1

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file - Votre référence

Our file - Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Design and VLSI Implementation of CMOS Decimation and Interpolation Half-band FIR Digital Filters

Steven Fatine, M.Sc., B.Eng.

**A thesis submitted to the Faculty of Graduate
Studies and Research in partial fulfillment
of the requirements of the degree of Master of Engineering**

**Ottawa-Carleton Institute of Electrical Engineering
Department of Electronics
Faculty of Engineering
Carleton University
Ottawa, Canada
January, 1996**

© Copyright 1996

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file *Votre référence*

Our file *Notre référence*

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-612-08975-4

Canada

Name Steven T. Feltz

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Psychological and Physiological Bases of Learning

SUBJECT TERM

0000

SUBJECT CODE

U·M·I

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture	0729
Art History	0377
Cinema	0900
Dance	0378
Fine Arts	0357
Information Science	0723
Journalism	0391
Library Science	0399
Mass Communications	0708
Music	0413
Speech Communication	0459
Theater	0465

EDUCATION

General	0515
Administration	0514
Adult and Continuing	0516
Agricultural	0517
Art	0273
Bilingual and Multicultural	0282
Business	0688
Community College	0275
Curriculum and Instruction	0727
Early Childhood	0518
Elementary	0524
Finance	0277
Guidance and Counseling	0519
Health	0680
Higher	0745
History of	0520
Home Economics	0278
Industrial	0521
Language and Literature	0279
Mathematics	0280
Music	0522
Philosophy of	0598
Physical	0523

Psychology	0525
Reading	0535
Religious	0527
Sciences	0714
Secondary	0533
Social Sciences	0534
Sociology of	0340
Special	0529
Teacher Training	0530
Technology	0710
Tests and Measurements	0288
Vocational	0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language	
General	0679
Ancient	0289
Linguistics	0290
Modern	0291
Literature	
General	0401
Classical	0294
Comparative	0295
Medieval	0297
Modern	0298
African	0316
American	0591
Asian	0305
Canadian (English)	0352
Canadian (French)	0355
English	0593
Germanic	0311
Latin American	0312
Middle Eastern	0315
Romance	0313
Slavic and East European	0314

PHILOSOPHY, RELIGION AND

THEOLOGY

Philosophy	0422
Religion	
General	0318
Biblical Studies	0321
Clergy	0319
History of	0320
Philosophy of	0322
Theology	0469

SOCIAL SCIENCES

American Studies	0323
Anthropology	
Archaeology	0324
Cultural	0326
Physical	0327
Business Administration	
General	0310
Accounting	0272
Banking	0770
Management	0454
Marketing	0338
Canadian Studies	0385
Economics	
General	0501
Agricultural	0503
Commerce-Business	0505
Finance	0508
History	0509
Labor	0510
Theory	0511
Fallure	0358
Geography	0366
Gerontology	0351
History	
General	0578

Ancient	0579
Medieval	0581
Modern	0582
Black	0328
African	0331
Asia, Australia and Oceania	0332
Canadian	0334
European	0335
Latin American	0336
Middle Eastern	0333
United States	0337
History of Science	0585
Law	0398
Political Science	
General	0615
International Law and Relations	0616
Public Administration	0617
Recreation	0814
Social Work	0452
Sociology	
General	0626
Criminology and Penology	0627
Demography	0938
Ethnic and Racial Studies	0631
Individual and Family Studies	0628
Industrial and Labor Relations	0629
Public and Social Welfare	0630
Social Structure and Development	0700
Theory and Methods	0344
Transportation	0709
Urban and Regional Planning	0999
Women's Studies	0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture	
General	0473
Agronomy	0285
Animal Culture and Nutrition	0475
Animal Pathology	0476
Food Science and Technology	0359
Forestry and Wildlife	0478
Plant Culture	0479
Plant Pathology	0480
Plant Physiology	0817
Range Management	0777
Wood Technology	0746
Biology	
General	0306
Anatomy	0287
Biostatistics	0308
Botany	0309
Cell	0379
Ecology	0329
Entomology	0353
Genetics	0369
Immunology	0793
Microbiology	0410
Molecular	0307
Neuroscience	0317
Oceanography	0416
Physiology	0433
Radiation	0821
Veterinary Science	0778
Zoology	0472
Biophysics	
General	0786
Medical	0760

EARTH SCIENCES

Biogeochemistry	0425
Geochemistry	0996

Geodesy	0370
Geology	0372
Geophysics	0373
Hydrology	0388
Mineralogy	0411
Paleobotany	0345
Paleocology	0426
Paleontology	0418
Paleozoology	0985
Polymology	0427
Physical Geography	0368
Physical Oceanography	0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences	0768
Health Sciences	
General	0566
Audiology	0300
Chemotherapy	0992
Dentistry	0567
Education	0350
Hospital Management	0769
Human Development	0758
Immunology	0982
Medicine and Surgery	0564
Mental Health	0347
Nursing	0569
Nutrition	0570
Obstetrics and Gynecology	0380
Occupational Health and Therapy	0354
Ophthalmology	0381
Pathology	0571
Pharmacology	0419
Pharmacy	0572
Physical Therapy	0382
Public Health	0573
Radiology	0574
Recreation	0575

Speech Pathology	0460
Toxicology	0383
Home Economics	0386

PHYSICAL SCIENCES

Pure Sciences	
Chemistry	
General	0485
Agricultural	0749
Analytical	0486
Biochemistry	0487
Inorganic	0488
Nuclear	0738
Organic	0490
Pharmaceutical	0491
Physical	0494
Polymer	0495
Radiation	0754
Mathematics	0405
Physics	
General	0605
Acoustics	986
Astronomy and Astrophysics	0606
Atmospheric Science	0608
Atomic	0748
Electronics and Electricity	0607
Elementary Particles and High Energy	0798
Fluid and Plasma	0759
Molecular	0609
Nuclear	0610
Optics	0752
Radiation	0756
Solid State	0611
Statistics	0463
Applied Sciences	
Applied Mechanics	0346
Computer Science	0984

Engineering	
General	927
Aerospace	0538
Agricultural	0539
Automotive	0540
Biomedical	0541
Chemical	0542
Civil	0543
Electronics and Electrical	0544
Heat and Thermodynamics	0348
Hydraulic	0545
Industrial	0546
Marine	0547
Materials Science	0794
Mechanical	0548
Metallurgy	0743
Mining	0551
Nuclear	0552
Packaging	0549
Petroleum	0765
Sanitary and Municipal	0554
System Science	0790
Geotechnology	0428
Operations Research	0796
Plastics Technology	0795
Textile Technology	0994

PSYCHOLOGY

General	0621
Behavioral	0384
Clinical	0622
Developmental	0620
Experimental	0623
Industrial	0624
Personality	0625
Physiological	0989
Psychobiology	0349
Psychometrics	0632
Social	0451

The undersigned hereby recommend to
The Faculty of Graduate Studies and Research
acceptance of the thesis,

**"DESIGN AND VLSI IMPLEMENTATION OF
CMOS DECIMATION AND INTERPOLATION
HALF-BAND FIR DIGITAL FILTERS"**

submitted by
Steven Fatine, M.Sc., B.Eng.

in partial fulfillment of the requirements
for the degree of Master of Engineering.

Chair, Department of Electronics

Thesis Supervisor

Thesis Supervisor

Thesis Supervisor

Carleton University
January, 1996

The information used in this thesis comes in part from the research program of Dr. Tadeusz A. Kwasniewski and his associates in the VLSI in Communications group. The research results appearing in this thesis represent an integral part of the ongoing research program. All research results in this thesis including tables, graphs and figures but excluding the narrative portions of the thesis are effectively incorporated into the research program and can be used by Dr. Kwasniewski and his associates for education and research purposes, including publication in the open literature with appropriate credits. Matters of intellectual property may be pursued collaboratively with Carleton University and Dr. Kwasniewski where and as appropriate.

