to reduce the complexity of multiplication operation is to realize it using shift and add operations. In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, the BCSs-based shift and add unit are employed in the proposed filter architectures. In Chapter 5, two new reconfigurable FIR filter architectures are proposed based on the BSE algorithm proposed in Chapter 4. The architecture in Chapter 5 consisted of a shift and add unit which will generate all the 3-bit BCSs using three adders. In this chapter, a modification of the CSM architecture in Chapter 5 is proposed for reducing its complexity. In the CSM architecture proposed in Chapter 5, the filter coefficients are stored in LUT without any coding. As a result of this, if the first few bits are zeros, the adders employed in the architecture are unnecessarily used. This problem is solved by incorporating a PS in the proposed reconfigurable FIR filter architecture. The proposed architecture of the filter for an 8-bit coefficient is shown in Fig. 6.5.

Figure 6.5 Proposed reconfigurable Filter architecture.

The components $M1$ and $M2$ are 8:1 multiplexers; $M3$ is a 4:1 multiplexer and $M4$ and $M5$ are 2:1 multiplexers. The input is given to the shift and add unit whose output is shared among the multiplexers. The architecture of shift and add unit is same as in Chapter 5. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from $[0\ 0\ 0]$ to $[1\ 1\ 1]$. Since the shifts to obtain the BCSs are known beforehand, the shifts can be hardwired. All these eight BCSs (including $[000]$) are then fed to the multiplexer units as shown in Fig. 6.5. Thus the use of 3-bit BCSs reduces the number of adders needed to implement the shift and add unit compared to conventional shift and add units. The filter coefficients are stored in the LUT in a coded format $SDDDDXXXXXXXX$, where S is the sign bit, $DDDD$ is the shift value of the most significant non-zero bit in the coefficient and X represent the bit values after the coefficients are shifted left so that the MSB to the right of the decimal point (position value corresponding to 2^{-1}) is always '1'. This can be illustrated with an 8-bit coefficient $h_k=0.00010011$. In h_k , the most significant non-zero bit is at position corresponding to 2^{-4} , thus $DDDD=0100$ and the new coded format for storing the coefficient is **0010010011000**. It must be noted that three zeros (coefficient part shown in *italics*) are inserted in the coefficient to avoid the use of adder A_2 when the output of $M3$ is 0. Each row in LUT corresponds to one coefficient. Note that only half the number of coefficients needs to be stored due to the coefficient symmetry of FIR filters. The MSB of the modified coefficient, S , stored in the LUT is given as the select signal to the Mux, $M5$. The Mux $M5$ determines whether the output needs to be complemented depending on the sign bit of the coefficient. The values $DDDD$ forms the select signal to the PS which will perform the shifting corresponding to the most significant non-zero bit in the original coefficient set (2^{-4} in the above example). The least significant 2 bits (corresponding to locations 2^{-7} , 2^{-8}) form the select signal to $M4$. If these 2 bits are '00' (as in the above example), then adder A_2 is not used, output of A_1 is selected as r_4 . Otherwise r_4 is the output of A_2 . The values r_1 , r_2 , r_3 and r_4 correspond to the outputs of multiplexers $M1$, $M2$, $M3$ and $M4$ respectively. The X values of coded-coefficients in the LUT are partitioned into 3-bit groups and given as the select signal to $M1$, $M2$ and $M3$ such that $(2^{-1}, 2^{-2}, 2^{-3})$ forms select signal to $M1$, $(2^{-4}, 2^{-5}, 2^{-6})$ to $M2$ and $(2^{-7}, 2^{-8})$ to $M3$ respectively. Reconfigurability can be achieved by changing the coefficients in the LUT. This reconfigurable architecture was employed for realizing all the sub-filters in the proposed FB. If an adder is not selected by the multiplexer when the corresponding bits in the coefficient are zero, the corresponding adder will not be

loaded and hence there will not be dynamic power consumption by the adder. Hence the proposed architecture will offer savings in dynamic power compared to the CSM architecture in Chapter 5.

6.3.3 Proposed Channelizer Architecture

In this section, the details of the proposed reconfigurable FB architecture are presented. The architecture of a mode-10 FB is shown in Fig. 6.6. This architecture in Fig. 6.6 receives channels with passband and stopband specifications as discussed in Section 6.3.1. It consists of a single modal filter, complementary delays (delays for obtaining the complementary output) and seven masking filters. As mentioned in Section 6.3.1, four delays, $M = 2, 3, 4$ and 6 are employed to obtain a $4 + C_4^2 = 10$ channels at the output. There are basically four channels with specifications $f_p = 0.1$ and $f_s = 0.15$, $f_p = 0.2$ and $f_s = 0.217$, $f_p = 0.3$ and $f_s = 0.325$ and $f_p = 0.4$ and $f_s = 0.433$. In order to extract all the ten channels simultaneously (the four basic channels and channels obtained by subtracting one basic channel from the other), employing the architecture in Fig. 6.1, eight masking filters are required. But for the case $M = 2$, $f_p = 0.1$ and $f_s = 0.15$ and this output can be obtained by masking the modal filter directly with a masking filter, and thus a complementary masking filter can be saved. As a result, only seven masking filters are required. The architecture of modal filter is developed to simultaneously generate all the basic channels with $M = 2, 3, 4$ and 6 .

Figure 6.6 Architecture of mode-10 proposed FB.

In Fig. 6.6, y_1 gives a channel with passband and stopband edges $f_p = 0.1$ and $f_s = 0.15$, y_2 gives a channel with passband and stopband edges $f_p = 0.2$ and $f_s = 0.217$, y_3 gives a channel with passband and stopband edges $f_p = 0.3$ and $f_s = 0.325$ and y_4 gives a channel with passband and stopband edges $f_p = 0.4$ and $f_s = 0.433$. These are the four basic channels. Additional $C_4^2 = 6$ channels can be obtained by subtracting one basic channel from the other one. For example, the output y_5 is obtained by subtracting the output of y_1 from y_2 . The channel y_5 is basically a band-pass channel with specification $[0.1, 0.15, 0.2, 0.217]$ which means stopbands from 0.1 to 0.15 and 0.2 to 0.217, and passband from 0.15 to 0.2. Note that the delays in obtaining y_1 and y_2 must be kept identical before the subtraction operation. This is based on the logic that if y_1 contains the output x_1+x_2 , and y_2 (which is a much wider passband filter output) contains $x_1+x_2+x_3$, then the output of y_5 will contain $x_1+x_2+x_3 - (x_1+x_2) = x_3$. Thus y_5 works as a band-pass filter allowing only x_3 to pass through. In Fig. 6.6, only y_5 is shown. The other channels y_6 to y_{10} can be obtained by subtracting appropriate channels.

The architecture of modal filter is shown in Fig. 6.7. The same modal filter is capable of working as modal filter for the four basic channels mentioned earlier.

Figure 6.7 Architecture of modal filter for mode-10 filter bank.

As discussed in Section 6.3.1, the passband and stopband edge specifications of modal filter are fixed as $f_{ap} = 0.2$ and $f_{as} = 0.3$ respectively. The peak passband ripple (PPR), δ_p , is taken as 0.1 dB and peak stopband ripple (PSR), δ_s , as -40 dB. The optimum filter-length, N , of the filter was calculated using the expression (6.6). From (6.6), by

substituting, $f_p = f_{ap} = 0.2$ and $f_s = f_{as} = 0.3$, $N = 21$. The notation N_{modal} represents the length of the modal filter in the rest of this chapter. In this example, N_{modal} is 21. Since TDF filter structure is employed and FIR filter coefficients are symmetric, only eleven coefficient multipliers are required for realizing the entire filter (Eleven symmetric coefficients of total twenty one coefficients). The redundant multiplications in these eleven multipliers are further reduced by employing the proposed BSE method as explained in Section 6.3.2. These eleven multipliers are common for all the four basic channels. The only difference is in the number of delays as shown in Fig. 6.7. The modal filter produces four outputs: y_{A1} , y_{A2} , y_{A3} and y_{A4} corresponding to $M = 2, 6, 4$ and 3 respectively. For obtaining y_{A1} , y_{A2} , y_{A3} and y_{A4} each delay of the conventional FIR filter is replaced by four distinct delays, 2, 6, 4 and 3 respectively. The frequency responses of the modal filter are similar to frequency responses shown in Fig. 6.2 (c).

The specifications of the masking filters, FMA and FMC , for obtaining y_{A1} to y_{A4} are shown in Table 6.4 discussed in Section 6.3.1. The PPR and PSR are fixed as 0.1 dB and -40 dB respectively for all the cases. The lengths of the masking filters for realizing y_{A1} , y_{A2} , y_{A3} and y_{A4} obtained using (6.6) are $N_{FMA1} = 8$, $N_{FMA2} = 26$, $N_{FMC2} = 26$, $N_{FMA3} = 16$, $N_{FMC3} = 16$, $N_{FMA4} = 12$ and $N_{FMC4} = 12$ respectively. The complementary response is obtained by using a series of $((N-1) \times M/2)$ delays. The number of delays required for obtaining the complementary outputs for y_{A2} , y_{A3} and y_{A4} are $(21-1) \times 6/2 = 60$ delays, $(21-1) \times 4/2 = 40$ delays and $(21-1) \times 3/2 = 30$ delays respectively. The output y_5 is obtained by subtracting the output of y_1 from y_2 , and y_5 is basically a band-pass channel with specification $[0.1, 0.15, 0.2, 0.217]$ which means stopbands from 0.1 to 0.15 and 0.2 to 0.217, and passband from 0.15 to 0.2. Note that the delays in obtaining y_1 and y_2 must be kept identical before the subtraction operation. This is to make the group delay identical to obtain y_5 accurately. For example, the total length for obtaining the output $f_p = 0.1$ and $f_s = 0.15$ is $21+8 = 29$ and that of $f_p = 0.2$ and $f_s = 0.217$ is $21+26 = 47$. In order to get the correct low-pass output as mentioned earlier, additional buffering of about $47-29 = 18$ delays are required to be added in the output of filter with specifications of $f_p = 0.1$ and $f_s = 0.15$. Thus by using the same FB architecture, it is possible to obtain four basic channels and six channels by subtracting one basic channel from the other appropriate basic channel.

It can be seen that when each delay is replaced by M delays, an $(M+1)$ -band filter is formed. In previous paragraphs, only different possibilities of low-pass channels have been considered. However it is possible to directly mask the $(M+1)$ bands using appropriate low-pass, high-pass and band-pass filters and consider as $(M+1)$ -independent channels. Thus for $M = 2$, $2+1 = 3$ channels are possible. Similarly, for $M = 3$, $M = 4$ and $M = 6$, four, five and seven channels are possible. Thus a total of 19 channels can be extracted using 19 separate masking filters. Since these masking filters are wide transition-band filters, they are of lower-order and hence less complex. A more detailed generalization of extraction of $(M+1)$ bands using $(M+1)$ masking filters can be done as follows: The $(M+1)$ channels obtained by replacing each delay of the modal filter by M consist of the following:

The output of modal filter which has the specifications as follows:

$$f_{se} = \begin{cases} 0 & \text{for } n = 0 \\ (n - f_{as}) / M & \text{for } n \neq 0 \end{cases} \quad (6.10)$$

$$f_{pb} = \begin{cases} 0 & \text{for } n = 0 \\ (n - f_{ap}) / M & \text{for } n \neq 0 \end{cases} \quad (6.11)$$

$$f_{pe} = \left\lceil \frac{n + f_{ap}}{M} \right\rceil \quad (6.12)$$

$$f_{sb} = \left\lceil \frac{n + f_{as}}{M} \right\rceil \quad (6.13)$$

The output of complementary filter which has the specifications as follows:

$$f_{se} = \left\lceil \frac{n + f_{ap}}{M} \right\rceil \quad (6.14)$$

$$f_{pb} = \left\lceil \frac{n + f_{as}}{M} \right\rceil \quad (6.15)$$

$$f_{pe} = \left\lceil \frac{(n+1) - f_{as}}{M} \right\rceil \quad (6.16)$$

$$f_{sb} = \left\lceil \frac{(n+1) - f_{ap}}{M} \right\rceil \quad (6.17)$$

where f_{se} and f_{pb} are the stopband and passband specifications of the rising edge and f_{pe} and f_{sb} are the passband and stopband specifications of the falling edge of a band-pass channel (frequency band) respectively and $n=0,1,2$ etc till the values of f_{se} , f_{pb} , f_{pe} and f_{sb} are less than 0.5 (Nyquist Criterion). If the values are more than 0.5, they are equated to 0.5 thus converting the channel as a low-pass channel. Similarly if the values of frequency edges are zeros, then the channel is a low-pass channel. The expressions (6.10)-(6.13) can be represented using Fig. 6.8 (a) and the expressions (6.14)-(6.17) using Fig. 6.8 (b). Fig. 6.8 (a) represents the frequency response of the modal filter, when each delay of the modal filter is replaced by M delays. Fig. 6.8 (b) represents the complementary response to response in Fig. 6.8 (a). Thus from Figures 6.8 (a) and 6.8 (b), it can be seen that the whole frequency spectrum is covered. Hence by using masking filters whose response as shown in Fig. 6.8 (c) and Fig. 6.8 (d), all the different frequency bands for a particular M can be isolated into $(M+1)$ channels as discussed earlier. Fig. 6.8 (c) represents the frequency responses of the masking filters for isolating the frequency bands in Fig. 6.8 (a) i. e., interpolated modal filter response. Similarly Fig. 6.8 (d) is employed to isolate the frequency bands in Fig. 6.8 (b).

Figure 6.8 Frequency edge specifications for expressions (6.10)-(6.17).

For obtaining these $(M+1)$ independent channels, the specifications of masking filters can be generalized depending on the above specifications (6.10-6.17) as follows:

1. The channels at the output of modal filter can be masked using filters of specifications:

$$f_{se} = \begin{cases} 0 & \text{for } n = 0 \\ (n-1 + f_{as}) / M & \text{for } n \neq 0 \end{cases} \quad (6.18)$$

$$f_{pb} = \begin{cases} 0 & \text{for } n = 0 \\ (n - f_{ap}) / M & \text{for } n \neq 0 \end{cases} \quad (6.19)$$

$$f_{pe} = \left\lceil \frac{n + f_{ap}}{M} \right\rceil \quad (6.20)$$

$$f_{se} = \left\lceil \frac{(n+1) - f_{as}}{M} \right\rceil \quad (6.21)$$

2. The channels at the output of complementary filter can be masked using filters of specifications:

$$f_{se} = \begin{cases} 0 & \text{for } n = 0 \\ (n - f_{ap}) / M & \text{for } n \neq 0 \end{cases} \quad (6.22)$$

$$f_{pb} = \left\lceil \frac{n + f_{as}}{M} \right\rceil \quad (6.23)$$

$$f_{pe} = \left\lceil \frac{(n+1) - f_{as}}{M} \right\rceil \quad (6.24)$$

$$f_{sb} = \left\lceil \frac{(n+1) + f_{ap}}{M} \right\rceil \quad (6.25)$$

The expressions (6.18)-(6.25) are derived on the basis that, the transition-band of the masking filters can be as wide as possible without overlapping with the passband of the adjacent channels. Fig. 6.8 (c) represents the expressions (6.18)-(6.21) and Fig. 6.8 (d) represents expressions (6.22)-(6.25). This can be clearly explained using the architecture shown in Fig. 6.9. The value of M is chosen as '4' for illustration. This means each delay of modal filter in Fig. 6.9 is replaced by $M = 4$ delays. As a result of this, the modal filter response consists of three frequency bands and complementary

response consists of two frequency bands and thus a total of $M+1 = 4+1 = 5$ frequency bands. To isolate each of these five frequency bands, five masking filters $FMA1$ to $FMA5$ are employed as shown in Fig. 6.9. The masking filters $FMA1$ to $FMA3$ are employed to isolate frequency bands from the modal filter output and $FMA4$ to $FMA5$ are employed to isolate frequency bands from the complementary output. Note that, the same FB architecture can support $M = 3, 2$ and 1 also. It should also be noted that the proposed architecture can be extended for larger values of M .

Figure 6.9 Architecture for extracting $(M+1)$ frequency bands.

6.4 Extraction of Fractional Bandwidths Using the Proposed Filter Bank Architecture

In the case of an SDR channelizer, the FB used should be capable of extracting a channel whose channel spacing is a fractional multiple of the spacing of another channel. For example, the bandwidth of D-AMPS standard is 30 kHz and that of PDC standard is 25 kHz. Thus the passband width of filter employed for extracting D-AMPS channel is 1.2 times greater than that employed for extracting PDC channel. The existing DFTFBs are not able to perform extraction of channels whose spacings are related by non-integer factors. The proposed FB architecture can be employed to tackle such a scenario. This is achieved by cascading the modal filters as shown in Fig. 6.10. The architecture of each modal filter is same as that in Fig. 6.7. In Fig. 6.10, stage-I consists of a cascaded arrangement of modal filter and a masking filter. The masking filter is employed to remove the images created in the frequency response other than the low-pass frequency band during interpolation by M . The masking filter employed in stage-I ensures that, at the output of each stage, only a low-pass channel will be obtained. Two parameters can be changed for each stage of the scheme in Fig.

6.10: (1) the interpolating factor, M and (2) the filter specifications at each stage. Based on this, the architecture in Fig. 6.10 can be operated in three modes as follows:

Mode-1: All the modal filters of cascaded arrangement having same specifications, but different values of delays M .

Mode-2: All the modal filters having the same value of delays M , but different filter specifications.

Mode-3: All the modal filters having different filter specifications and values of delays M .

Figure 6.10 Cascaded connection of modal filters.

6.4.1 Mode-1 Operation

Mode-1 operation is as follows: Assume $M_1, M_2, \dots, M_n$ are the delays of first, second, ..., n^{th} stage respectively and $M_1 < M_2 < \dots < M_n$. Let the passband and stopband specifications of the modal filter of all the stages are identical given by f_{ap} and f_{as} respectively. In this mode of operation, the output of first stage modal filter will be an (M_1+1) -band frequency response. But the masking filter employed in the stage-I masks all the frequency bands other than the low-pass channel. Thus the output of stage-I consists of a low-pass channel of passband width f_{ap}/M_1 and passband and stopband specifications of f_{ap}/M_1 and f_{as}/M_1 respectively. At the second stage, the passband and stopband edges as well as passband width of the modal filter are f_{ap}/M_2 , f_{as}/M_2 and f_{ap}/M_2 respectively. This means the ratio of change in passband width between stage-I output and stage-II output is $f_{ap}/M_1 / f_{ap}/M_2 = M_2/M_1$. Thus it can be concluded that the output of stage-I has a passband width of f_{ap}/M_1 and the output of stage-II has a passband width of f_{ap}/M_2 which means the passband width at the output of stage-II is reduced by a factor of M_2/M_1 . If M_1 and M_2 are chosen such that M_2 is not an integer multiple of M_1 , then a fractional reduction in passband width can be obtained as M_2/M_1 is a fractional value. Similarly, at the n^{th} stage, the passband width will be reduced by a factor of M_n/M_{n-1} compared to the $(n-1)^{\text{th}}$ stage. In general, the bandwidth of the signal at the output of a specific stage will be different from its

previous stage by the ratio of the delay value of the chosen stage to that of its previous stage. Also since the output of stage-I is a low-pass frequency band, the outputs of all the other stages will also be low-pass bands.

Figure 6.11 (a) Frequency response of modal filter.

Figure 6.11 (b) Frequency response of stage-I.

Figure 6.11 (c) Frequency response of stage-II.

Figure 6.11 (d) Frequency response of stage-III.

The mode-1 case can be illustrated using a three-stage cascaded connection with $M_1 = 2$, $M_2 = 6$ and $M_3 = 9$. Assume that the frequency response specifications of the modal filters of all the three stages are identical, i.e., $f_{ap} = 0.2$ and $f_{as} = 0.25$ and passband and stopband ripples be 0.1 dB and -100 dB respectively. The length of each modal filter obtained from (6.6) is 79. The masking filter associated with stage-I can have passband and stopband specifications of 0.1 and 0.375 from expressions (6.12-6.13). The length of the masking filter obtained from (6.6) is 14. The frequency responses for each stage are shown in Fig. 6.11. Fig. 6.11 (a) shows the original modal filter response. The over-all response of stage-I (the over-all frequency response of modal filter and masking filter) is shown in Fig. 6.11 (b). The passband width at the output of stage-I is $1/M_1 = 1/2$ of the modal filter response of Fig. 6.11 (a). Fig. 6.11 (c)

represents the over-all response of stages I and II. The passband width is reduced by $M_2/M_1 = 6/2 = 3$ compared to the response of stage-I in Fig. 6.11 (b). The over-all frequency response of stages I, II and III is shown in Fig. 6.11 (d). It can be seen from Fig. 6.11 (d) that, the passband width is further reduced by $M_3/M_2 = 9/6 = 1.5$ compared to the over-all frequency response of stages I and II. Thus, from the illustrative example in Fig. 6.11, it is evident that, the passband width (and hence the passband and stopband edges) of any stage can be reduced by fractional values of the passband width of previous stage using appropriate values of the delays (M) for the two stages. To the best of author's knowledge, none of the existing filterbanks possess this capability.

6.4.2 Mode-2 Operation

Mode-2 operation is as follows: The delays of all the stages are fixed as M and the specifications of modal filters of each stage is different. A masking filter is cascaded with the modal filter of stage-I (similar to mode-1 operation) to remove all the images other than low-pass frequency band due to interpolation. Let the passband and stopband edge specifications of the modal filter in the i^{th} stage be f_{api} and f_{asi} respectively. Then the output of stage-I will be a low-pass frequency band with passband and stopband edge specifications of f_{ap1}/M and f_{as1}/M and passband width of f_{ap1}/M . The output of stage-II will be a low-pass frequency band with passband and stopband edge specifications of f_{ap2}/M and f_{as2}/M and passband width of f_{ap2}/M . Similarly the output of stage- n will be a low-pass frequency band with passband and stopband edge specifications of f_{apn}/M and f_{asn}/M and passband width of f_{apn}/M . Thus it can be seen that, the passband width of stage-II with respect to stage-I is f_{ap2}/f_{ap1} . This can be illustrated using an example as follows. Consider a three stage arrangement with $M=4$. Let the specifications of modal filters of stages I, II and III be $f_{ap1} = 0.3$ and $f_{as1} = 0.4$, $f_{ap2} = 0.2$ and $f_{as2} = 0.3$ and $f_{ap3} = 0.1$ and $f_{as3} = 0.15$ respectively. The peak passband and stopband ripples are taken 0.1 dB and -148 dB respectively. The passband and stopband specifications of the masking filter associated with stage-I are 0.075 and 0.15 respectively from (6.12-6.13). The frequency responses of the three stages are shown in Fig. 6.12. It can be noted from Fig. 6.12 that, the passband width is proportional to the modal filter specifications of the three stages. At the output of stage-I, the passband width is determined by the value of M , which is 4 in this case.

The passband width of the low-pass channel is reduced by a factor of M as shown in Fig. 6.12 (a). The output of stage-II is determined by the relation between specifications of modal filters of stage-I and stage-II and the value of M .

Figure 6.12 (a) Frequency response of stage-I.

Figure 6.12 (b) Frequency response of stage-II.

Figure 6.12 (c) Frequency response of stage-III.

In this case, since the specifications (passband and stopband edges) of modal filter of stage-III is half of that of stage-II, the passband width of output at stage-III is reduced by half compared to that of stage-II as shown in Fig. 6.12 (c). The response of Fig. 6.12 (c) is seen deteriorated and this can be rectified by appropriate designing of filter at stage-III. All the filters chosen in this example are for illustration purpose only and the primary focus is on extraction of channels with bandwidths related by fractional factors.

6.4.3 Mode-3 Operation

The mode-3 operation is a hybrid of modes 1 and 2. In mode-3 operation, the specifications of the modal filters and the number of delays at each stage are not identical. Let the delays of first, second and that of the n^{th} stage be M_1 , M_2 and M_n

respectively. A masking filter is cascaded with the modal filter of stage-I (similar to modes 1 and 2 operation) to remove all the images other than low-pass frequency band. Let the passband and stopband edge specifications of the modal filter in the i^{th} stage be f_{api} and f_{asi} . Thus the output of stage-I will be a low-pass frequency band with passband and stopband edge specifications of f_{ap1}/M_1 and f_{as1}/M_1 and passband width of f_{ap1}/M_1 . The output of stage-II will be a low-pass frequency band with passband and stopband edge specifications of f_{ap2}/M_2 and f_{as2}/M_2 and passband width of f_{ap2}/M_2 . Similarly the output of stage- n will be a low-pass frequency band with passband and stopband edge specifications of f_{apn}/M_n and f_{asn}/M_n and passband width of f_{apn}/M_n . Thus it can be seen that, the passband width of stage-II with respect to stage-I is $(f_{ap2}M_1)/(f_{ap1}M_2)$. Thus by selecting different values of passband and stopband specifications and values of M , it is possible to obtain different frequency responses using the same architecture of Fig. 6.10.

Thus it can be seen that, the three cases mentioned above give good flexibility for the proposed FB. Also it is possible to extract channels, whose spacings are related by fractional factors, at very low complexity using the proposed FB. In order to obtain the overall filter response shown in Fig. 6.11 (c), conventional FIR filter design algorithms such as Parks-McClellan algorithm would require a filter order as high as 1500 using (6.6). But using the proposed FB, the total order of filters of the three stages will come only to 251 ($= 79 \times 3 + 14$) which is a complexity reduction of 83.3%.

6.5 Experimental Results

In this section, the experimental results of the proposed filter bank are presented. This section consists of a qualitative comparison of the proposed filter bank followed by a quantitative comparison.

6.5.1 Qualitative Comparison

The proposed FB is compared with the DFTFB, GFB [9] and PC approach for eight parameters. The parameters used for comparison are area efficiency, flexibility which includes filter shape, channel tuning and possibility of independent channels, ability to extract channels of non-uniform bandwidth, ability to extract very narrow passband channels and capability for multi-mode operation. *Area efficiency* is denoted to

represent the complexity in terms of area for each of the FB architectures. For a single-mode channelizer, the DFTFB and the proposed FB offers best area efficiency. However, in SDR where multi-mode operation is mandatory, distinct channelizers are required, if DFTFB is employed. As the proposed FB can be easily reconfigured, it offers much better area efficiency than the DFTFB. The complexity of GFB will increase with the number of received standards. The PC approach consumes lot of area as separate filters are required for each channel. The proposed FB architecture offers the best area efficiency as it is based on the inherently less complex FRM approach [75]. Also the complexity of each subfilter in the FB architecture is further reduced by employing the BSE algorithm proposed in Chapter 4. The *flexibility* of the channelizer is its ability to adapt to a change in any parameter in response to a new communication mode. The flexibility is excellent for proposed FB and the PC approach when compared to DFTFB and GFB. The flexibility is least for DFTFB and GFB because they use only a single low-pass filter which is modulated and used for other channel reception using polyphase components. The proposed FB architecture is highly flexible because each filter multiplier follows the architecture of Fig. 6.5 and hence can easily be reconfigured for any parameter change. By *filter shape* it means the specifications of the filter such as passband and stopband frequencies, PSR and PPR can be adjusted. The filter shape is selectable only to a limited extent in DFTFB and GFB. The channels are dependent for DFTFB and GFB and hence *channel tuning* is fixed for these approaches. Since PC approach works independently (as there are separate DDC blocks for each channel), *independent channels* are possible in this approach and hence channel tuning is also feasible. The proposed FB offers reconfigurability at two levels (architectural and filter levels). Therefore independent channels can be easily extracted and channel tuning can be done very efficiently. *Extraction of non-uniform bandwidth channels* is impossible in DFTFB and GFB approaches. The PC approach can extract non-uniform bandwidth channels as it has separate filters in each DDC branch. The non-uniform bandwidth extraction is easily possible in the proposed FB architecture as shown in Section 6.4. Similarly *extraction of very narrow passband channels* is not possible other than PC approach and the proposed FB architecture. In the case of PC approach, the complexity is extremely high as very narrow transition filters are required which increases the filter order. But FRM technique is specifically designed for very narrow transition filters and hence the complexity of the proposed FB architecture is very less compared to all other FBs as well as the PC approach. None of the existing

FBs offer *multimode operation* needed in SDR using the same architecture, whereas the proposed FB offers multi-mode operation due to its architectural and filter level reconfigurability.