Abstract

The design and VLSI implementation of CMOS decimation and interpolation half-band filters, targeted for use as rate converter modules in the design of digital mobile radio transceivers, are presented in this thesis. Two designs have been proposed with performance and characteristics evaluated through simulation and functional verification. The first design is of a dedicated integer fixed coefficient 27-tap decimate and interpolate-by-two half-band FIR digital filter, the design method based on Parks-McClellan algorithm yields power-of-two coefficients, thus reducing the filter hardware complexity and power consumption. The second design is of a programmable 7-tap decimate and interpolate-by-two half-band FIR digital filter. The impulse response of the programmable filter is user-selectable and can be generated using various half-band filter design techniques. The choice of polyphase filter architecture made a single-chip design feasible for implementing both decimation and interpolation half-band filters. Simulation results based on MOSIS 1.2 μ m CMOS technology, have shown the filters throughput to be 50 Msamples/sec under typical operating conditions.

Acknowledgements

First, I would like to thank my supervisors, Dr. T. A. Kwasniewski and Dr. C. H. Chan for their guidance and support of this thesis.

I would also to thank Dr. V. Szwarc of Communication Research Centre for his thoughtful suggestions and constant support of my research work. I wish to thank L. Desormeaux for his help and advices.

I am grateful to the staff and the graduate students of the Department of Electronics for providing such a fertile environment to pursue academic achievements.

I would like to express my most sincere thanks to the Communication Research Centre and Carleton University for the financial support throughout the period of my Master's thesis.

Finally, I extend my love and thanks to my parents who have been my inspiration forever.

Table of Contents

List of Figures.....	vii
List of Tables	xi
List of Symbols and Acronyms	xii

Chapter 1 Introduction

1.1. Background Information	1
1.2. Thesis Motivations.....	3
1.3. Thesis Outline	5

Chapter 2 Half-Band Filter Design and Implementation Issues

2.1. Introduction.....	6
2.2. Characteristics of FIR Digital Filters.....	7
2.2.1. Linear Phase Filters.....	7
2.3. Characteristics of Half-Band FIR Filters	8
2.4. Design Methods of Half-Band FIR Filters.....	12
2.4.1. Fourier Series Method.....	12
2.4.2. Window Design.....	13
2.4.3. Parks-McClellan Method	14
2.5. Sampling Rate Conversion	15
2.5.1. Sampling Rate Reduction	15
2.5.2. Sampling Rate Increase.....	17
2.6. VLSI Design Issues of Half-Band Filters	19
2.6.1. Multirate Systems Based on Half-Band Filters	20
2.7. Summary	23

Chapter 3 VLSI Architectures of Digital Half-Band FIR Filters

3.1. Introduction.....	24
3.2. Traditional Half-Band FIR Digital Filter Architectures	25
3.3. Polyphase Filtering Concept.....	27

3.4. Commutator Models	33
3.5. Polyphase Half-Band Decimation Filter Structures.....	35
3.6. Polyphase Half-Band Interpolation Filter Structures.....	38
3.7. Summary	40
Chapter 4 Design and VLSI Implementation of a Decimation and Interpolation Half-Band FIR Filter	
4.1. Introduction.....	41
4.2. Design Method of a Digital Half-Band FIR Filter.....	43
4.2.1. Introduction	43
4.2.2. Design Method.....	44
4.3. Filter Functional Block diagram Overview	49
4.3.1. Filter Controller Unit	49
4.3.2. Datapath Controller (DPC)	50
4.3.3. Even and Odd Delay Lines	51
4.3.4. Multiplier-accumulator (MAC1 and MAC2).....	52
4.3.4.a. Introduction	52
4.3.4.b. Concept of Architecture.....	53
4.3.4.c. High-order Counter Design	55
4.3.4.d. Implementation Procedure	58
4.3.5. Interpolation and Mode Selection Multiplexers	59
4.3.5.a. Interpolation Multiplexer (MUX1)	59
4.3.5.b. Mode Selection Multiplexer (MUX2)	60
4.3.6. Vector Merge Adder (VMA).....	60
4.3.7. Output Data Flow Controller (ROUNDER)	60
4.4. Implementation Considerations	61
4.5. Summary	63
Chapter 5 Hardware Synthesis of a Programmable Decimation and Interpolation FIR Digital Filter	
5.1. Introduction.....	64
5.2. Generic Implementation Issues.....	65

5.2.1. Design Specification	65
5.2.2. Architectural Considerations	66
5.3. Programmable Filter Functional Block Overview	67
5.3.1. Filter Controller Circuit	69
5.3.2. Coefficient Loading Interface	72
5.3.3. Symmetry Adders	74
5.3.4. The 20-bit Adder	76
5.3.5 12x8-bit Array Multiplier	76
5.4. Summary	80
Chapter 6 Verification Procedures and Simulation Results	
6.1. Introduction	81
6.2. Design Tools and Technology	82
6.2.1. Introduction	82
6.2.2. Schematic Capture Tools	82
6.2.3. Simulations Tools	83
6.2.4. Hardware Synthesis and Optimization Tools	84
6.3. Design Verification	84
6.3.1. Introduction	84
6.3.2. Functional Simulation of the 27-Tap Decimate/Interpolate By Two Half-Band FIR Filter	84
6.3.3. Functional Simulation of the Programmable 7-Tap Decimate/Interpolate By Two Half-Band FIR Filter	85
6.4. Summary	88
Chapter 7 Conclusion	
Conclusion	92
Appendix Hardware Description of The Programmable Half-Band Filter	
A-1 Behavioral Description of a 7-Tap Half-Band Filter	97
A-2 RTL Level Description of the Control Circuit	98
A-3 RTL Level Description of the Programmable Interface	100
A-4 Top-Level Structural Description of the Programmable Filter	102
References	106

List of Figures

Figure 1.1 Digital radio receiver chip functional block diagram.....	2
Figure 1.2 A proposed architecture of a digital receiver chain.....	4
Figure 2.1 Impulse response of half-band filter with $N=35$	10
Figure 2.2 Transfer function of half-band with $N=35$	10
Figure 2.3 Half-band filter frequency response requirements	11
Figure 2.4 Downsampler	16
Figure 2.5 Sampling rate reduction by a factor of two ($M=2$).....	16
Figure 2.6 Spectrum obtained using downsampling.....	17
Figure 2.7 Upsampler	18
Figure 2.8 Sampling rate increase by a factor of two ($L=2$).....	18
Figure 2.9 Spectrum obtained by upsampling	19
Figure 2.10 Multistage half-band decimator filter implementation.....	20
Figure 3.1 Direct-form realization of half-band FIR filter	25
Figure 3.2 Transposed-form of half-band FIR filter	26
Figure 3.3 Direct-form realization of symmetric FIR filter	27
Figure 3.4 Multirate identities representation.....	27
Figure 3.5 Relationship between $h(n)$ and its k^{th} polyphase component $e_k(n)$	28
Figure 3.6 Filter with embedded decimation showing redundancy among samples.....	30
Figure 3.7 Decimation filter after grouping the filter taps into polyphase subfilters.....	31
Figure 3.8 (a) The impulse response of the overall filter, (b),(c), and (d) impulse response of the polyphase subfilters.....	32
Figure 3.9 Ideal frequency response of polyphase network filters	32
Figure 3.10 Counterclockwise commutator model for decimate-by-three filter	33
Figure 3.11 Polyphase implementation of decimate-by-three filter using the commutator model.....	34
Figure 3.12 Clockwise commutator model for interpolate-by-three filter.....	34
Figure 3.13 Anti-aliasing half-band filter with M -to-1 decimator.....	35
Figure 3.14 Parallel form realization of half-band M -to-1 decimation filter.....	34
Figure 3.15 M -to-1 half-band polyphase filter structure	36
Figure 3.16 Half-band polyphase decimate-by-two filter structure	37
Figure 3.17 Decimation half-band filter with polyphase components at lower data rate	37
Figure 3.18 1-to- L interpolator with half-band filter anti-imaging filter	38

Figure 3.19 1-to-L half-band filter interpolation polyphase filter structure.....	38
Figure 3.20 Interpolation half-band filter with polyphase components at lower data rate.....	39
Figure 4.1 Polyphase filter block diagram.....	42
Figure 4.2 A single filter implementation for decimation and interpolation.....	42
Figure 4.3 Block diagram of a 27-tap decimate/interpolate-by-two half-band FIR filter.....	43
Figure 4.4(a) Transfer function of a 27-tap half-band FIR filter.....	46
Figure 4.4(b) Stopband detail of 27-tap half-band FIR filter.....	46
Figure 4.4(c) Passband detail of 27-tap half-band FIR filter.....	46
Figure 4.5 Frequency response deviation from an equiripple design.....	47
Figure 4.6 Frequency response of the polyphase half-band filters (a) $ H_0(e^{j\omega}) $, and (b) $ H_1(e^{j\omega}) $	48
Figure 4.7 Phase response of the polyphase half-band filters (a) $ H_0(e^{j\omega}) $, and (b) $ H_1(e^{j\omega}) $	48
Figure 4.8 Decimation mode control.....	50
Figure 4.9 Interpolation mode control.....	50
Figure 4.10 DPC operation in decimate-by-two mode.....	51
Figure 4.11 DPC in interpolate-by-two mode.....	51
Figure 4.12 Bit pattern of generated partial-products.....	54
Figure 4.13 Gate level schematic of a full-adder.....	54
Figure 4.14 Overall architecture of the 27-taps half-band filter multiplier-accumulator MAC1.....	56
Figure 4.15 Constructing a (5,3) counter from (3,2) counters.....	57
Figure 4.16 Constructing a (9,5) counter from (3,2) counters.....	57
Figure 4.17 Slice of MAC1 array where counters are used for partial-products reduction.....	59
Figure 4.18 Operation of interpolation multiplexer MUX1.....	60
Figure 4.19 Block diagram of multiplier-accumulator structure with pipeline registers.....	62
Figure 5.1 Simulation results of VHDL behavioral model of half-band FIR filter.....	66
Figure 5.2 Block diagram of a programmable decimate/interpolate-by-two half-band FIR filter.....	67
Figure 5.3 The internal VLSI architecture of the odd taps (a) and even taps (b) subfilters.....	68
Figure 5.4 Gate level schematic of the programmable filter control circuit.....	70
Figure 5.5 Control signals timing diagram in decimate-by-two mode.....	71
Figure 5.6 Control signals timing diagram in interpolate-by-two mode.....	71
Figure 5.7 Gate level schematic of the programmable coefficients loading interface.....	73
Figure 5.8 Timing diagram for the programmable half-band filter coefficients loading operation.....	73
Figure 5.9 Add with generate and propagate outputs.....	75

Figure 5.10 Four-bit carry-lookahead logic	75
Figure 5.11 12-bit adder using hierarchical carry-lookahead logic	75
Figure 5.12 Gate level schematic of Booth decode cell	78
Figure 5.11 Gate-level schematic of Booth multiplexer	79
Figure 5.12 Module structure in a 12x8-bit Booth array multiplier	79
Figure 6.1 ASIC design flow for hardware synthesis using Mentor Graphics tools	83
Figure 6.2 Power spectra of the 27-tap decimation half-band filter	86
Figure 6.3 Power spectra of the 27-tap interpolation half-band filter	87
Figure 6.4 Timing simulations of the programmable 7-tap half-band filter in (a) decimate-by-two and (b) interpolate-by-two modes	89
Figure 6.5 Power spectra of decimate-by-two programmable half band filter	90
Figure 6.6 Power spectra of interpolate-by-two programmable half-band filter	91

List of Tables

Table 4.1	The coefficients of a 27-tap half-band FIR filter.....	45
Table 5.1	The control signal states in decimate and interpolate by two mode.....	71
Table 5.2	Multiplier bit-pair recoding in Booth algorithm.....	77

List of Symbols and Acronyms

Inverter
2:1 Multiplexer
Exclusive OR gate, XOR
D-type flip-flop
OR gate
AND gate
0-volt reference, Gnd
$H(z), H_k(z)$	Transfer function of a digital FIR filter
$h(n)$	Impulse response of an FIR filter
ω	Signal frequency (rad/sec)
z^{-1}	Delay or storage element
f_p, f_s, F	Passband, stopband and sampling frequencies (Hz)
W_p, W_s	Passband and stopband weighting factors
$x(n)$	Digital signal sample at time n
$w(n)$	Window function
δ_p, δ_s	Passband and stopband ripple amplitudes (dB)
$X(e^{j\omega})$	Spectrum of a digital signal $x(n)$
A/D	Analog-to-Digital Converter

ASIC	Application Specific Integrated Circuit
BPF	Bandpass Filter
CAD	Computer Aided Design
CMOS	Complementary Metal Oxide Semiconductor
D/A	Digital-to-Analog Converter
DFT	Discrete Fourier Transform
DSP	Digital Signal Processing
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
IC	Integrated Circuit
IF	Intermediate Frequency
IIR	Infinite Impulse Response
LPF	Lowpass Filter
PLD	Programmable Logic Devices
RF	Radio Frequency
VHDL	VHSIC Hardware Description Language.
VHSIC	Very High Speed Integrated Circuits.
VLSI	Very Large Scale of Integration.

Chapter 1

Introduction

1.1. Background Information

Progress in digital mobile radio technology has been accomplished by the steady reduction in cost, terminal size, and power consumption. With the advent of digital radio and high-speed digitizers, the benefits of modern digital signal processing techniques are applied to the processing of IF and baseband signals. A representative block diagram of a digital radio receiver is shown in Figure 1.1.

There are many benefits to increasing the portion of radio that is implemented digitally. One of the biggest advantages of a digital receiver implementation is the potential for programming the hardware to accommodate different data rates, modulation formats, and filter specifications using a single hardware platform [1].

Figure 1.1 Digital radio receiver chip functional block diagram

Digital radio transceiver chips perform down and up conversion, digital filtering, decimation and interpolation of sampled RF signals. Several methods are available to perform digital filtering with sampling rate alteration including a technique that involves infinite impulse response (IIR) filtering [1], and another that uses finite impulse response (FIR) filtering [2]. FIR filters compared to their IIR equivalent, provide several advantages such as guaranteed stability, absence of limit cycles and exact linear phase. Half-band filters [3, 4] are symmetric linear phase FIR filters with nearly half the coefficients equal to zero. These filter characteristics have enabled the implementation of efficient half-band filter hardware. Goodman [5], Crochiere and Rabiner [6] have shown that half-band filters can be implemented in terms of simple coefficients and modest coefficient storage requirements leading to even more efficient designs. Vaidyanathan [7] and Fliege [8] have shown half-band filters to be effective building blocks of multirate DSP systems and filter banks.

1.2. Thesis Motivations

The Communication Research Centre (CRC) in Ottawa along with several collaborating companies and universities, is involved in VLSI implementation and development of an all-digital narrowband radio receiver. Figure 1.2 shows a proposed architecture of a radio receiver chain [1]. After an RF signal is extracted, further decimation is performed until the sampling rate is suitable for processing by a microprocessor. The implementation of the final decimate-by-N stage of the radio receiver can be accomplished by means of a cascade of decimate-by-two half-band filters. Similar concepts can be applied to the implementation of the radio transmit chain, where a cascade of interpolate-by-two half-band filters is used to implement the interpolate-by-N stage. The goal of this research project is the design and implementation of a dedicated 27-tap decimate/interpolate-by-two half-band FIR filter to be used as a building block of the decimation and interpolation chain. To broaden even further the scope of application of half-band filters, the project has also as objective the development of a family of generic and programmable decimation and interpolation half-band FIR filters. By using VHDL as a design tool, generic and programmable half-band decimation and interpolation filters have been designed, simulated, and synthesized. High speed and low power integrated circuit design methodology were addressed as part of the research. Architectural issues pertaining to the design of cascadable and reconfigurable filters with minimum computational requirements were also studied [1, 9, 10].