6.5.2 Quantitative Comparison

A quantitative comparison of the proposed channelizer with conventional channelizers is presented in this section. Table 6.5 shows the comparison of the multiplication rate of the proposed channelizer with that of the PC approach, DFTFB, GFB [9] and MPRB [18]. Multiplication rate of a channelizer is defined as the total number of multiplications per sampling period for extracting N_I number of channels simultaneously. The multiplications involved in a channelizer can be grouped into three as follows: 1) Multiplications involved with channel filtering, 2) Multiplications involved with digital down conversion and 3) Multiplications involved with modulation of filters (this is not applicable for PC approaches).

Table 6.5
Multiplication rate of channelizers

	PC Approach	DFTFB	GFB [9]	MPRB [18]	Proposed FRM Channelizer
Filter	$\frac{2N_I}{D} L \cdot f_s$	$L \cdot f_s$	$L \cdot f_s$	$2L \cdot f_s$	$l \cdot f_s$
DDC	$2N_I \cdot f_s$	-	-	-	$2(C_M^2) \cdot f_s$
Modulation of Filters	-	$\frac{\log_2 D}{2} f_s$	$N_I \cdot f_s$	$\log_2 D \cdot f_s$	-
Sum	$2N_I(\frac{L}{D} + 1)f_s$	$(L + \frac{\log_2 D}{2})f_s$	$(L + N_I)f_s$	$(2L + \log_2 D)f_s$	$(l + 2(C_M^2))f_s$

In Table 6.5, L represents the number of non-zero filter coefficients of the prototype filter for the PC approach, DFTFB, GFB and MPRB (only non-zero coefficients have been considered as they will only result in multiplication complexity), f_s represents the sampling frequency and D represents the down-sampling rate. For the proposed channelizer, l represents the sum of the non-zero coefficients for modal and masking filters and M represents the number of delays involved with the modal filter. The multiplication rates for PC approach, DFTFB and GFB are taken directly from [9]. The MPRB [18] consists of two prototype filters (one at analysis section and one at synthesis section) and one DFT and IDFT unit each. Thus the complexity will be

double that of the DFTFB. In the case of the proposed channelizer, the multiplication complexity of the filter is given by $l.f_s$. The modal filter with M different delays can form $M + C_M^2$ mode channelizer. Out of these $M + C_M^2$ channels, M channels are basic channels located in baseband and hence there is no requirement of any digital down conversion (This is because M channels are obtained as low-pass filter outputs). The remaining C_M^2 channels are obtained with the help of low-pass filters as discussed in Section 6.3.2 and hence require digital down conversion to baseband. As a result of this, the multiplication rate for the digital down conversion is $2.C_M^2.f_s$. Thus the total multiplication rate of the proposed FB channelizer is $(l + 2.C_M^2).f_s$. Note that the proposed channelizer complexity is independent of the number of extracted channels, N_l . Also ' l ' for the proposed channelizer is fixed for a particular set of modes. For example, if $M=4$, it is possible to have $M + C_M^2 = 10$ modes of operation. Hence for extracting 1 to 10 channels, the same architecture can be used. Similarly if $M=5$, $M + C_M^2 = 15$ modes of operation is possible. Hence ' l ' is not directly dependent on the number of extracted channels and so as the multiplication rate of the proposed channelizer. Note that l is very small compared to the L . For the 10-mode channelizer designed in Section 6.3.1, $l=137$ and $L= 228$ for obtaining a dual mode with specification $f_p=0.2$ and $f_s=0.217$ and $f_p= 0.3$ and $f_s= 0.325$ according to the expression in (6.6). The value for L will be definitely much larger than l especially when more than two modes of operation are required.

Table 6.6 shows the multiplication rate for a single channel channelizer which is a measure of per-channel complexity. Multiplication rate for a single channel channelizer is the number of multiplications per sampling period for extracting a single channel. The comparison in terms of f_s is obtained from Table 6.5. In Table 6.6, the down-sampling rate is fixed as $D = 4$ for the sake of easier comparison. The stopband and passband edges of the filters are shown in the Table 6.6. From Table 6.6, it is evident that, for a single channel reception, the PC approach offers the least multiplication rate and complexity. The DFTFB, GFB and MPRB are not appropriate for single channel reception as they have high multiplication rates. From Table 6.6 the multiplication rate for PC approach is $2.(L/4+1).f_s$, that of DFTFB and GFB are

$(L+1)f_s$, that of MPRB is $2.(L+1)f_s$ and that of the proposed channelizer is lf_s . Thus multiplication rate can be considered directly proportional to the L or l . The PC approach has an advantage due to the decimation $L/4$. The multiplication rates of DFTFB, GFB and MPRB are always on the higher side as L is very high compared to l . The value of l is low because of the use of FRM technique, which is specifically designed for the sharp transition-band FIR filter design with low complexity. On an average for a single channel channelizer, the proposed channelizer offers multiplication rate reduction of 32.7% over DFTFB and GFB. Though the proposed FRM based FB has more multiplication rate for single channel extraction than PC approach in most cases, the former outperforms the latter when multiple channel extraction and multi-mode operation are needed as in the case of an SDR.

Table 6.6
Multiplication rate of a single channel channelizer

Passband & Stopband Edges ($D=4$, $N_f=1$)	PC Approach	DFTFB	GFB [9]	MPRB [18]	Proposed FRM Channelizer
	$2(\frac{L}{4}+1)f_s$	$(L+1)f_s$	$(L+1)f_s$	$2(L+1)f_s$	$(l+2(MC2))f_s$
0.1, 0.15	$22f_s$	$40f_s$	$40f_s$	$80f_s$	$29f_s$
0.1, 0.15, 0.2, 0.217	$61f_s$	$118f_s$	$118f_s$	$236f_s$	$81f_s$
0.1, 0.15, 0.3, 0.325	$42f_s$	$80f_s$	$80f_s$	$160f_s$	$61f_s$
0.1, 0.15, 0.4, 0.433	$32f_s$	$61f_s$	$61f_s$	$122f_s$	$53f_s$
0.2, 0.217	$61f_s$	$118f_s$	$118f_s$	$236f_s$	$73f_s$
0.2, 0.217, 0.3, 0.325	$61f_s$	$118f_s$	$118f_s$	$236f_s$	$105f_s$
0.2, 0.217, 0.4, 0.433	$61f_s$	$118f_s$	$118f_s$	$236f_s$	$97f_s$
0.3, 0.325	$42f_s$	$80f_s$	$80f_s$	$160f_s$	$53f_s$
0.3, 0.325, 0.4, 0.433	$42f_s$	$80f_s$	$80f_s$	$160f_s$	$77f_s$
0.4, 0.433	$32f_s$	$61f_s$	$61f_s$	$122f_s$	$45f_s$

Table 6.7 shows the comparison of multiplication rates of different channelizers for multiple channel extraction. Table 6.7 is obtained from Table 6.5 by substituting appropriate values. In this case also down-sampling rate, D , is fixed as 4 and the stopband and passband ripples are chosen as -40 dB and 0.1 dB respectively. The frequency response specifications of the channels are given below Table 6.7. For example, considering number of channels N_f is 3, the passband and stopband specifications of these three channels are $f_p = 0.1$ and $f_s = 0.15$, $f_p = 0.2$ and $f_s = 0.217$, and $f_p = 0.3$ and $f_s = 0.325$. For such a mode-3 channelizer, the PC, DFTFB, GFB and MPRB require distinct prototype filters with lengths, L , of 39 (for $f_p = 0.1$ and $f_s = 0.15$), 117 (for $f_p = 0.2$ and $f_s = 0.217$) and 79 (for $f_p = 0.3$ and $f_s = 0.325$) from (6.6). Thus considering the mode-3 case, from Table 6.7, the multiplication rates for PC

approach, DFTFB, GFB and MPRB are $2N_I[(L/4)+1]f_s = 2(6)[((39+117+79)/4)+1]f_s = 359f_s$, $(L+1)f_s = (39+117+79+1)f_s = 236f_s$, $(L+N_I)f_s = (39+117+79+3)f_s = 238f_s$ and $2(L+1)f_s = 2(39+117+79+1)f_s = 472f_s$. For the proposed FB, the lengths of modal filter, *FMA1*, *FMA2*, *FMC2*, *FMA3* and *FMC3* are 21, 8, 26, 26, 16 and 16 respectively. Hence the multiplication rate is only $(21+8+26+26+16+16)f_s = 113f_s$. Thus it is evident that the proposed FB offers good reduction in multiplication rate. As expected, it can be seen from Table 6.7 that the complexity of PC approach linearly increases with the number of channels.

Table 6.7
Multiplication rate of a multiple channel channelizer

No. of Channels (D=4)	PC Approach	DFTFB	GFB [9]	MPRB [18]	Proposed FRM Channelizer
	$2N_I(\frac{L}{4}+1)f_s$	$(L+1)f_s$	$(L+N_I)f_s$	$2(L+1)f_s$	$(I+2(MC2))f_s$
1	22 f_s	40 f_s	40 f_s	80 f_s	29 f_s
2	160 f_s	157 f_s	158 f_s	314 f_s	81 f_s
3	359 f_s	236 f_s	238 f_s	472 f_s	113 f_s
4	598 f_s	296 f_s	299 f_s	592 f_s	137 f_s
5	1040 f_s	413 f_s	417 f_s	826 f_s	137 f_s
6	1485 f_s	492 f_s	497 f_s	984 f_s	137 f_s
7	1943 f_s	552 f_s	558 f_s	1104 f_s	137 f_s
8	2688 f_s	669 f_s	676 f_s	1338 f_s	137 f_s
9	3024 f_s	786 f_s	794 f_s	1572 f_s	137 f_s
10	4340 f_s	865 f_s	874 f_s	1730 f_s	137 f_s

1={0.1, 0.15}, 2={0.1, 0.15}, [0.2,0.217]}, 3={0.1, 0.15}, [0.2,0.217], [0.3,0.325]},
4={0.1, 0.15}, [0.2,0.217], [0.3,0.325], [0.4,0.433]},
5={0.1, 0.15}, [0.2,0.217], [0.3,0.325], [0.4,0.433], [0.1, 0.15, 0.2,0.217]}
6={0.1, 0.15}, [0.2,0.217], [0.3,0.325], [0.4,0.433], [0.1, 0.15, .2,0.217], [0.1, 0.15, 0.3,0.325]}
7={0.1, 0.15}, [0.2,0.217], [0.3,0.325], [0.4,0.433], [0.1, 0.15, 0.2,0.217], [0.1,0.15,0.3,0.325],
[0.1, 0.15, 0.4,0.433]}
8={0.1, 0.15},[0.2,0.217],[0.3,0.325],[0.4,0.433], [0.1, 0.15, 0.2,0.217],[0.1, 0.15, 0.3,0.325],
[0.1, 0.15, 0.4,0.433],[0.2,0.217, 0.3,0.325]}
9={0.1, 0.15},[0.2,0.217],[0.3,0.325],[0.4,0.433], [0.1, 0.15, 0.2,0.217],[0.1, 0.15, 0.3,0.325],[0.1,
0.15, 0.4,0.433],[0.2,0.217, 0.3,0.325], [0.2,0.217, 0.4,0.433]}
10={0.1, 0.15},[0.2,0.217],[0.3,0.325],[0.4,0.433], [0.1, 0.15, 0.2,0.217],[0.1, 0.15, 0.3,0.325],
[0.1, 0.15, 0.4,0.433],[0.2,0.217, 0.3,0.325], [0.2,0.217, 0.4,0.433],[0.3,0.325, 0.4,0.433]}

The DFTFB and GFB have almost same multiplication rates, MPRB has twice the multiplication rate of DFTFB and their complexity is substantially higher than the proposed FB. The complexity of the proposed FB increases till mode-4 but still substantially lower than other channelizers. For any number of modes larger than 4 (up to 10 modes), all the channels can be generated with the same set of modal and masking filters and hence the complexity of the proposed FB remains constant. On an

average, considering more than two communication standards, the proposed channelizer offers multiplication rate reduction of 83.5% over PC approach, 70.8% over DFTFB, 72% over GFB [9] and 81% over MPRB [18].