As a part of a large DSP system, the design of the half-band decimation and interpolation by two filter is based on a number of architectural issues which reflect both technology and certain performance and system level requirements. These constraints lead to the selection made for specific VLSI implementations of arithmetic blocks (adder, multipliers), the input and coefficient word size, and number of filter taps.

Figure 1.2 A proposed architecture of a digital radio receiver chain

1.3. Thesis outline

This thesis is organized in the following manner. Chapter 2 provides an overview of half-band FIR filter characteristics and their use in the implementation of multirate processing DSP systems. Chapter 3 reviews VLSI architectures of digital FIR filters, and provides the background for understanding the polyphase filtering concept. Chapter 3 is devoted to the use of polyphase filters in the design of multirate digital filters and filter banks, as well as their support of single-chip design of decimation and interpolation filters.

In Chapter 4, the design of a 27-tap decimate and interpolate by two half-band FIR filter is presented. The design method and a detailed functional block description are also included.

In Chapter 5, a top-down ASIC design approach with hardware synthesis and logic optimization using VHDL is applied to design a generic and programmable decimating and interpolating half-band FIR filter. Chapter 5 gives a detailed functional block description of the filter and also addresses the architectural considerations in programmable DSP designs. Design tools, functional simulation and performance results are presented in Chapter 6. Chapter 7 summarizes the thesis and provides recommendations for further research in the area.

Chapter 2

Half-Band Filter Design and Implementation Issues

2.1. Introduction

Digital signal processing greatly simplifies the design of radio transceivers, where programmable digital half-band filters can be used to compensate the passband magnitude and phase responses of the analog front end filter, allowing a lower cost filter to be used. Half-band filters are a class of symmetrical FIR filters, with the property that approximately half of the filter coefficients are zero. Hence the number of multiplications in implementing such filters is reduced to half of that needed for a typical symmetric FIR design and to one-fourth of that of an arbitrary FIR filter design. Half-band filters are inherently suitable for applications where the decimation and interpolation sampling rate changes are equal to two. However, a cascade of such filters can be used to achieve higher sampling rate ratios [2, 11]. For large changes in the sampling rate, it has been recognized that it is more efficient to reduce/increase the sampling rate with a series of smaller decimation/interpolation stages rather than to make the entire rate change with one stage [6, 11, 12].

2.2. Characteristics of FIR Digital Filters

Digital filters can be partitioned into two distinct classes: FIR and IIR filters. FIR filters are finite-length sum-of-products digital filters. Each output sample is a weighted sum of the new input value and $L-1$ previous inputs, where L is the order of the filter. The main advantages of FIR digital filters compared to their IIR counterparts are as follows.

- FIR filters can be designed with exactly linear phase.
- FIR filters are inherently stable, and have a simple architecture.
- Quantization noise due to finite precision arithmetic, and coefficient accuracy problems can be made less severe in FIR filter implementations.
- Multirate DSP systems can be efficiently implemented using FIR filters.

2.2.1. Linear Phase Filters

In many communication and image processing applications it is essential to have filters whose transfer function exhibits a phase characteristic which changes linearly with a change in frequency [7]. A very useful feature of FIR filters is that for a simple relationship of the coefficients, $h(n)$, the resulting filter is guaranteed to have linear phase response. The derivation of this relationship is illustrated as follows. The transfer function of a FIR causal filter of length N is given by

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n} \quad (2.1)$$

A linear phase relationship to the frequency means that

$$H(e^{j\omega}) = |H(e^{j\omega})| e^{j\theta(\omega)} \quad (2.2)$$

The phase response is defined as

$$\theta(\omega) = \arctan \frac{-\operatorname{Im}[H(e^{j\omega})]}{\operatorname{Re}[H(e^{j\omega})]}, \quad -\pi \leq \omega \leq \pi \quad (2.3)$$

Therefore for the phase response to be linear, it is required that

$$\theta(\omega) = -\tau\omega = \arctan \frac{-\sum_{n=0}^{N-1} h(n) \sin n\omega}{\sum_{n=0}^{N-1} h(n) \cos n\omega} \quad (2.4)$$

where τ is a constant phase delay. It can be shown that $\tau = (N-1)/2$ is a solution to Equation 2.4, and hence the FIR filter coefficients $h(n)$ satisfy the following property:

$$h(n) = h(N-1-n), \quad 0 \leq n \leq N-1. \quad (2.5)$$

Therefore, FIR filters will have constant phase if the condition of Equation 2.5 is satisfied. From this symmetry property in the impulse response $h(n)$, efficient FIR filter realization can be achieved as will be shown later in the chapter.

2.3. Characteristics of Half-Band FIR Filters

Half-band FIR filters are symmetric, linear-phase filters of N -odd valued order (i.e., $N-1$ is even) [2, 7, 12], where the impulse response $h(n)$ is assumed to be hermitian-symmetric with respect to $n = 0$ (i.e., $h(n) = h^*(-n)$). Thus, half-band filters are by definition linear-phase filters [14] whose impulse response $h(n)$ satisfies

$$\begin{aligned} h(k) &= h(N+1-k), & 1 \leq k \leq \frac{N-1}{2} \\ h(2k) &= 0, & k \neq \frac{N+1}{4} \end{aligned} \quad (2.6)$$

where $k = 1, 2, \dots, N$, and N is the filter length. In other words, the even-numbered samples of $h(n)$, with the exception of the center tap, are equal to zero. A typical plot of a half-band filter impulse response $h(n)$ is shown in Figure 2.1. As noted, $h(n)$ is symmetric about the time origin and crosses the zero axis at times which are non zero multiples of $2T$, where T is the sampling period. If $H(z)$ denotes the transfer function of a half-band filter of order N , therefore, the frequency response is of the form [8]:

$$H(e^{j\omega}) = e^{\frac{-j\omega(N-1)}{2}} |H(e^{j\omega})| \quad (2.7)$$

where $|H(e^{j\omega})|$ represents the real-valued amplitude response. Figure 2.2 shows the frequency response of a half-band filter where $N=35$. In general, the frequency response has the following properties [2, 8, 15]:

- Equal amplitude ripples in the passband and stopband, i.e.,

$$\delta = \delta_p = \delta_s \quad (2.8)$$

- The stopband frequency is given by

$$f_s = \frac{F}{2} - f_p \quad (2.9)$$

- The transition band is determined by

$$\Delta F = \frac{F}{2} - 2f_p \quad (2.10)$$

where F is the sampling frequency. Figure 2.3 illustrates the frequency response characteristics required for a half-band FIR filter.

Figure 2.1 Impulse response of half-band filter with $N=35$

Figure 2.2 Transfer function of half-band filter with $N=35$

Figure 2.3 Half-band filter frequency response requirements

Half-band filters of order N , given by $N = 4n+3$, for $n = 0, 1, 2, 3, \dots$, result in the following general $h(n)$ sequence [15]:

$$h(n) = \{x, 0, x, 0, \dots, x, x, x, \dots, 0, x, 0, x\} \quad (2.11)$$

where x denotes an actual non zero coefficient. Note that in this case the end coefficients are nonzero, and for every zero coefficient a multiplication is eliminated. The number of nonzero coefficients is then $M = (N+5)/4$. Half-band filters of order N , given by $N = 4n+5$ for $n = 0, 1, 2, 3, \dots$ result in the following general $h(n)$ sequence:

$$h(n) = \{0, x, 0, x, \dots, 0, x, x, x, 0, \dots, x, 0, x, 0\} \quad (2.12)$$

Note that in this case the end coefficients are zero. The number of non zero coefficients is then $M = (N+3)/4$.

The efficiency of half-band filters derives from the fact that nearly half of the filter coefficients are zero, thus cutting down the implementation cost and improved performance [9]. The corresponding transfer function of the N -tap causal half-band filter is:

$$H(z) = h_0 z^{-2M+1} + \sum_{i=1}^M h_i (z^{-2M+2i} + z^{-2M-2i+2}) \quad (2.13)$$

where $M = (N+1)/4$. By taking advantage of the zero coefficients and the symmetry, it can be seen that a half-band filter of length N can compute each output with only $(N+5)/4$ multiplications and $(N+1)/2$ additions, and $(3N+5)$ memory elements, instead of $(N+5)/2$ multiplications and $(N+1)$ additions that are normally required for a symmetric FIR filter of the same length. This is nearly a two-fold reduction in the computational complexity.

2.4. Design Methods of Half-Band FIR Filters

The specifications of the magnitude response of a digital filter are usually given in terms of the passband, transition band, and stopband tolerances as depicted in Figure 2.3 for the lowpass case. Generally, the standard techniques used for the design of digital FIR filters can be applied to half-band filters. These design methods differ only in their criteria of optimality. Therefore, only the widely used half-band filter design methods will be summarized below.

2.4.1. Fourier Series Method

The frequency response of a digital filter is periodic with the period equal to the sampling frequency. From Fourier series analysis, the desired frequency response of FIR digital filter can be represented by

$$H(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_d(n) e^{-jn\omega} \quad (2.14)$$

where the Fourier coefficients $h_d(n)$ are the desired impulse response sequence of the filter, which can be determined by

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{j\omega}) e^{jn\omega} d\omega. \quad (2.15)$$

Note that there is an implementation problem in equation 2.14, because the transfer function represents a noncausal digital filter of infinite duration. This causality problem can be solved by multiplying the transfer function by a delay factor. This modification does not affect the amplitude response, however, the abrupt truncation of the Fourier series results in oscillations in the passband and stopband. These oscillations are due to slow convergence of the Fourier series, particularly near the points of discontinuity. This effect is known as the Gibbs phenomenon [7, 15].

2.4.2. Window Design

The previous section showed that truncating the Fourier series results in FIR filters with undesirable oscillations. To reduce these effects, a particular class of functions is used to modify the Fourier coefficients (impulse response). These time-limited weighting functions are generally referred to as window functions. A simple approach to designing a lowpass FIR half-band filter with a cutoff frequency $\omega_c = \pi/2$, is to start by truncating the impulse response of an ideal lowpass filter. The effect of truncation can then be reduced by multiplying the desired impulse response coefficients $h_d(n)$ by an appropriate window function [15]:

$$h(n) = \frac{\sin((\pi n)/2)}{\pi n} \cdot \omega(n). \quad (2.16)$$

Possible window functions $\omega(n)$ are the Han window, Hamming window, Blackman window, Kaiser window, and rectangular window. In Equation 2.16, the term $\sin(\pi n/2)$ is equal to zero for nonzero values of n that are multiples of 2, which implies that the property of the impulse response $h(n)$ given in Equation 2.16 is indeed satisfied, justifying the

use of this technique in the design of half-band FIR filters.

2.4.3. Parks-McClellan Method

There is no known closed form solution for uniform approximation of FIR filters. The Parks-McClellan design has evolved as a standard numerical method which uses the Remez exchange algorithm [8] to iteratively calculate the coefficients of linear phase FIR filters. From Figure 2.3 the filter approximation problem can be expressed as

$$\begin{aligned} 1 - K\delta_s &\leq |H(e^{j\omega})| \leq 1 + K\delta_s, & 0 \leq f \leq f_p \\ -\delta_s &\leq |H(e^{j\omega})| \leq \delta_s, & f_s \leq f \leq \frac{F}{2} \\ \delta_p &= K\delta_s \end{aligned} \quad (2.17)$$

In the Parks-McClellan method, the number of coefficients N of the filter, the passband and stopband cutoff frequencies, the weighting factors W_p and W_s associated with each band have to be predefined. The parameters N , ω_p , ω_s are fixed and δ_s is allowed to vary. The iterative procedure is terminated when finding the frequencies at which the extrema occur in the passband and stopband. The impulse response $h(n)$, $n = 1, 2, \dots, N$ is determined as a result of the algorithm.

For half-band filters, the Parks-McClellan algorithm necessitates that the filter length N is suitably taken as $N = 4M - 1$ where $M = 1, 2, \dots$, the cutoff frequencies are chosen to be symmetric about $\omega = \pi/2$ and the weighting factors W_p and W_s are made equal. Computational complexity increases as the number of filter coefficients N increases [8]. Recently, a variation of Parks-McClellan design, was introduced, with the aim of reducing design time [16]. The method is based on the exploitation of the half-band filter's impulse response properties. Initially, the Parks-McClellan method is used to design a "one band" filter with an impulse response $g(n)$, with $(N+1)/2$ coefficients, a passband cutoff frequency $\omega_p' = 2\omega_p$, where ω_p is the pass-band of the desired half-band filter and a stopband

cutoff frequency $\omega_c = \pi$. The “one band” filter has no stopband, since $(N+1)/2$ is even. In the second step, the impulse response is up-sampled by a factor 2 by inserting zeros. Finally the center tap is set to 0.5. The resulting filter is a half-band filter with a transfer function $H(z)$

$$H(z) = \frac{1}{2} [G(z^2) + z^{-(N+1)/2}] \quad (2.18)$$

where $G(z)$ is the transfer function of the “one band” filter.

2.5. Sampling Rate Conversion

The most important characteristic common to all the filters and filter banks is the use of multirate processing of digital signals. A multirate system can be designed, when desired, to increase or decrease the sampling frequency of individual signals, either at the front end or the back end of the multirate system. To understand multirate systems, it is essential to first understand how the sampling rate is changed, and the effect on the signal spectrum. This topic is covered in the next section.

2.5.1. Sampling Rate Reduction

Reducing the sampling rate of a signal can be useful when the sampling rate is considerably greater than twice the bandwidth of the signal. It consists of two stages: the first contains an anti-aliasing filter, and the second contains a decimator which performs downsampling.

The sampling rate of a discrete signal $x(n)$ is reduced by a factor of M by taking only one of every M -th sample of the signal. The relationship between the resulting decimated signal $y_D(n)$ and the original signal $x(n)$ is as follows [7, 8]:

$$y_D(n) = x(n \cdot M) \quad (2.19)$$

where M and n are integers. Equation 2.19 shows that the output at time n is equal to the input at time nM . As a consequence, only the input samples $x(n)$ which occur at times equal to multiples of M are retained by the decimator. Figure 2.4 shows a signal flow representation of this process.

Figure 2.4 Downsampler

Figure 2.5 illustrates, the case of sampling rate reduction by a factor of two ($M = 2$).

Figure 2.5 Sampling rate reduction by a factor of two ($M=2$)

Since decimation corresponds to compression in time domain, the spectrum of the discretely sampled signal $Y_D(e^{j\omega})$ can be expressed using the modulation components as [7]

$$Y_D(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(e^{j\left(\omega - \frac{2\pi k}{M}\right)}\right) \quad (2.20)$$

where k is an integer. Figure 2.6 shows an example of the original signal spectrum, and the resulting output spectrum. The original signal is assumed to be band-limited to $\omega = \pi/M$.

Figure 2.6 Spectrum obtained using downsampling

Note that the individual spectra obtained by downsampling will overlap if the original signal is not band-limited to $\omega = \pi/M$. This overlapping, known as aliasing, causes a loss of information and changes the signal irreversibly. For many applications it is possible to decrease the bandwidth of a signal with a lowpass filter before downsampling it. This process of filtering and downsampling is referred to as decimation.

2.5.2. Sampling Rate Increase

The sampling rate needs to be increased if a narrow-band signal is to be observed with a finer resolution in the time domain. This is useful, for instance, for detecting zero-crossings of the signals. Increasing the sampling rate is called interpolation, and consists of upsampling followed by anti-imaging filtering.