6.5.3 Design Example

In this section, the design results of the proposed channelizer are presented for a dual-mode CDMA/WCDMA SDR receiver. The sampling rate chosen is 50 MHz. The channel filters can extract 1250 kHz CDMA channels and 5 MHz WCDMA channels from the input signal. The CDMA channel is assumed to be located at 0 - 1250 kHz and the WCDMA channel at 7.5 - 12.5 MHz. The peak passband ripple is chosen as 0.1 dB. The filter stop-band specification is chosen as -40 dB for both the standards for illustration. Reconfigurability can be achieved as follows. Consider a modal filter with frequency specifications as shown in Fig. 6.13 (a). The passband and stopband edges are fixed at 6250 kHz and 8750 kHz respectively. The implementation of such a filter is less complex as the transition is very wide. The architecture of the proposed FB for obtaining CDMA and W-CDMA channels is shown in Fig. 6.6. For the filter response in Fig. 6.13 (a), the filter-length N obtained using (6.6) is 42. Employing the architecture of Fig. 6.7, and replacing each delay of the modal filter with $M_1=10$, a multi-band response as shown in Fig. 6.13 (b) is obtained. By using a very wide transition-band masking filter $FMA1$ whose response is shown in Fig. 6.13 (b), the CDMA channel (passband of 1250 kHz for IS-95) can be extracted. If each delay of the modal filter is replaced by $M_2=12$, another multi-band response as shown in Fig. 6.13 (c) is obtained. The complementary response of case $M_2=12$ is shown in Fig. 6.13 (d). The frequency responses of masking filters for obtaining the W-CDMA channel (passband of 5 MHz) are also shown in Fig. 6.13 (c) and Fig. 6.13 (d). Using the architecture shown in Fig. 6.6, the overall W-CDMA filter response is shown in Fig. 6.13 (e). As shown in the architecture of Fig. 6.6, the outputs of $FMA2$ (whose frequency response is shown in Fig. 6.13 (c)) and that of $FMC2$ (whose frequency response is shown in Fig. 6.13(d)) are added to obtain the W-CDMA filter response shown in Fig. 6.13 (e). It must be noted that the passband and stopband attenuations are not altered in the FRM technique as can be seen from the filter frequency responses shown in Figures 6.13 (a) to 6.13 (e). This robustness of magnitude response is an inherent property of FRM technique [75].

Figure 6.13 (a) Frequency response of modal filter.

Figure 6.13 (b) Frequency response of case M=10.

Figure 6.13 (c) Frequency response of modal filter and FMA for case M=12.

Figure 6.13 (d) Frequency response of complementary delay output and FMC for case M=12.

Figure 6.13 (e) Overall frequency response of case M = 12 obtained by adding FMA2 (Fig. 6.13 (c)) and FMC2 (Fig. 6.13 (d)) as shown in the architecture of Figure 6.6.

The PPR and PSR are maintained as 0.1 dB and -40 dB respectively as shown in Figures 6.13 (a) to 6.13 (e). Using (6.6), the total number of taps required for the dual-mode CDMA/W-CDMA operation is 168 (42 taps for modal filter + 32 taps for CDMA masking filter *FMA1* (shown in Fig. 6.13 (b)) + 32 taps for W-CDMA masking filter *FMA2* (shown in Fig. 6.13 (c)) + 62 taps for W-CDMA complementary filter *FMC2* (shown in Fig. 6.13 (d))). It can be seen that for obtaining the CDMA filter response using DFTFB, a filter with 1000 taps is required using (6.6), which is approximately six times more than the proposed filter bank. Also, a separate DFTFB would be required for W-CDMA, which would further increase the cost.

6.6 Synthesis Results

The synthesis of the proposed channelizer has been done on 0.18μm CMOS technology. The proposed FB, PC approach and DFTFB are implemented based on the filter architecture proposed in Section 6.3.2 (Fig. 6.5). Since all the filters used in these methods and the proposed channelizers are based on the same filter architecture in Fig. 6.5, the comparison is fair. The mode-10 channelizer is synthesized and the results are shown in Table 6.8. The parameter ‘delay’ represents the latency of the architecture.. The filter coefficient wordlength is 16 bits for all the cases.

Table 6.8
Synopsys Synthesis Results

	PC Approach	DFTFB	Proposed FB
Area (mm ²)	50.738	23.2077	7.591
Delay (ns)	43	32.4	18.65
Power (mW)	106.8	78.3	58.36

For PC approach, distinct filters are required and hence the total length of the filters for the specifications given below Table 6.7 is 1147 according to (6.6). For DFTFB, the total length of the filters obtained using (6.6) is 902 as the low-pass channels from 5 to 10 can be obtained by using the DFT from the same prototype filters for channels 1 to 4. The total length of filters in proposed FB architecture is 137 (obtained by summing the length of all the subfilters of the proposed architecture). From the filter lengths itself, it is evident that the proposed FB architecture offers the minimum area, power and delay in operation followed by DFTFB and PC approach. From Table 6.8, it can be noted that the proposed FB architecture offers area reduction of 85% over PC approach and 67.3% over DFTFB, power reduction of 48.5% over PC approach and 25.5% over

DFTFB and improvement in speed of 56.7% over PC approach and 42.4% over DFTFB. It must be noted that the channel filters of the proposed FB architecture as well as the PC and DFTFB architectures employ the same architecture in Fig. 6.5, which is based on the low complexity BSE method proposed in Chapter 4. On the other hand, if DFTFB and PC architectures were based on conventional filter implementations, the results would have been much inferior.

6.7 Implementation Results

The proposed FB and DFTFB have been implemented and tested on Xilinx Virtex 2v3000ffl152-4 FPGA associated with the dual DSP-FPGA Signalmaster kit provided by Lyrtech [74]. A model based design using Matlab’s Simulink and Xilinx’s System generator was employed for the implementation purpose. A multi-tone input signal was generated by summing up sine waves of different frequencies ranging from 100 Hz to 20 MHz each sampled at 50 MHz as discussed in the design example. The purpose of using a multitone signal is to test the functionality of the channelizer by reconfiguring the filters to select the desired frequency. By dynamically changing the input frequencies using the function in Simulink, it was verified that the proposed FB architectures work well for frequencies of several tens of MHz. The implementation results are shown in Table 6.9. The area and delay results are obtained using log viewer block. The power dissipation is obtained using Xilinx XPower. Table 6.9 shows that the proposed FB offers area reduction of 37%, speed improvement of 42% and power reduction of 25% over the DFTFB (over PC approach also). The sampling frequency in Table 6.9 implies the maximum frequency at which the architectures can work.

Table 6.9
Implementation results for Proposed FB architecture

	Proposed FB	DFTFB
LUTs	15100	23456
Slices	10107	12345
Flip-Flops	15513	24567
Data Arrival Time (ns)	38.672	77
Sampling Frequency (MHz)	24	12
Power Dissipation (mW)	558.77	2235

6.8 Summary

In this chapter, new reconfigurable low complexity multi-mode filters and filter banks based on the frequency response masking (FRM) technique have been presented. The

proposed architectures are inherently less complex and offer two levels of reconfigurability: 1) at architectural level and 2) at filter level. The proposed architectures can be easily reconfigured for multi-mode operation. The FRM technique is modified to improve the speed and reduce the complexity. The proposed filter bank (FB) architecture can extract non-uniform bandwidth and very narrow bandwidth channels compared to conventional channelizers. It was also shown the extraction of channels, whose bandwidths are related by fractional factors, using the proposed FB architecture. Design examples show that the proposed FB offers complexity reduction of 83.5% over per-channel (PC) approach, 70.8% over discrete Fourier transform filter bank (DFTFB), 72% over Goertzel filter bank (GFB) and 81% over modulated perfect reconstruction filter banks. The proposed channelizer has been implemented on 0.18 μ m CMOS technology and compared with PC and DFTFB approaches. Synthesis results show that the proposed channelizer offers area reduction of 85% over PC approach and 67.3% over DFTFB, power reduction of 48.5% over PC approach and 25.5% over DFTFB and improvement in speed of 56.7% over PC approach and 42.4% over DFTFB. Even though the filters and filter banks based on the FRM technique are substantially less complex and easily reconfigurable, the design of these reconfigurable filters and filter banks lack absolute control over the passband width and the location of these passbands, which will impose constraints on the degree of frequency response flexibility. The design of FRM-based filter bank is tedious when multiple channels of uniform bandwidth, i. e., multiple channels of same communication standard, needs to be extracted. In such a case, DFTFB is still preferred, even though the implementation is expensive. In the next chapter, a new method, known as coefficient decimation approach is proposed, to design filters and filter banks, which have absolute control over the passband width and center frequencies of the passbands. A reconfigurable low complexity substitute for DFTFB is also presented in the next chapter.

Chapter 7

Coefficient Decimation based Filter Banks with Absolute Control over Passband Width and Centre Frequencies

In this chapter, a new approach to implement computationally efficient reconfigurable filters and filter banks for SDR receivers is presented. The proposed filters and filter banks based on a coefficient decimation approach have absolute control over the passband width and passband locations i. e., center frequencies of passbands, when compared to the frequency response masking-based filters and filter banks presented in Chapter 6. The coefficient decimation (CD) approach is as follows: If the coefficients of a finite impulse response filter are decimated by M , i.e., if every M -th coefficient of the filter is kept unchanged and remaining coefficients are replaced by zeros, a multi-band frequency response will be obtained. The frequency response of the decimated filter will have passbands with centre frequencies at $2\pi k/M$, where k is an integer ranging from 0 to $M-1$. If these multi-band frequency responses are subtracted from each other or selectively masked using inherently less complex wide transition-band masking filters, different low-pass, high-pass, band-pass, and band-stop frequency bands can be obtained. The resulting filter bank (FB), whose bands' centre frequencies are located at integer multiples of $2\pi/M$, is a low complexity alternative to the well known uniform discrete Fourier transform (DFT) filter banks. It can be shown that the proposed FB is more flexible and easily reconfigurable than the DFTFB. Furthermore, the proposed FB is able to receive channels of multiple standards simultaneously, whereas separate filter banks would be required for simultaneous reception of multi-standard channels in a DFTFB based receiver. It is also shown that the channelizer based on the proposed FB does not require any DFT for its implementation. This is significant in low complexity implementation of SDR channelizers.

7.1 Proposed Coefficient Decimation Approach

In this section, the coefficient decimation (CD) approach is presented. If the coefficients of an FIR filter are decimated by M , i.e., every M -th coefficient is kept unchanged and all others are replaced by zero values, a frequency response similar to images created during upsampling is obtained. The definition of *coefficient decimation* in this context is that unused coefficients (i.e., coefficients other than every M -th coefficient) are replaced by zero values as opposed to the conventional notion of discarding unused samples in the decimation of a signal.

7.1.1 Theoretical Background

Let $h(n)$ be the original set of coefficients. If all the coefficients other than every M^{th} coefficient are replaced by zeros,

$$h'(n) = h(n)c_M(n) \quad (7.1)$$

$$\text{where, } c_M(n) = \begin{cases} = 1 & n = mM; m = 0, 1, 2, \text{etc} \\ = 0 & \text{otherwise} \end{cases} \quad (7.2)$$

The function $c_M(n)$ is periodic with period M , and hence the Fourier series expansion is given by

$$c_M(n) = \frac{1}{M} \sum_{k=0}^{M-1} C(k) e^{\frac{j2\pi kn}{M}} \quad (7.3)$$

where $C(k)$ are complex-valued Fourier series coefficients defined by

$$C(k) = \sum_{n=0}^{M-1} c_M(n) e^{\frac{-j2\pi kn}{M}} \quad (7.4)$$

Substituting (7.2) into (7.4) it follows that $C(k) = 1$ for all k . Hence,

$$c_M(n) = \frac{1}{M} \sum_{k=0}^{M-1} e^{\frac{j2\pi kn}{M}} \quad (7.5)$$

Now the Fourier transform of the modified coefficients, $h'(n)$,

$$H'(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h'(n) e^{-j\omega n} = \sum_{n=-\infty}^{\infty} h(n) c_M(n) e^{-j\omega n}$$

$$= \sum_{n=-\infty}^{\infty} h(n) \left(\frac{1}{M} \sum_{k=0}^{M-1} e^{j \frac{2\pi k n}{M}} \right) e^{-j \omega n} \quad (7.6)$$

Finally by interchanging the sums in (7.6)

$$\begin{aligned} H'(e^{j\omega}) &= \frac{1}{M} \sum_{k=0}^{M-1} \sum_{n=-\infty}^{\infty} h(n) e^{-j n (\omega - \frac{2\pi k}{M})} \\ &= \frac{1}{M} \sum_{k=0}^{M-1} H(e^{j(\omega - \frac{2\pi k}{M})}) \end{aligned} \quad (7.7)$$

It can be noted from (7.7) that, the frequency response is scaled by M and the replicas of the frequency spectrum are introduced at integer multiples of $2\pi/M$. Thus in order to recover the original signal, the output of the filter needs to be scaled by M . Note that for each value of M , different multi-band frequency responses are obtained. If these multi-band responses are subtracted from each other or masked using suitably designed masking filters, different low-pass, band-pass and high-pass channels are obtained. This forms the basic principle of the proposed coefficient decimation-based reconfigurable filters and filter banks.

7.1.2 Frequency Response Illustration of CD Approach

The proposed CD approach can be illustrated with the help of Fig. 7.1. The frequency response of the modal filter with normalized (with respect to sampling frequency) passband and stopband specifications of $f_p = 0.05$ and $f_s = 0.075$ is shown in Fig. 7.1 (a). The passband and stopband ripple specifications (δ_p and δ_s respectively) are 0.1 dB and -55 dB respectively. Then according to the expression (6.6), the length of the filter is obtained as 120. The Fig. 7.1 (b) represents frequency response for $M = 2$, i.e., the case when every 2nd coefficient is kept unchanged and remaining coefficients are replaced by zero values. Note that the frequency response is obtained by scaling the coefficients by $M = 2$. In the proposed CD implementation, this can be achieved by scaling the output of the filter by $M=2$. This is possible because convolution holds good for $(M \times h) \otimes x = M \times (h \otimes x)$, where x is the input and h represent the filter coefficients. As seen from (7.7) and Fig. 7.1 (b), for $M = 2$, the frequency responses are obtained at $2\pi k/2 = \pi k$, for $k=0$ and 1. Similarly,

Figures 7.1 (c) and 7.1 (d) represent the case $M=3$ and $M=4$ respectively. Fig. 7.1 (e) is obtained as a special case of $M=4$. If every 4th coefficients are grouped together, a decimated frequency response compared to original frequency response of Fig. 7.1 (a) is obtained with $M=4$.

Figure 7.1 (a) Frequency response of original modal filter.

Figure 7.1 (b) Frequency response of modal filter with $M=2$.