The sampling rate of a discrete signal $x(n)$ is increased by a factor L by inserting $L-1$ equally spaced zeros between each pair of samples. The resulting interpolated signal $y_1(n)$ signal is then given by:

$$y_I(n) = \begin{cases} x(n/L) & \text{for } n = mL \\ 0 & \text{otherwise} \end{cases} \quad (2.21)$$

where L , n and m are integers. Figure 2.7 shows a signal flow representation of this process.

Figure 2.7 Upsampler

Figure 2.8 illustrates the case of sampling rate increase by a factor of 2. The spectrum of the output signal $Y_I(e^{j\omega})$ can be expressed as [7, 8]:

$$Y_I(e^{j\omega}) = X(e^{j\omega L}) \quad (2.22)$$

Figure 2.8 Sampling rate increase by a factor of two ($L=2$)

Figure 2.9 shows an example of the original signal spectrum, and the resulting output spectrum.

Figure 2.9 Spectrum obtained by upsampling

Note that $Y_L(e^{j\omega})$ is an L -fold compressed version of $X(e^{j\omega})$. The appearance of multiple copies of the original spectrum is called the imaging effect and the extra copies are the images created by the interpolator. Imaging effect, on the other hands does not lead to any loss of information. The process of upsampling and anti-image filtering is referred to as interpolation.

2.6. VLSI Design Issues of Half-Band Filters

With low complexity and phase linearity, a number of multirate applications benefit from the use of half-band FIR filters. In VLSI implementation of FIR filters, the symmetry is usually exploited to reduce the number of multiplications by half [7, 8]. VLSI designers prefer the very basic direct form filter structure or its transpose for their hardware design simplicity. This symmetry can be exploited easily only in the direct form or its transpose [16, 17].

The main advantage of half-band filters follows from the symmetry in their frequency response, thereby reducing the hardware complexity by approximately 50%. With such reduced complexity, these filters are extensively used for implementing multistage decimation and interpolation filters [13]. Widely used structures for their hardware implementation, are the direct form, transpose form, and polyphase representation. While poly-

phase filtering is an efficient approach to multirate digital signal processing, polyphase architecture suffers from difficulties in exploiting the symmetry [17].

The only cost-effective way to implement all-digital filters is through the use of custom integrated circuits which implement dedicated signal processing algorithms therefore significantly increasing the throughput and reducing the hardware complexity for most digital signal processing systems. For low-speed applications (e.g., voice-band modem), programmable DSP chips can implement digital filters very efficiently. A multiplier-accumulator is the ideal arithmetic element for implementing transversal filters at low-speeds. However, for high-speed applications (e.g., digital radio modem), a separate multiplier would probably be required for each tap of the filter which is generally not acceptable from the hardware complexity viewpoint [18]. For dedicated digital filtering applications, a custom VLSI design can be tailored.

2.6.1. Multirate Systems Based on Half-Band Filters

Multirate processing provides the method for changing the sampling rate and forms the basis for implementing complex signal processing applications with reduced computational requirements [2, 7, 8]. A cascaded half-band filter structure offers the following advantages [2]: significantly reduced number of computations, reduced memory requirements, simplified filter design problem, and reduced finite wordlength effects.

Let us consider now a cascade of K decimators used to achieve a sampling rate reduction by a factor D -to-1 as illustrated in Figure 2.10.

Figure 2.10 Multistage half-band decimator filter implementation

Due to the symmetry property, half-band filters are used for the design of multistage decimators and interpolators in which D is a power of 2. The number of computations and storage for stage k of a K multistage implementation is expressed by [15]:

$$\begin{aligned} MPS &= \frac{N(k) + 5}{4} \\ ADS &= \frac{N(k) + 1}{2} \\ MEM &= \frac{3N(k) + 5}{2} \end{aligned} \quad (2.23)$$

where MPS , ADS and MEM denote respectively the required number of multiplications, additions and memory elements. $N(k)$ is the filter order of stage k ($k = 1, 2, 3, \dots, K$). As can be noted, Equation 2.18 represents a reduction by approximately a factor of two from that of a conventional design. Therefore the number of multiplications per unit time (MPU) and the number of additions per unit time (APU) for a stage k of a K multistage are

$$\begin{aligned} MPU(k) &= F(k) \cdot \frac{N(k) + 5}{4} \\ APU(k) &= F(k) \cdot \frac{N(k) + 1}{2} \end{aligned} \quad (2.24)$$

where $F(k)$ is the sampling frequency of the half-band filter in stage k , with $F(k) = F(k-1)/D(k)$, ($D(k)$ is a decimation factor of stage k) and $F(0) = F$, F is the sampling frequency at the input. The frequency response requirements for stage k of a K -stage multistage half-band filter are defined by [2, 15]. The amplitudes of the passband and stopband ripple are $\delta_p(k)$ and $\delta_s(k)$ respectively. $f_p(k)$, $f_s(k)$ are the passband and stopband frequencies of the half-band filter in stage k . In stage 1, δ_p , δ_s are the passband and stopband ripple amplitudes. f_p , f_s are the passband and stopband frequencies of the half-band filter.

$$\begin{aligned}
 \delta_p(k) &= \delta_s(k) = \min\left(\frac{\delta_p}{K}, \delta_s\right), & k &= 1, 2, \dots, K \\
 f_p(k) &= f_p, \\
 f_s(k) &= \frac{F}{\prod_{i=1}^k D_i} - f_p(k) = F(k) - f_p.
 \end{aligned} \tag{2.25}$$

In the case where no aliasing is allowed in the final transition band, the frequency response is extended to meet the no aliasing requirement. Equation 2.26 defines the frequency constraints in such a case.

$$\begin{aligned}
 f_p(k) &= \begin{cases} \frac{F(K)}{2}, & k = 1, 2, \dots, K \\ f_p, & k = K \end{cases} \\
 f_s(k) &= \frac{F}{2^k} - f_p(k), & k &= 1, 2, \dots, K.
 \end{aligned} \tag{2.26}$$

Any power of two decimation/interpolation ratio can be obtained by cascading half-band filters in a multistage structure. In cases where changes in sampling rate are not a power of two, only a portion of the multistage filter can be implemented using half-band filters. The number of stages K required for D a power of two is given by [15]

$$K = \text{Log}_2(D). \tag{2.27}$$

The advantages of multirate processing, however, must be weighed against the additional complexity in the control portion of the hardware/software implementation required to handle the changes in the sampling rates and the selection of certain samples and the deletion of others.

2.7. Summary

The preceding review of digital FIR filters characteristics and design issues has shown that half-band FIR filters are linear phase with odd-order values, used as sampling rate converters in many DSP and communication applications. The frequency response of half-band filters exhibits a symmetry, which results in the even numbered coefficients being zero with the exception of the center tap. As a result, the realization of half-band digital FIR filters requires only half as many multipliers as standard FIR filters. This property leads to a simpler architecture with less hardware complexity. Half-band filters are used for decimating or interpolating the data rate with sampling rate changes of two, for such simple systems the reduction in computation by nearly a half is significant. However, half-band filters are even more important in multistage decimation (interpolation) by successive halving (doubling) of the sampling rate, where cascaded half-band filter stages are used to realize a large overall change in sampling rate. This design approach is widely used to design multirate processing systems and can lead to a substantial reduction in computations and memory requirements.

Chapter 3

VLSI Architectures of Digital Half-Band FIR Filters

3.1. Introduction

Multirate digital systems utilize high speed decimation and interpolation filters in order to reduce the sampling rate so that complex processing may be performed at a lower data rate. These sampling rate converters are generally represented by sampling switches and filters, which remove undesired aliasing when decimating and unwanted imaging when interpolating a given signal. Polyphase structures provide efficient implementations for the realization of multirate systems based on a bank of filters operating in parallel. These include real-time implementation of decimators and interpolators, fractional sampling rate changing devices, uniform DFT filter banks, and perfect-reconstruction analysis and synthesis systems [2, 7].