Figure 7.1 (c) Frequency response of modal filter with $M=3$.

Figure 7.1 (d) Frequency response of modal filter with $M=4$.

Figure 7.1 (e) Frequency response of decimated modal filter with $M=4$.

It can be seen from Figures 7.1 (b) to 7.1 (d) that the stopband attenuation reduces as M increases. But it should be noted that the transition-band width and passband width remains unaltered for any M . Therefore, based on the desired stopband attenuation specification of the channel to be extracted, the original modal filter is designed with larger stopband attenuation keeping in account of the deterioration of the final filter's or filter bank's stopband attenuation.

The complexity of the CD approach reduces as M increases. An N -tap FIR filter has N coefficient multiplications whereas the filter whose coefficients are decimated by M will have only N/M multiplications. In this illustrative example, the modal filter in Fig. 7.1 (a) has 120 taps ($N = 120$) and consequently 120 multiplications, but for the case $M=2$, the number of multiplications is only 60. Therefore, even if the initial modal filter in the CD approach is designed with more taps taking into account of the stopband attenuation reduction after coefficient decimation, the effective filter length and overall multiplication complexity of the CD approach is less as coefficient decimation reduces filter length by a factor of M .

7.1.3 Proposed CD-Based Reconfigurable FIR Filter Architecture

The proposed reconfigurable FIR filter architecture based on the CD approach is shown in Fig. 7.2. The architecture in Fig. 7.2 consists of two types of multiplexers, Mux-I and Mux-II. The multiplexers Mux-I are used to select every M^{th} coefficient and to select zeros for all the other coefficients. The main role of Mux-I is to unload the unwanted coefficient multipliers thus saving significant amount of dynamic power. The multiplexers Mux-II are used to bypass the delay-adder unit and to group every M^{th} coefficient together so that decimated frequency response as in the case of Fig. 7.1 (e) can be obtained. The output, y , of the filter needs to be scaled by M to recover the original signal. Thus using the architecture in Fig. 7.2, the following filter specifications can be obtained.

- Original frequency specification
- Multi-band frequency specification
- Decimated frequency specification

The multiple frequency bands can be extracted using suitable masking filters. The masking filters can be designed with wider transition-band widths and consequently low complexity due to lower order as the frequency bands (pass bands) are located sufficiently wide apart as can be seen from Figures 7.1 (b) to 7.1 (d).

Figure 7.2 Architecture of the proposed CD-based FIR filter.

7.1.4 Proposed CD-Based Reconfigurable Filter Bank Architecture

The proposed CD approach can be extended to develop a filter bank, known as coefficient decimation filter bank (CDFB), to extract multiple frequency bands simultaneously, i.e., parallel reception of several channels. This can be illustrated as follows: If the frequency responses in Figures 7.1 (a) to 7.1 (d) are obtained simultaneously using the architecture shown in Fig. 7.3, then by subtracting the outputs of Fig. 7.1 (b) and Fig. 7.1 (c) from Fig. 7.1 (a) and the output of Fig. 7.1 (d) from Fig. 7.1 (b), different frequency bands located at integer multiples of $2\pi/M$ can be extracted. A generalized architecture for the proposed FB is shown in Fig. 7.3. The frequency responses in Figures 7.1 (a) to 7.1 (d) are obtained using the filter bank structure in Fig. 7.3 as outputs y_1 to y_4 respectively. Also, the responses in Figures 7.4 (a) to 7.4 (c) are obtained as outputs $y_2 - y_1$, $y_3 - y_1$ and $y_4 - y_2$ in the architecture shown in Fig. 7.3. Thus the proposed CDFB is capable of extracting channels corresponding to the frequency responses in Figures 7.1 (a) to 7.1 (d) and 7.4 (a) to 7.4 (c) simultaneously without the need of any extra filters or modulation operations.

Note that neither the passband width nor the transition-band width is altered while obtaining all the above frequency responses. An alternative method to spectral subtraction for obtaining individual frequency responses from the multi-band response is to employ frequency masking filters to mask out unwanted bands. In Fig. 7.3, it is shown that the channels y_{41C} and y_{42C} are obtained from the multi-band response y_{4C} using masking filters $HM1$ and $HM2$ respectively. The masking filters $HM1$ and $HM2$ have wide transition-band widths and therefore their complexities are low as they can be realized using short-length filters. The principle of masking filtering employed to obtain the channels y_{41C} and y_{42C} is explained in Section 7.3.

7.2 Design Procedure for the Proposed CD Approach

The steps in the design of proposed CD approach are as follows.

Step 1: Generate an N -tap low-pass filter (modal filter) with desired passband and stopband edge specifications. While fixing the band edges, it should be ensured that aliasing will not occur due to the replacement of coefficients other than every M -th coefficients by zero values. This can be achieved by ensuring that the passband and stopband edges are well within quarter of the Nyquist rate, i.e., the normalized frequency band edges should be within 0.25.

Step 2: Select an appropriate value for M so that desired multi-band frequency response is obtained. The selection of M can be done by using (7.7) which gives the exact locations of the center frequencies of passbands. It should be noted that there is an upper limit on the value of M beyond which aliasing can occur. A check on the upper limit of M can be done by ensuring $Mf_s < 0.5$, where f_s is the stopband edge of the modal filter.

Once the multi-band frequency responses are obtained, the individual responses (low-pass, band-pass and high-pass bands) can be isolated using one of the following steps:

Step 3 (a): By subtracting one multi-band frequency response from the other so that the common frequency bands are subtracted and the desired frequency band will be obtained. This can be done for those multi-band responses for which all the frequency bands other than the desired frequency band are located at same positions in both the multi-band responses. In this case, no additional filter is required.

Step 3(b): If the above case is not satisfied, the individual frequency responses can be isolated by employing masking filters. These masking filters are of low complexity as the transition-band widths are very wide.

Step 4: If the selected M^{th} coefficients are grouped together eliminating the zero coefficients in between, then a frequency response which is a decimated version of original frequency response can be obtained.

7.3 Channelization Comparison – CDFB V/s DFTFB

The proposed CDFB can be employed as a low complexity alternative to a uniformly modulated digital filter bank such as DFTFB. This can be illustrated this with the help of an 8-channel DFTFB example (which can be extended to n -channel FB without loss of generality). The block diagram of 8-channel DFTFB is as shown in Fig. 7.5 (a). The output of an 8-point DFTFB consists of only 5 channels within the quadrature frequency (i.e., $F_s/2$) which are of interest to us. To extract these 5 channels, a frequency response as shown in Fig. 7.5 (b) is required. The peak passband ripple (PPR) is chosen as 0.1 dB and peak stopband ripple (PSR) is chosen as -40dB.

Figure 7.5 (a) 8-Channel DFTFB.

For implementing the 8-channel DFTFB, a prototype filter whose frequency response as that of *Ch1* in Fig. 7.5 (b) is required. The prototype filter in polyphase form is followed

by an 8-point inverse DFT. If a down sampling factor of 8 is brought inside the polyphase components, all the channels (*Ch1* to *Ch5*) can be obtained as baseband channels.

Figure 7.5 (b) Frequency response showing 5-channels.

Figure 7.6 (a) Frequency response at y_1 .

Figure 7.6 (b) Frequency response at y_2 .

Figure 7.6 (c) Frequency response at y_4 .

Figure 7.6 (d) Frequency response at y_{4C} .

Using proposed CDFB, the filter bank shown in Fig. 7.3 can be used to extract the channels, *Ch1* to *Ch5*. In Fig. 7.3, the delay lines for obtaining y_1 , y_2 , y_4 and y_{4C} are only required for obtaining an 8-channel DFTFB equivalent. The coefficients for the modal filter of the proposed FB are same as that for the prototype filter of DFTFB. The outputs at y_1 , y_2 , y_4 and y_{4C} have frequency responses as shown in Figures 7.6 (a) to 7.6 (d) respectively. Note that y_{4C} is obtained by employing complementary delays as shown in Fig. 7.3. The combined channels *Ch2* and *Ch4* in Fig. 7.6 (d) obtained at the output of y_{4C} can be isolated by employing separate masking filters *HMI* and *HM2* (which can be designed with wide transition-band widths and consequently lower order and less complex) as shown in Fig. 7.3. The output at y_1 gives *Ch1*, output at y_{41C} gives *Ch2*, output at y_2 - y_4 gives *Ch3*, output at y_{42C} gives *Ch4* and output at y_2 - y_1 gives *Ch5*. Note that the stopband attenuations of *Ch3* (Fig. 7.6 (c)) *Ch2* and *Ch4* (Fig. 7.6 (d)) are less than that of *Ch1* and *Ch5* (Figures 7.6 (a) and 7.6 (b) respectively). This is due to the fact that decimating by larger values of M would decrease the stopband attenuation as stated in Section 7.1.2. Nevertheless, it can be noted that the stopband attenuations of above channels (*Ch3*, *Ch2* and *Ch4*) still satisfy the ultimate stopband attenuation requirement of -40 dB as that of the DFTFB in Fig. 7.5 (b). Thus by designing the original modal filter with a larger stopband attenuation taking into account of the deterioration of the final filter bank's stopband attenuation, the desired stopband attenuation specification of final filter can be met (i. e. say start with -55 dB so that the desired attenuation of -40 dB can be achieved). The passband and stopband specifications of $f_p = 0.125$ and $f_s = 0.13$ are chosen for the prototype filter in DFTFB and modal filter in CDFB. The lengths of the prototype filter in DFTFB and the modal filter in the proposed CDFB can be obtained from (6.6) as 420 and 520 respectively. In the proposed CDFB, for obtaining *Ch1*, the modal filter has been employed itself, for obtaining other channels (cases with $M=2$ and $M=4$), only some of the coefficients out of 520 are employed (others are considered as zeros). The masking filter lengths for obtaining *Ch2* and *Ch4* are obtained as 20 taps using (6.6). Thus the effective filter length for the proposed CDFB is $520+20 \times 2 = 560$. This is 140 taps more than the DFTFB. However the proposed CDFB do not require any DFT implementation to obtain the channels. For an N -point DFTFB with prototype filter of length L , the total number of multiplications is $L+N^2$. In an SDR channelizer, the number

of channels to be extracted (which is same as N) need not be a power of two. Therefore, while computing the complexity of the filter bank, it is assumed that fast Fourier transform (FFT) is not used so that the computation is valid for cases when N is not a power of two. Note that in the above example, the proposed CDFB has an overhead of $140 - 64 = 76$ multiplications compared to the DFTFB. However the complexity of DFT implementation increases with N and becomes more than that of the CDFB when $N \geq 12$ in above example. It can be seen from [17] that the complexity of DFT implementation becomes approximately same as that of the prototype filter implementation when the number of channels to be extracted is high (typically more than 16 channels). Thus the proposed CDFB is a low complexity alternative to DFTFB when several channels need to be extracted, which is the case in an SDR channelizer.

Thus in the proposed method, employing only one modal filter and without the use of any DFT, the uniformly modulated 8-channel filter bank can be realized at very low complexity. It can be noted that, in DFTFB all the channels are located at integer multiples of $2\pi/M$ and M is fixed (In this example, $M = 8$). But for the proposed FB, M is also variable and hence the centre frequencies of channels can be varied. For example, by employing $M = 3$, the centre frequencies of channels will be integer multiples of $2\pi/3$. The frequency response for case $M = 3$, is shown in Fig. 7.4 (b), which is not realizable by employing DFTFB. Similarly, different values of M will shift the centre frequencies to different $2\pi/M$ locations without any change in the passband width. In short, the value of M can be changed in the proposed FB to extract channels whose centre frequencies are located at $2\pi/M$. This technique can be used to tackle the drawback of fixed channel stacking in DFTFBs.

7.4 Complexity Evaluation

In this section, the complexity comparison of the proposed CDFB with other filter banks in literature is presented. The complexity is compared in the context of receiving multiple channels simultaneously (multiple channel reception complexity) and reconfigurability (reconfiguration complexity).

7.4.1 Multiple Channel Reception Complexity

The complexity associated with the extraction of more than one channel simultaneously is analyzed. Table 7.1 shows the comparison of the multiplication complexity of the proposed channelizer with that of the PC approach, DFTFB, MPRB [18] and GFB [9]. Multiplication complexity of a channelizer is defined as the total number of multiplications for extracting N_f number of channels simultaneously. The multiplications involved in a channelizer can be grouped into three categories: Multiplications associated with (1) Channel filtering; (2) Digital down conversion and (3) Modulation of filters (this is not applicable for PC approach). In Table 7.1, L represents the number of non-zero coefficients of the prototype filter for the PC approach, DFTFB and GFB and l represents the additional number of non-zero coefficients of the modal filter (because of over design) and masking filters in the proposed CDFB (only non-zero coefficients are considered as they will only result in multiplication complexity), and F_s represents the sampling frequency.

The multiplication complexities for PC approach, DFTFB and GFB are taken directly from [9]. From Table 7.1, it is clear that the complexity of PC approach is directly proportional to the number of channels, N_f . Thus higher the number of channels, the PC approach is not hardware efficient. It can be seen that, the complexity of filtering (multiplication) operation is same for DFTFB and GFB and slightly higher for CDFB (because of overdesigning and masking filters). The MPRB [18] consists of an analysis DFTFB and a synthesis DFTFB and hence the complexity is exactly twice that of DFTFB. As mentioned in Section 7.3, there is no modulation complexity associated with the proposed CDFB. However separate (N_f-1) digital down converters are required in the CDFB for converting all the channels except the low-pass channel to baseband. The FFT has not been considered for the implementation of IDFT in DFTFB and MPRB, as FFT is appropriate only if the number of channels to be extracted is a power-of-two. From Table 7.1, it can be seen that the complexity of the proposed CDFB is lower than that of the PC approach, MPRB and GFB. Also, the proposed CDFB is less complex compared to DFTFB, when the number of channels, N_f , increases as discussed in Section 7.3. It is to be noted that, the multiplication complexity presented in this section is only a theoretical

estimate of complexity and the actual complexity evaluation is presented in Section 7.6 which shows the implementation costs in FPGA.

Table 7.1
Multiplication complexity of channelizers

	PC Approach	DFTFB	MPRB [11]	GFB [2]	Proposed Filter bank (CDFB)
Filter	$N_I \bullet L$	L	$2L$	L	$L + l$
Down conversion	$N_I - 1$	-	-	-	$N_I - 1$
Modulation of Filters	-	N_I^2	$2N_I^2$	$N_I \bullet L$	-
Total multiplications	$N_I \bullet (L + 1) - 1$	$L + N_I^2$	$2(L + N_I^2)$	$L \bullet (1 + N_I)$	$L + l + N_I - 1$

7.4.2 Reconfiguration Complexity

Reconfiguration complexity is defined as the cost associated with the reconfiguration of the filter bank when the receiver switches its operation from one standard (current standard) to another one (new standard). Ideally, the reconfigurability of the filter bank must be accomplished by reconfiguring the same prototype filter in the filter bank to process the signals of the new communication standard with the least possible overhead, instead of employing separate filter banks for each standard. However reconfiguration of DFTFB and its modified architectures such as GFB would suffer from following overheads compared to proposed CDFB:

1. The prototype filter needs to be reconfigured. Generally DFTFB and GFB employ polyphase decomposition. Hence reconfiguration can involve changing the number of polyphase branches which is a tedious and expensive task.
2. Downsampling factor needs to be changed. As a result of this, it is not appropriate to do down sampling before filtering. Hence for SDR receivers, the advantage of incorporating the digital down sampling to the left of filtering is not always feasible. Thus the prototype filter needs to operate at the same speed of ADCs.
3. The DFT needs to be reformulated according to the new polyphase architecture which is also expensive.

For example, if the circuit is switching from a 8-channel FB to 16-channel FB, the number of polyphase branches need to be changed from 8 to 16 (first limitation of DFTFB stated in Section 1). Also the prototype filter coefficients and the number of coefficients per polyphase branch need to be changed which becomes very expensive because of the higher order of the prototype filter. The downsampling factor needs to be adjusted from 8 to 16 (second limitation of DFTFB) and the 8-point DFT needs to be expanded to 16-point DFT. On the contrary, these limitations do not exist in proposed CDFB. Note that the down sampling factor in the proposed method is independent on the communication standard. Hence the reconfiguration complexity of CDFB is much less compared to the DFTFB and GFB.

7.5 Design Example

A dual standard code division multiple access (CDMA)/Wideband CDMA (WCDMA) SDR channelizer is presented for illustration. The sampling rate chosen is 20 MHz. The proposed FB can extract 1250 kHz CDMA channels and 5 MHz WCDMA channels from the input signal simultaneously. Assume a dual standard co-existence scenario where the input signal has both the CDMA band and the WCDMA band. Let the CDMA signals be located in the range from 5000 kHz to 6250 kHz and 8000 kHz to 9250 kHz (two CDMA channels of bandwidth 1250 kHz each) and the WCDMA signal is located in the range from 0 to 5000 kHz (one WCDMA channel of 5 MHz bandwidth). The simultaneous extraction of CDMA and WCDMA channels is very expensive in the case of DFTFB or GFB. The peak passband ripple (PPR) is chosen as 0.1 dB. The filter stop-band specification (adjacent channel attenuation specification) is chosen as -40 dB for both the standards. Although the passband and stopband ripples are chosen as same for both CDMA and WCDMA, the proposed architecture can handle different passband and stopband ripples. This is achieved by designing the modal filter for the worst case ripple specifications.

Figure 7.7 (a)Frequency Response with M=7.

Figure 7.7 (b)Frequency Response with M=3.

Figure 7.7 (c)Frequency Response with M=5.

The modal filter is designed for the CDMA response i.e., a low-pass filter with passband width of 625 kHz ($f_p = 0.625/10 = 0.0625$ and $f_s = 0.08$). By using $M = 7$, in the architecture of Fig. 7.3, the frequency response as shown in Fig. 7.7 (a) is obtained. Thus CDMA channels (CDMA1 and CDMA2 as shown in Fig. 7.7 (a)) can be obtained from Fig. 7.3 by employing masking filters. The WCDMA subbands (W1 and W3) can be obtained by using suitable masking filters at the output of $M=7$ and the other two subbands W2 and W4 can be obtained by using masking filters at the output of complementary to $M=7$ case. Thus by using a single case of $M=7$, and by using four masking filters, it is possible to extract the WCDMA and CDMA channels

Figure 7.8 (a) Spectrum of WCDMA output and input signal.

Figure 7.8 (b) Spectrum of CDMA output and input signal.

simultaneously. Using (6.6), the filter length of the modal filter for the proposed CDFB is found to be 500 and that of masking filters as 60. However only the case $M=7$ is used and hence the effective length of the modal filter is $500/7 = 72$. Thus the total effective filter length = $72 + 60 \times 4 = 312$. Using DFTFB, the prototype filter alone will require a filter length of 400 (from (6.6)) for obtaining the CDMA output. Thus in this design example, the prototype filter complexity of the DFTFB is more than that of the CDFB. The DFTFB also requires the implementation of IDFT which increases the complexity where as the proposed CDFB does not require any IDFT for its operation. Also a DFTFB-based channelizer would require separate filter banks for simultaneous reception of CDMA/WCDMA signals as the channel bandwidths of these two standards are different. Two special cases of CDMA channel extraction using the proposed CDFB are shown in Figures 7.7 (b) and (c). In Fig. 7.7 (b), the case, $M = 3$ is shown. In this case, the CDMA channels located at baseband and $6.67 \text{ MHz} (= 2\pi/3)$ can be obtained from y_3 in the architecture of Fig. 7.3. Similarly, for the case $M = 5$ shown in Fig. 7.7 (c), the CDMA channels located at baseband, $4 \text{ MHz} (= 2\pi/5)$ and $8 \text{ MHz} (= 4\pi/5)$ can be obtained from y_5 (y_M where $M = 5$) in the architecture of Fig. 7.3. These CDMA channels located at $2\pi/3$, $2\pi/5$ and $4\pi/5$ cannot be extracted using a single DFTFB because of fixed channel stacking. This clearly shows the ability to adapt to the center frequencies of the passbands of the desired channels in the proposed CDFB. Fig. 7.8 shows the spectrum of the output and input for the CDMA/WCDMA design example. From Fig. 7.8, for the WCDMA case, the mean error in passband and stopband are found to be 0.0439 and 0.0987 respectively. Similarly for both the CDMA channels, the mean error in passband and stopband are

found to be 0.0345 and 0.0876 respectively. Thus it can be concluded that the input spectrum closely resembles the output spectrum.

Note that a dual standard channelizer is used for ease of explanation and the proposed CDFB can be easily extended for simultaneous reception of signals of more than two standards. Also, there is no restriction on the total number of received channels to two.

7.6 Implementation Results

In this section, the implementation result of 8-channel filter bank based on DFTFB and the proposed CDFB discussed in Section 7.3 are presented. The implementation has been done on Xilinx XC40150XV-09-BG560 FPGA with input wordlength of 8 bits and coefficient wordlength of 12 bits. The design details are as given in Section 7.3. Table 7.2 shows the implementation results. The implementation area in terms of configurable logic blocks (CLBs) and timing in terms of frequency (MHz) are used for comparing the two techniques. The implementation results in terms of full design and design of individual components are also shown. For implementing the prototype FIR filter bank, the DFTFB requires 2352 CLBs of available 5184 CLBS, which means 45.6% area occupancy of available chip area. The proposed FB requires 1200 CLBs more compared to DFTFB. These additional CLBs include the additional delay lines for each M and necessary masking filters. The DFTFB requires extra area for DFT core, which occupies additional 45.4% of chip area, whereas no DFT is required for the proposed FB. Over-all design of DFTFB occupies 83.4% of chip area, whereas the proposed FB occupies only 69.4% of chip area. Timing analysis also shows that the proposed FB offers speed improvement of 23.6% over DFTFB, which is significant in SDR channelizers.

Table 7.2
Implementation results

Design	DFTFB		Proposed CDFB	
	Area (CLBs)	Speed (MHz)	Area (CLBs)	Speed (MHz)
FIR filter bank	2352/5184	30.5	3552/5184	30.5
DFT	2350/5184	18.6	0/5184	0
Full Design	4321/5184	19.4	3600/5184	25.4

7.7 Summary

In this chapter, a new method for designing reconfigurable filters and filter banks based on the multirate signal processing concept of decimation was proposed. In the proposed coefficient decimation (CD) approach, firstly design an N -tap low-pass FIR filter as the modal filter. Subsequently, every M -th coefficient of the modal filter is retained and all other coefficients are replaced with zero values. The frequency response of the resulting decimated modal filter will have replicas of the passband of the modal filter at integer multiples of $2\pi/M$. The desired channels of different bandwidths can be extracted from the identical bandwidth spectrum replicas of the decimated modal filter using one or more of following operations - subtraction, frequency masking or complementary filtering. As all the desired channels are obtained using the same modal filter, a fixed-coefficient implementation is feasible for the modal filter, which leads to a low complexity filter bank. Furthermore, as coefficient decimation reduces the number of nonzero coefficients of the original modal filter to N/M , the complexity of the resulting filter bank is substantially reduced. The proposed design also has the advantage that the passband widths and the centre frequencies of passbands can be easily changed to extract the channels located almost anywhere in the available bandwidth of wideband signal. Thus the proposed coefficient decimated filter bank (CDFB) can be employed as a low complexity and highly flexible alternative to inherently less flexible and high complex DFTFB. Also the proposed CDFB solves the problem of lack of control over the passband width and location of passbands, associated with frequency response masking based filter banks presented in Chapter 6.

Chapter 8

Conclusions and Future Work

In this chapter, a brief summary of the works accomplished in this dissertation is presented. Some topics for future work in this research area are also discussed.

8.1 Conclusions

This thesis addressed the hardware-efficient implementation issues of the reconfigurable channelizers in the digital front end of a software defined radio (SDR) receiver. The channelizer extracts narrowband channels from a wideband input signal using digital filter banks, down converts each of these narrowband channels to baseband using digital down converters and finally reduce the sampling rate of the narrowband channels to the sampling rate of the desired standard using a sample rate converter. The channelizer comes directly after the ADC and thus needs to operate at very high sampling rate. Also the channelizer needs to be implemented with low power to work on battery operated devices. In addition to this, reconfigurability is also a requirement in the channelizer to support the multi-standard operation. Thus, a reconfigurable, low power and high speed channelizer is an essential part of SDR receiver. Digital filter banks, which extract the narrowband channels from the wideband input signal, constitute the main power consuming and area consuming block in the channelizer. In this thesis, new techniques to implement reconfigurable area-efficient digital filter banks with low power and high speed for SDR channelizers have been proposed.

Digital filters are basic building blocks for any digital filter bank. Hence optimization of digital filters is an essential requirement of digital filter banks employed for channelization in SDR receivers. FIR filters are widely employed in digital filter banks as they have absolute stability and linear phase property which are essential for wireless communication systems. The most computationally intensive operation in an FIR filter is coefficient multiplication. Since higher-order FIR filters, i. e., filters with several hundreds of coefficients, are required in SDR receivers to meet

the stringent adjacent channel attenuation specifications of wireless communication systems, low complexity techniques such as common subexpression elimination (CSE) algorithms have been employed to reduce the complexity of these digital filters. In CSE algorithms, coefficient multiplications are considered as shift and add operations. The goal of CSE is to identify multiple occurrences of identical bit patterns, called common subexpressions (CSs), that are present in the coefficients, and eliminate redundant multiplications (due to these CSs) to minimize the number of adders needed to realize the coefficient multipliers. Conventionally CSE algorithms use canonical signed digit (CSD) representation filter coefficients because of reduced number of non-zero digits compared to binary representation of filter coefficients. But in the first work in this thesis, it was found that statistically the number of adders required to implement the coefficient multiplication by employing any CSE technique is heavily dependent on the number of un-paired bits i. e., bits which do not form CSs and are left unpaired after the application of CSE algorithm. It was found that, the number of un-paired bits is least when the filter coefficients are represented in the binary form. It was also reported in [42] that CSD-based CSE techniques have constraints in reducing the number of adders due to the presence of signed bits, which may require additional adders. Based on these findings, a binary representation-based CSE algorithm known as binary subexpression elimination (BSE) algorithm was proposed. The reduction of adders achieved using the proposed BSE method is slightly inferior for short filters. However, the BSE method offers better reduction of adders for higher-order filters. Therefore, the proposed BSE method is best suited for implementing higher-order FIR filters in SDR channelizers. The design examples show that the proposed BSE method offers an average adder reduction of 24% over the method in [39] and 18% over method in [46].

In the second work, reconfigurability was incorporated into the proposed BSE algorithm to implement reconfigurable low complexity channel filters. Two new methods were proposed, a constant shifts method (CSM) and a programmable shifts method (PSM), for implementing channel filters. In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, the binary common subexpressions-based shift and add unit was used in the proposed CSM and PSM architectures, which results in better reduction of addition operations compared to conventional shift and add units. In the CSM, the filter coefficients, which are

directly stored in look up tables (LUTs) are partitioned into fixed groups of three bits and are used as select signal for multiplexers to select the appropriate partial product from the shift and add unit. As the partitioning of coefficients is fixed, the shifts involved in the CSM are constant shifts and hence can be hardwired. On the other hand, in the PSM, the filter coefficients are optimized using the proposed BSE algorithm and stored in LUTs in a coded format. Thus the shifts in PSM are variable and hence programmable shifters (PSs) are required. The use of PS results in reduced speed of operation for the PSM architecture compared to the CSM architecture. However in the PSM, the BSE algorithm is applied to filter coefficients before they are stored in LUTs. Hence the PSM results in low power implementations when compared to the CSM. The proposed CSM and PSM architectures have been implemented on Virtex-II 2v3000ff1152-4 FPGA and 0.18 μ m CMOS technology with a coefficient wordlength of 16 bits and compared with various reconfigurable FIR filter architectures proposed in literature. Design example of D-AMPS channel filter shows that the PSM architecture offers an average reduction of 23% in the number of addition operations compared to other FIR filter implementations. The proposed reconfigurable architectures are not restricted to BSE method, and can be easily modified to employ any other CSE method.