In the following sections, various half-band FIR filter VLSI architectures will be examined and compared in terms of performance for the purpose of selecting an efficient

hardware implementation.

3.2. Traditional Half-Band FIR Digital Filter Architectures

The most common VLSI structure for implementing time-invariant FIR half-band filters of length N is the direct-form structure as shown in Figure 3.1.

$$y(n) = \sum_{k=0}^{N-1} h(k) x(n-k) \quad (3.1)$$

It corresponds to a direct implementation of the convolution equation 3.1 [2, 7, 15] where $x(n)$ and $y(n)$ are the input and output samples at time n , and the N -point impulse response of the filter, $h(k)$, is assumed to be zero for $k < 0$ and $k > N - 1$.

Figure 3.1 Direct-form realization of half-band FIR filter

The computation of output samples $y(n)$ is obtained from the present and past $N-1$ samples of the input signal $x(n)$, respectively, weighted by the filter coefficients $h(k)$. By transposition of the direct-form FIR half-band filter structure, the transposed filter structure is obtained. Figure 3.2 shows the transposed filter structure derived from the direct-form implementation. In the transposed structure, the branch points in the shift registers become summing points in the transposed structure, thus these shift registers become accumulator shift registers.

Figure 3. 2 Transposed-form realization of half-band FIR filter

Half-band FIR filters are symmetric, the exploitation of the symmetry property is achieved by adding data samples prior to performing multiplication operations [2, 11]. In the case where N (filter length) is even, the application of symmetry leads to

$$y(n) = \sum_{k=0}^{\frac{N}{2}-1} h(k) [x(n-k) + x(n-N+1+k)] \quad (3.2)$$

Similarly, when N is odd, the application of symmetry leads to

$$y(n) = \sum_{k=0}^{\frac{N-3}{2}} h(k) [x(n-k) + x(n-N+k+1)] \quad (3.3)$$

$$+ h\left(\frac{N-1}{2}\right) x\left(n - \frac{N-1}{2}\right)$$

where n and k are integers. Figures 3.3 shows the FIR half-band filter symmetric structure implemented with only half the multipliers of that of a standard direct implementation.

Figure 3.3 Direct-form realization of symmetric FIR filter

3.3. Polyphase Filtering Concept

In Chapter 2, cascades of decimators and interpolators with linear time-invariant filters were introduced. A different type of cascade is shown in Figure 3.4.a, where a filter $H(z)$ follows a decimator, and in Figure 3.4.c where $H(z)$ precedes an interpolator. If the function $H(z)$ is rational (i.e. a ratio of polynomials in z and z^{-1}) Figure 3.4.a can be redrawn as in Figure 3.4.b and Figure 3.4.c as in Figure 3.4.d. These permutations are called multirate identities and are useful in the theory and implementation of multirate systems [7, 8]

Figure 3.4 Multirate identities representations

An important advancement in multirate signal processing is the invention of polyphase representation. This permits simplification of theoretical results and leads to computationally efficient implementations of decimation and interpolation half-band FIR filters, as well as of filter banks (both single and multirate). Suppose $H(z)$ to represent the transfer function of a linear-phase FIR filter, and M is a given integer. $H(z)$ can be written as [7, 22, 23]:

$$H(z) = \sum_{k=0}^{M-1} z^{-k} H_k(z^M) \quad (3.4)$$

where

$$H_k(z) = \sum_{n=-\infty}^{\infty} e_k(n) z^{-n} \quad (3.5)$$

each output sample of the new filter structure $H_k(z)$ is computed using $e_k(n)$ as a set of coefficients whose relationship with respect to the impulse response of the original filter $h(n)$ is expressed as

$$e_k(n) = h(Mn + k), \quad k = 0, 1, 2, \dots, M-1 \quad (3.6)$$

where n and k are integers. The factors z^{-k} represent the delays either preceding the decimators or following the interpolators, $H_k(z)$ are called polyphase filters, $h(n)$ and $e_k(n)$ are respectively the impulse response of the original filter and its k^{th} polyphase component. Figure 3.5 illustrates the generation of $e_k(n)$ from $h(n)$. Note that $H_k(z)$ depend entirely on the choice of M .

Figure 3.5 Relationship between $h(n)$ and its k^{th} polyphase component $e_k(n)$

Equation 3.4 shows that the original filter structure can be partitioned into M subfilters which operate at the low sampling rate, and each subfilter contributes one nonzero output which corresponds to one of the M outputs of the network. The individual polyphase filters $H_k(z^M)$ have a number of interesting properties. This is a consequence of the fact that the impulse response of the k^{th} subfilter is the decimated version of the impulse response of the original half-band filter shifted by k units.

To have a better understanding of polyphase filter properties, a specific example will be considered, namely an FIR symmetric filter with nine taps $\{h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8\}$, operating as a decimator with a decimation ratio of $M = 3$. As shown in Figure 3.6, there is a considerable amount of redundancy among the input sequences. For example, the input to h_0 is the sequence $\{x(n), x(n+3), x(n+6), \dots\}$, while the input to h_3 is $\{x(n-3), x(n), x(n+3), \dots\}$, which is simply a delayed version of the input to h_0 . Similarly, the input to h_6 is a delayed version of the input to h_0 . Corresponding conditions apply to h_1, h_4 , and h_7 , and likewise to h_2, h_5 and h_8 . These observations indicate that only $M-1=2$ delays are needed at the high rate end of the filter to consolidate the three groups of filter taps as shown in Figure 3.7. Note that with this structure, consisting of a chain of $M-1$ delays followed by M decimators, the top subfilter receives as its input $\{x(n), x(n+3), x(n+6), \dots\}$; the middle subfilter receives the sequence $\{x(n-1), x(n+2), x(n+5), \dots\}$; and the bottom subfilter receives $\{x(n-2), x(n+1), x(n+4), \dots\}$. Thus the delay/decimator chain has the effect of distributing consecutive samples among the three subfilters. Figure 3.8 shows the samples of the impulse response of the original filter $h(n)$ where it is assumed to be symmetric about $n = 4$. Note that $h(n)$ has a delay of 4 samples with respect to $n=0$. $e_0(n)$ has a delay of $4/3$ samples, $e_1(n)$ has a delay of one sample and $e_2(n)$ has a delay of $2/3$ samples. Thus different fractional sample delays and consequently different phase shifts are associated with every filter path. The fact that the phase difference in every filter path is the reason for the term polyphase representation.

Figure 3. 6 Filter with embedded decimation shows redundancy among samples

Figure 3. 7 Decimation filter after grouping the filter taps into polyphase subfilters

Figure 3.8 (a) The impulse response of the overall filter, (b), (c), and (d) impulse responses of the polyphase subfilters

Another important property of polyphase filters is that in case when the frequency response of the original filter $H(e^{j\omega})$ approximates an ideal lowpass filter. The frequency range $0 < \omega < \pi/M$ of $H(e^{j\omega})$ scales to the range $0 < \omega' < \pi$ for the filter $H_k(e^{j\omega'})$. Thus the polyphase filters approximate all-pass frequency functions, this is shown in Figure 3.9

Figure 3.9 Ideal frequency response of polyphase network filters

3.4. Commutator Models

Commutator models are conceptual tools that graphically depict the operation of polyphase filter implementations, mainly the data feeding mode at the input, and the data interlacing mode at the output [2, 7, 8]. Figure 3.10 demonstrates a commutator-model equivalent circuit, with a decimation ratio $M=3$ that is useful for polyphase decimation filters. The switch rotates at uniform speed in counterclockwise direction, taking one of the three indicated positions. The commutator takes three input samples of the signal $x(n)$ and distributes them alternatively to the polyphase branches. Figure 3.11 shows the implementation of such a model. It is clear that each subfilter has only to operate at one third of the input data rate. A second commutator model that is useful for polyphase interpolation filters is shown in Figure 3.12. This is called the clockwise model, as the switch rotates in a clockwise direction. For each input sample $x(n)$ the switch sweeps through the three polyphase branches to obtain three output samples of $y(n)$. Thus, every output sample is obtained by multiplexing between the three subfilters at three times the input sampling rate [2, 7].