FIR filters in SDR receivers require sharp transition-bands and thus very higher-order because of stringent adjacent channel attenuation specifications. The higher-order FIR filters have large number of multiplications and consume significant power and thus have high complexity. A frequency response masking (FRM) technique was employed in [75] to design sharp transition-band FIR filters with low complexity. The basic idea was to compose the over-all FIR filter with sharp transition-band requirements using three wide transition-band FIR filters i.e., one modal filter and two masking filters. The modal filter is a special type of filter, where each unit delay of the normal filter is replaced by M delays. This is equivalent to interpolating the frequency response of the modal filter by M . The output of the modal filter consists of $M+1$ frequency bands. The complementary to the output at the modal filter can be obtained using simple delays. The un-desired frequency bands at the outputs of modal filter and complementary delays are masked out by employing two masking filters. The outputs of the masking filters can be summed up to obtain wider passband frequency bands. Since the subfilters have wide transition-band

specifications, resulting filters are of lower-order. Consequently, the over-all filter complexity will be much less than conventional design of sharp transition-band FIR filters. As the third work, reconfigurability has been extended into the inherently less complex FRM architecture. A new reconfigurable FIR filter which can dynamically change the frequency response characteristics with low computational overhead has been proposed. The proposed reconfigurable filter architecture consists of a modal filter with the delay line (structural adder line) consisting of a multiplexer which can dynamically select different values of M to obtain different frequency responses. The masking filters are also programmable similar to the proposed CSM and PSM architectures and thus different frequency response specifications can be dynamically obtained using the proposed FIR filter architecture. FIR filter banks based on the FRM technique were also proposed. The proposed filter bank consists of a common hardware block at the front-end (modal filter) for multiple communication standards and a reconfigurable masking filter at the back-end. The complexity of the channelizer is dominated by the block at the front-end as the order of the modal filter is substantially higher than that of the masking filter. Since the front-end hardware in proposed scheme is the same (common) for all the communication standards, its complexity can be significantly reduced using the proposed BSE method, which will ensure low complexity of the over-all channelizer. Coefficient multiplication, which is the most power consuming operation, is done only once for obtaining different frequency responses simultaneously in the proposed modal filter architecture. As a result, power consumption of the modal filter is less. The proposed reconfigurable filter bank has the advantages of extracting non-uniform bandwidth channels, extracting extremely narrowband channels and multi-mode operation, which were not achieved using conventional filter banks. The proposed filter bank has been compared with conventional per-channel (PC) approach, DFT filter bank (DFTFB), Goertzel filter bank (GFB) [9] and the modulated perfect reconstruction bank (MPRB) [18] and implemented and tested the proposed filter bank on Virtex FPGA. Design examples show that the proposed filter bank offers complexity reduction of 83.5% over PC approach, 70.8% over DFTFB, 72% over GFB and 81% over MPRB.

In SDR channelizers, the filter banks need to extract channels, which have bandwidths that are related by non-integer factors. Based on the FRM approach, a new filter bank has been proposed, which can extract channels, whose bandwidths are

related by fractional factors whose extraction is not an easy task in conventional filter banks. This forms the fourth contribution. The proposed filter bank architecture consists of multiple stages of FRM approach and can be operated in three modes. The three modes of operation are based on changing either the frequency specifications of the filters or the value of delays, M , or both. These three modes of operation give much more flexibility to the proposed filter bank compared to the existing filter bank approaches.

As the fifth contribution a new method has been devised for designing reconfigurable filters and filter banks based on the multirate signal processing concept of decimation. In the design called coefficient decimation-based approach, firstly an N -tap lowpass FIR filter as the modal filter is designed. Subsequently, every M -th coefficient of the modal filter is retained and all other coefficients are replaced with zero values. The frequency response of the resulting decimated modal filter will have replicas of the passband of the modal filter at integer multiples of $2\pi/M$. The desired channels of different bandwidths can be extracted from the identical bandwidth spectrum replicas of the decimated modal filter using one or more of following operations - subtraction, frequency masking or complementary filtering. If the retained M -th coefficients are grouped together, then a frequency response which is a decimated version of original frequency response can be obtained. As all the desired channels are obtained using the same modal filter, a fixed-coefficient implementation is feasible for the modal filter, which leads to a low complexity implementation. Furthermore, as coefficient decimation reduces the number of nonzero coefficients of the original modal filter to N/M , the complexity of the resulting filter or filter bank is substantially reduced. The proposed design also has the advantage that the passband widths and the centre frequencies of passbands can be easily changed to extract the channels located almost anywhere in the available bandwidth of wideband signal, which was not possible in previous filter banks. Thus the proposed coefficient decimated filter bank can be employed as a low complexity and highly flexible alternative to inherently less flexible and highly complex DFT filterbanks.

8.2 Future Work

In this dissertation, the challenges in designing reconfigurable low complexity digital filter banks for SDR receivers have been investigated. Five new techniques to tackle these challenges were proposed. However many optimization possibilities to pursue further research in this area in future have been identified. Some of them are discussed below:

1) Optimization of the proposed BSE algorithm

In future, it is interesting to address the possibilities of further hardware optimization possibilities of channel filters used in the SDR receiver. It is mentioned in [48] that, the actual cost of coefficient multiplication operation is determined by the number of full adders (FAs) required for the realization of each adder in the multiplier, called adder-width. A coefficient partitioning method (CPM) was proposed in [50] which resulted in the best FA reduction compared to other approaches in literature. However the CPM was applied to CSD-based CSE algorithms. In Chapter 4, it was shown that the CSE algorithm is more efficient when applied to coefficients represented in the binary form. As a future work, it is worth to incorporate the CPM approach into the proposed BSE algorithm to implement low complexity FIR filters for SDR receivers.

It is also possible to significantly reduce the number of FAs by efficiently hardwiring the partial product adders. The approach can be illustrated using the example of an 8-bit quantized input signal, $x = 0.11111111$, and a 4-bit coefficient, $h_k = 1.001$. The filter tap output, y_k is given by $y_k = x + 2^{-3}x$. If r_1 and r_2 are the ranges (number of bits) of the operands of an adder, such that $r_2 > r_1$, then $(r_2 + 1)$ FAs are needed to realize the adder [50]. This requirement of 1 FA more than r_2 FAs is due to the extended signed bit position for two's complement addition to take care of the overflow when there is a need to sign extend the operands. The adder assumed is a ripple carry adder on account of its low power consumption as in [50]. For y_k , r_1 is 8 and r_2 is 11. Thus the direct implementation (implementation using shifts and additions) of y_k would require 12 FAs. However it is possible to reduce the number of FAs by hardwiring the datapath corresponding to the bits that do not essentially require FAs to compute the sum. For an adder, if S_1 and S_2 are the shifts of the operands P_1 and P_2 respectively, the most significant $S_1 - 1$ bits of P_1 can be hardwired

as zero bits to the output of the adder, and the least significant S_2-S_1 bits of P_2 can be hardwired by tapping the original S_2-S_1 bit values to the output. In Fig. 8.1, the datapath corresponding to the bits inside the solid rectangle is hardwired. Therefore, the addition of the bits inside the solid rectangle does not require FAs. Thus FAs are used only for computing the sum of bits inside the dotted rectangle in Fig. 8.1. Using hardwiring, only 9 FAs are required to compute y_k as shown in Fig. 8.1 which means FA reduction of 25% over the direct approach.

Figure 8.1 Hardwiring of partial product adders.

2) Optimization of proposed Reconfigurable FIR filter architectures

In the CSM architecture, the final adder unit is implemented using binary adder tree structure. However it is possible to optimize further by using carry save adders and carry look-ahead adders. The 3:2 or 4:2 compressors can be employed for carry free addition making the entire final adder unit more power efficient with improved speed of operation. In future, different methods to further optimize the CSM and PSM architectures will be tackled.

It can also be noted that, in case of shift and add architectures, if the shift difference between the operands is greater than the length of the shorter operand, the operands can be hardwired instead of using adders. This sort of hardwiring can result in significant reduction of adders and full adders for the implementation of FIR filters. As a subsequent step, it is intended to look for incorporating the hardwiring of partial product adders to obtain minimum number of full adders which can result in a reconfigurable FIR filter with minimum number of additions at adder and full adder levels.

3) Optimization of FRM-based filters and filter banks

In this dissertation, reconfigurable low complexity filters and filter banks based on the FRM technique [75] have been proposed. However if the desired frequency response has very narrow transition-band width, the modal filter needs to be interpolated using a

very high value of M . When M is large, the resulting $(M+1)$ multi-band channels lie closer compared to the case when M is small. Consequently, higher-order masking filters would be required to extract the bands of interest. This will increase the over-all cost of the proposed filters and filter banks. This cost can be reduced by employing multiple stages of the FRM technique. In a multi-stage FRM approach, the frequency response of single stage is employed as the modal filter for the next stage. In other words, the modal filter as well as the masking filters is interpolated. As a result of this, there is more flexibility of changing delays in the modal filter and masking filters, which provides multiple levels of flexibility for such filters and filter banks.

Bibliography

- [1] J. Mitola, *Software radio architecture*, John Wiley, 2000.
- [2] T. Hentschel, M. Henker, and G. Fettweis, "The digital front-end of software radio terminals," *IEEE Personal Communications Magazine*, vol. 6, no. 4, pp. 40-46, Aug. 1999.
- [3] A. Wiesler and F. K. Jondral, "A software radio for second- and third-generation mobile systems," *IEEE Transactions on Vehicular Technology*, vol. 51, No. 4, pp. 738-748, July 2002.
- [4] E. Buracchini, "The software radio concept," *IEEE Communications Magazine*, vol. 38, no. 9, pp. 138-143, Sep. 2000.
- [5] T. Hentschel and G. Fettweis, "Software radio receivers," *CDMA techniques for third generation mobile systems*, Kluwer Academic Publishers, 1999.
- [6] J. Mitola, *Software radio technologies: selected readings*, IEEE Press, New York 2001.
- [7] R. H. Walden, "Performance trends for analog-to-digital converters," *IEEE Communications Magazine*, vol. 37, no.2, pp. 96-101, Feb. 1999.
- [8] A. Haghighat, "A review on essentials and technical challenges of software defined radio," in *Proceedings of military communications conference*, vol. 1, pp. 377-382, Oct. 2002.
- [9] T. Hentschel, "Channelization for software defined base-stations," *Annales des Telecommunications*, ISSN 0003-4347, vol. 57, no. 5-6, pp. 386-420, May-June 2002.
- [10] Lee Pucker, "Channelization techniques for software defined radio," in *Proceedings of SDR forum conference*, Nov. 2003.
- [11] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice Hall, 1993.
- [12] T. Hentschel, *Multirate Systems for Sample Rate Conversion in Software Radio Terminals*, Artech House Books, 2002.
- [13] P. P. Vaidyanathan, "Multirate digital filters, filter banks, polyphase networks and applications: A tutorial," *Proceedings of the IEEE*, vol. 78, no. 1, pp. 56-93, Jan 1990.
- [14] E. B. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 2, pp. 155-162, Apr 1981.
- [15] Phil Schniter, <http://cnx.org/content/m10424/latest/>
- [16] C Kim, Y Shin, S Im, and W Lee, "SDR-based digital channelizer/de-channelizer for multiple CDMA signals," in *Proceedings of IEEE Vehicular technology Conference*, vol. 6, pp. 2862-2869, 2000.

- [17] CY Fung and SC Chan, "A multistage filter bank-based channelizer and its multiplier less realization", in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol.3, pp. 429-432, May 2002.
- [18] W. A. Abu-Al-Saud and G. L. Stuber, "Efficient wideband channelizer for software radio systems using modulated PR filter banks," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, pp. 2807-2820, Oct. 2004.
- [19] J. Lillington, "Comparison of wideband channelization architectures," Apr. 2003. <http://www.techonline.com/community/techgroup/dsp/tech paper/33204>
- [20] A. P. Vinod, E. M-K. Lai, A.B. Premkumar, and C.T.Lau, "A reconfigurable multi-standard channelizer using QMF trees for software radio receivers," in *proceedings of 14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communication*, pp. 119-123, Sep. 2003.
- [21] J. Lillington, "TPFT - Tunable pipelined frequency transform," RF Engines Ltd, Technical Report, Sep. 2002. <http://www.rfel.com/whipapdat.asp>
- [22] K. Zangi and R. Koilpillai, "Software radio issues in cellular base stations," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 561-573, Apr. 1999.
- [23] K. Zangi and R. Koilpillai, "Efficient filter bank channelizers for software radio receivers," in *Proceedings of International Conference on Communications*, vol. 3, pp. 1566-1570, June 1998.
- [24] David B. Chester, "Digital IF filter technology for 3G systems: an introduction," *IEEE Communications Magazine*, vol. 37, no. 2, pp. 102-107, Feb. 1999.
- [25] J. G. Prokias and D. G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*, Prentice-Hall, Upper Saddle River, NJ 1998.
- [26] G. K. Ma and F. J. Taylor, "Multiplier policies for digital signal processing," *IEEE Signal Processing Magazine*, vol. 7, no. 1, pp. 6-20, Jan 1990.
- [27] L. H. Garner, "Number systems and arithmetic," *Advances in Computers*, vol. 6, pp. 131-194, Academic Press, Orlando, Florida 1965.
- [28] 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*, vol. 42, no. 7, pp. 453-460, Jul. 1995.
- [29] S. Arno and F.S Wheeler, "Signed digit representation of minimal hamming weight," *IEEE Transactions on Computers*, vol. 42, no. 8, pp. 1007-1010, Aug. 1993.
- [30] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044-1047, Jul. 1989.
- [31] D. Li, Y. C. Lim, Y. Lian and J. Song, "A polynomial-time algorithm for designing FIR filters with powers-of-two coefficients," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1935-1941, Aug. 2002.

- [32] M. Haseyama and D. Matsuura, "A filter coefficient quantization method with genetic algorithm, including simulated annealing", *IEEE Signal Processing Letters*, vol. 13, no. 4, pp. 189-192, April 2006.
- [33] M. Potkonjak, M. B. Srivastava, and P. A. Chandrakasan, "Multiple constant multiplications: 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, Feb. 1996.
- [34] M. Mehendale, S. D. Sherlekar, and G. Venkatesh, "Synthesis of multiplierless FIR filters with minimum number of additions," in *Proceedings of the 1995 IEEE/ACM International Conference on Computer-Aided Design*, pp. 668-671, IEEE Computer Society Press, Nov. 1995.
- [35] 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, Oct. 1996.
- [36] M. Yagyu, A. Nishihara, and N. Fujii, "Fast FIR digital filter structures using minimal number of adders and its application to filter design," *ICICE Transactions on Fundamentals of Electronics Communications and Computer Science*, vol. E79-A, no. 8, pp. 1120-1129, Aug. 1996.
- [37] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "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, Jan. 1999.
- [38] I. C. Park and H. J. Kang, "Digital filter synthesis based on minimal signed digit representation," in *Proceedings of Conference on Design Automation*, pp. 468-473, June 2001.
- [39] 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, Mar. 2002.
- [40] C. Y. Yao, H. H. Chen, 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, Nov. 2004.
- [41] Y. Jang and S. Yang, "Low-power CSD linear phase FIR filter structure using vertical common subexpression," *Electronics letters*, vol. 38, no. 15, pp. 777-779, Jul. 2002.
- [42] A. P Vinod and E. M-K Lai, "Comparison of the horizontal and the vertical common subexpression elimination methods for realizing digital filters," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 496 – 499, May 2005.
- [43] D. R. Bull and D. H. Horrocks, "Primitive operator digital filters," *IEE Proceedings on Circuits, Systems and Devices*, vol. 138, no. 3, pp. 401-412, June 1991.
- [44] A. G. Dempster and M.D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Transactions on Circuits and Systems-II*, vol. 42, no. 9, pp. 569-577, Sep. 1995.

- [45] 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, Aug. 2001.
- [46] F. Xu, C. H. Chang, and C. C. Jong, "Contention resolution algorithm for common subexpression elimination in digital filter design," *IEEE Trans. on Circuits and Systems-II*, vol. 52, no. 10, pp.695-700, Oct. 2005.
- [47] Y. Voronenko and M. Pushcel, "Multiplierless multiple constant multiplication," *ACM Transactions on Algorithms*, vol. 3, no. 2, pp. 1-39, May 2007.
- [48] A.P. Vinod and E. M-K. Lai, "On the implementation of efficient channel filters for wideband receivers by optimizing common subexpression elimination methods", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 2, pp. 295-304, Feb. 2005.
- [49] D. Maskell, and J. Liewo, 'Hardware efficient FIR filters with a reduced adder step' *Electronic letters*, vol. 41, No.22, pp. 1211-1213, Oct. 2005.
- [50] 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, Dec. 2005.
- [51] T. Solla and O. Vainio, "Comparison of programmable FIR filter architectures for low power," in *Proceedings of 28th European Solid-State Circuits Conference*, pp. 759-762, Sept. 2002.
- [52] T. Solla, R. Mäkelä, M. Liljeroos, and O. Vainio, "Application-specific filter processor for flexible receivers," in *Proceedings of 19th NORCHIP Conference*, pp. 53-58, Nov. 2001.
- [53] D. Hwang, C. Mittelsteadt, and I. Verbauehede, "Low power showdown: comparison of five DSP platforms implementing an LPC speech codec," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol.2, pp. 1125-1128, May 2001.
- [54] K. H. Chen and T. D. Chiueh, "A low-power digit-based reconfigurable FIR filter," *IEEE Transactions on Circuits and Systems-II*, vol. 53, no. 8, pp. 617-621, Aug. 2006.
- [55] T. Zhangwen and 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, Nov. 2002.
- [56] X. Chenghuan, C. He, Z. Shunan and W. Hua, "Design and implementation of a high speed programmable polyphase FIR filter," in *Proceedings of 5th International Conference on ASIC*, vol. 2, pp. 783-787, Oct. 2003.
- [57] S. S. Demirsoy, I. Kale, A. G. Dempster, "Efficient implementation of digital filters using novel reconfigurable multiplier blocks," in *Proceedings of Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 461-464, Nov. 2004.
- [58] P. Tummelshammer, J. C. Hoe and M. Puschel, "Time-multiplexed multiple-constant multiplication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 9, pp. 1551-1563, Sept. 2007.

- [59] K. Muhammad and K. Roy, "Reduced computational redundancy implementation of DSP algorithms using computation sharing vector scaling," *IEEE Transactions on VLSI Systems*, vol. 10, no. 3, pp. 292-300, June 2002.
- [60] J. Park, W. Jeong, H. Mahmoodi-Meimand, Y. Wang, H. Choo and K. Roy, "Computation sharing programmable FIR filter for low-power and high-performance applications," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 2, pp. 348-357, Feb. 2004.
- [61] R. A. Walker and D. E. Thomas, A Survey of High-Level Synthesis Systems, Kluwer, Boston, MA, 1991.
- [62] Z. Iqbal, M. Potkonjak, S. Dey, and A. Parker, "Critical path minimization using retiming and algebraic speed-up," in *Proceedings of Conference on Design Automation*, pp. 573-577, June 1993.
- [63] A. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and R. W. Brodersen, "Optimizing power using transformations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 1, pp. 12-31, Jan. 1995.
- [64] 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. 30, no. 10, pp.723-739, Oct. 1983.
- [65] N. Spencer, "An overview of digital telephony standards," in *Proceedings of IEE Colloquium on Design of Digital Cellular Handsets*, pp. 1/1 1/7, March. 1998.
- [66] R. Bagheri, A. Mirzaei, M. E. Heidari, S. Chehrazi, L. Minjae, M. Mikhemar, W. K. Tang, A. A. Abidi, "Software-defined radio receiver: dream to reality," *IEEE Communications Magazine*, vol. 44, no. 8, pp. 111-118, Aug. 2006.
- [67] A. A. Abidi, "The path to the software-defined radio receiver," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 5, pp. 954-966, May 2007.
- [68] Behrooz Parhami, Computer Arithmetic: Algorithms and Hardware Designs, Oxford Press, New York, 2000.
- [69] M. Meribout and M. Motomura, "A combined approach to high-level synthesis for dynamically reconfigurable systems," *IEEE Transactions on Computers*, vol. 53, no. 12, pp. 1508-1522, Dec. 2004.
- [70] X-J Zhang, K-W Ng and G. H. Young, "High-level synthesis using genetic algorithms for dynamically reconfigurable FPGAs," in *Proceedings of 23rd Euromicro Conference*, pp. 234-243, Sep. 1997.
- [71] R. Kress and A. Pyttel, "High-level synthesis for dynamically reconfigurable hardware/software systems," in *Proceedings of the 8th International Workshop on Field-Programmable Logic and Applications, From FPGAs to Computing Paradigm*, pp. 288-297, 1998.
- [72] A. Rettberg, F.-J. Rammig, "Integration of energy reduction into high-level synthesis by partitioning," in *Proceedings of the Working Conference on Distributed and Parallel Embedded Systems (DIPES 2006)*, pp.225-234, Oct. 2006.

- [73] N. Moreano, E. Borin, C. de Souza, and G. Araujo, "Efficient datapath merging for partially reconfigurable architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 7, pp. 969-980, July 2005.
- [74] http://www.lyrtech.com/DSP-development/dsp_fpga/signalmaster_quad_cpci.php
- [75] Y. C. Lim, "Frequency-response masking approach for the synthesis of sharp linear phase digital filters," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 4, pp. 357-364, Apr. 1986.
- [76] R. D. Koilpillai and P. P. Vaidyanathan, "Cosine-modulated FIR filter banks satisfying perfect reconstruction," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 770-783, Apr. 1992.
- [77] M. Bellanger, "On computational complexity in digital filters," in *Proceedings of the European Conference on Circuit Theory and Design*, pp. 58-63, Aug. 1981.
- [78] M. Jian, W. H. Yung and B. Songrong, "An efficient IF architecture for dual mode GSM/W-CDMA receiver of a software radio," in *Proceedings of IEEE International Symposium on Mobile Multimedia Communications*, pp. 21-24, Nov. 1999.
- [79] H. R. Karimi, N. W. Anderson and P. McAndrew, "Digital signal processing aspects of software definable radios," *IEE Colloquium on Adaptable and Multistandard Mobile Radio Terminals*, pp. 3/1-3/8, May 1998.
- [80] M. Radparvar, "A niobium nitride-based analog to digital converter using rapid single flux quantum logic operating at 9.5 K," *IEEE Transactions on Applied Superconductivity*, Vol. 4, no. 2, pp. 92-96, June 1994.

List of Publications

Refereed Journals

1. **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 & Systems*, pp. 217-229, vol. 27, no. 2, Feb 2008.
2. **R. Mahesh** and A. P. Vinod, "Reconfigurable frequency response masking filters for software radio channelization," *IEEE Transactions on Circuits and Systems-II*, vol. 55, no. 3, pp. 274-278, March 2008.
3. Jimson Mathew, **R. Mahesh**, A. P. Vinod and E. M-K. Lai, "Realization of low power high speed channel filters with stringent adjacent channel attenuation specifications for software radio receivers," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E91-A, no.9, pp.2564-2570, Sep 2008.
4. **R. Mahesh**, A. P. Vinod, E. M-K. Lai and Amos Omondi, "Filter bank channelizers for multi-standard software defined radio receivers," *Journal of Signal Processing Systems for Signal, Image and Video Technology*, Springer (Accepted in Jan 2009)

International Conferences

1. **R. Mahesh** and A. P. Vinod, "A new common subexpression elimination algorithm for implementing low complexity FIR filters in software defined radio receivers," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 4515 – 4518, May 2006, Greece. (ISCAS 2006).
2. **R. Mahesh** and A. P. Vinod, "Reconfigurable low complexity FIR filters for software radio receivers," in *Proceedings of 17th IEEE International Symposium on Personal, Indoor and Mobile radio communications*, pp. 1-5, Sep 2006, Finland. (PIMRC 2006).
3. **R. Mahesh** and A. P. Vinod, "Frequency response masking based reconfigurable channel filters for software radio receivers," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 2518-2521, May 2007, USA. (ISCAS 2007).
4. **R. Mahesh** and A. P. Vinod, "An architecture for integrating low complexity and reconfigurability for channel filters in software defined radio receivers," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 2514-2517, May 2007, USA. (ISCAS 2007).
5. **R. Mahesh** and A. P. Vinod, "A new low complexity reconfigurable filter bank architecture for software radio receivers based on interpolation and masking technique," in *Proceedings of Sixth IEEE International Conference on Information, Communications and Signal Processing*, pp. 1-5, Dec 2007, Singapore. (ICICS 2007).

6. Jimson Mathew, **R. Mahesh**, A. P. Vinod and E. M-K. Lai, "Realization of low power high speed channel filters with stringent adjacent channel attenuation specifications for software radio receivers," in *Proceedings of Sixth IEEE International Conference on Information, Communications and Signal Processing*, pp. 1-5, Dec 2007, Singapore. (ICICS 2007).
7. **R. Mahesh** and A. P. Vinod, "Coefficient decimation approach for realizing reconfigurable finite impulse response filters," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 81-84, May 2008, Seattle, USA. (ISCAS 2008).
8. Smitha K. G., **R. Mahesh** and A. P. Vinod, "A reconfigurable multi-stage frequency response masking filter bank architecture for software defined radio receivers," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 85-88, May 2008, Seattle, USA. (ISCAS 2008).
9. **R. Mahesh**, A. P. Vinod, Christophe Moy and Jacques Palicot, "A low complexity reconfigurable filter bank architecture for spectrum sensing in cognitive radios," in *Proceedings of 3rd International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, pp. 1-6, Singapore, May 2008. (CROWNCOM 2008).
10. **R. Mahesh**, A. P. Vinod, B. Y. Tan and E. M-K. Lai, "A tree-structured non-uniform filter bank for multi-standard wireless receivers," in *Proceedings of IEEE International Symposium on Circuits and Systems*, Taiwan, May 2009. (ISCAS 2009).

Papers Under Review

1. **R. Mahesh** and A. P. Vinod, "New reconfigurable architectures for implementing FIR filters with low complexity," *IEEE Transactions on Computer-Aided Design of Integrated Circuits & Systems*, revised and submitted, Oct 2008.
2. **R. Mahesh** and A. P. Vinod, "Low-Complexity Reconfigurable Filter Bank for Extracting Fractional Bandwidth Channels in Software Radio Channelizers," *IEEE Transactions on Circuits and Systems-I*, submitted, Nov 2008.
3. **R. Mahesh** and A. P. Vinod, "A Reconfigurable Low Complexity Filter Bank Architecture for Non-Uniform Channelization in Software Radio Receivers," *IEEE Transactions on Aerospace and Electronic Systems*, submitted, Sep 2008.
4. **R. Mahesh** and A. P. Vinod, "Coefficient Decimation Based Reconfigurable Filter Banks for Software Radio Receivers: A Low Complexity Alternative to DFT Filter Banks," *IEEE Transactions on Signal Processing*, submitted, Oct 2008.