Figure 3.10 Counterclockwise commutator model for decimate-by-three filter

Figure 3.11 Polyphase implementation of decimate-by-three filter using the commutator model

Figure 3.12 Clockwise commutator model for interpolate-by-three filter

3.5. Polyphase Half-Band Decimation Filter Structures

One of the simplest multirate systems is the M-to-1 decimator which reserves the first-out of every M consecutive samples. A decimation filter represented by the transfer function $H(z)$ can be then represented by the block diagram shown in Figure 3.13. In this system, only one of every M filter output samples will pass through the down-sampler. This fact should be used in the implementation of the filter $H(z)$. By applying Equation 3.4, $H(z)$ can be decomposed into M subfilters. Therefore, the M-to-1 decimation filter can be realized in a parallel form as shown in Figure 3.14 [7, 22, 23].

Figure 3.13 Anti-aliasing half-band filter with M-to-1 decimator

Figure 3.14 Parallel form realization of half-band M-to-1 decimation filter

By invoking multirate identities illustrated in Figure 3.4, $H_k(z)$ can be down-sampled and moved to the lower sampling rate side of the decimator without changing the input-output relationship as shown in Figure 3.15. The polyphase half-band filter array structure is then accomplished.

Figure 3.15 M-to-1 half-band decimation polyphase filter structure

In the case where $H(z)$ is an FIR filter, the arithmetic computational load is reduced by a factor of M , and a savings of a factor of M in the storage requirements for internal filter variables can be achieved. This is realized by implementing an M-to-1 decimation filter as in Figure 3.15 and each of the polyphase filters as a transposed structure. Then accumulator-shift register in each filter can be shared by summing the partial products of each respective tap of the polyphase filters with the respective taps of the other polyphase filters before storing them in the shift register [2].

In case of a 2-to-1 decimation process, M takes the value of two and $H(z)$ can be rewritten as

$$H(z) = H_0(z^2) + z^{-1}H_1(z^2). \quad (3.7)$$

The polyphase components $H_0(z)$ and $H_1(z)$ are expressed in the form of Equation 3.8 as

$$\begin{aligned} H_1(z) &= \sum_{n=-\infty}^{\infty} h(2n+1)z^{-n} \\ H_0(z) &= \sum_{n=-\infty}^{\infty} h(2n)z^{-n}. \end{aligned} \quad (3.8)$$

Note that the impulse response $h(n)$ of the original filter is now grouped into even-num-

bered samples $h(2n)$ and odd-numbered samples $h(2n+1)$. Figure 3.16 shows the polyphase implementation of the filter with a decimation ratio of two.

Figure 3.16 Half-band polyphase decimate-by-two filter structure

By invoking the cascades equivalence shown in Figure 3.4, the system in Figure 3.16 can be redrawn as shown in Figure 3.17.

Figure 3.17 Decimation half-band filter with polyphase components at lower data rate

If the N^{th} order FIR 2-to-1 decimation filter is implemented using the traditional direct form shown in Figure 3.1, only the even-numbered output samples $y(2n)$ are computed. Thus, the speed of operation should therefore correspond to $N+1$ multiplications and N additions per unit of time. However, during the odd instants of time, the structure is merely inactive, since the odd-numbered output samples $y(2n+1)$ are not computed. This results in inefficient hardware resource utilization [7].

Next, consider the polyphase implementation of Figure 3.17. The computation cost is also $N + 1$ multiplications and N adders. However, since $H_0(z)$ and $H_1(z)$ operate at the lower

rate, only a total of $(N + 1)/2$ multiplications per unit time, and $N/2$ additions per unit time are required. The multipliers and adders in each of the subfilter have two units of time available to compute each output, and they are continually operating, which means full exploitation of hardware resources [7].

3.6. Polyphase Half-Band Interpolation Filter Structures

Another simple multirate system is the 1-to- L interpolator which inserts $L - 1$ zero-valued samples between two consecutive input samples. The interpolator can also be characterized by the block diagram shown in Figure 3.18.

Figure 3.18 1-to- L interpolator with half-band anti-imaging filter

In this interpolator, only one of every L filter output samples can have a non-zero value because of the up-sampler. Similarly the transfer function $H(z)$ can be partitioned into L subfilters, and by moving the polyphase components to the lower data rate, the 1-to- L interpolation filter can be realized in parallel form [7, 8] as shown in Figure 3.19.

Figure 3.19 1-to- L half-band interpolation polyphase filter structure

By realizing each of the polyphase filters in Figure 3.19 as a direct-form FIR filter it can be observed that the same shift register for holding delayed samples can be shared with each of the polyphase filters. Therefore the total storage requirements for data storage, as well as the computation rate, is reduced by a factor of L since each of the polyphase filters is only $1/L$ as long as the original filter [2].

In case of a 1-to-2 interpolation process, L takes the value of two and $H(z)$ can be divided into two polyphase subfilters $H_0(z)$ and $H_1(z)$. Figure 3.20 shows such an implementation.

Figure 3.20 Interpolation half-band filter with polyphase components at lower data rate

If N^{th} order FIR 1-to-2 interpolation filter is implemented using a direct form, at most half of the input samples to $H(z)$ are nonzero, since zero samples are inserted at the input. This means that only half of the multipliers $h(n)$ have nonzero inputs. As a result, the remaining multipliers are not used, and the active multipliers must complete the computations in half-unit time. In Figure 3.20, $H_0(z)$ and $H_1(z)$ are operating at the input rate, and all the multipliers are operating. This results in an efficient hardware utilization. The computational complexity of the system is $(N + 1)$ multiplications per unit time and $(N - 1)$ additions per unit time [7, 8].

3.7. Summary

Since the polyphase filters are operating at the lower sampling data rate, the realization of an M -to-1 half-band decimator and an 1-to- L half-band interpolator using polyphase filters results in a reduction of the arithmetic computational load by a factor of M and L respectively. However, the delay units needed in the direct realization of the integral filter and the polyphase filter structure are the same. Polyphase architecture presents an efficient approach to digital filtering, and the main application of this structure lies in the realization of multirate DSP systems and uniform filter banks where the filter is split into a set of subfilters operating in parallel. The polyphase representation structure offers architectural advantages as compared to other conventional DSP architectures, allowing a single-chip realization for both decimation and interpolation processes, and the filtering operations are arranged so that the sampling rate is kept to the lowest value throughout the processing. This is achieved by lowering the internal clock speed thus reducing the power consumption, which makes it a suitable choice for low power digital signal processing and communication system design. The low sensitivity of this structure to coefficient wordlengths combined with the significant computation rate reduction can considerably simplify the hardware implementation of digital half-band filters.

Chapter 4

Design and VLSI Implementation of a Decimation and Interpolation Half-Band FIR Filter

4.1. Introduction

In polyphase architecture, a multirate filter system in which the input and output data sampling rates differ by a ratio of two, can be efficiently implemented by dividing the filter into two subfilters to allow processing to take place at the lower of the two data rates. As shown in Figure 4.1, these subfilters are connected to the external interfaces by input and output controllers which employ polyphase filters to produce the desired decimation and interpolation function. A dedicated 27-tap fixed coefficient, linear-phase VLSI half-band filter circuit, which allows both decimation and interpolation by two has been designed and implemented by applying polyphase filtering technique. By taking into consideration control and data flow issues, a single-chip solution is used to implement both the decimation and interpolation process as depicted in Figure 4.2,

Figure 4.1 Polyphase filter block diagram

Figure 4.2 A single filter implementation for decimation and interpolation

Figure 4.3 shows a functional block diagram of a 27-tap decimate and interpolate-by-two half-band FIR filter. A detailed circuit description will be discussed in detail in the next sections.

Figure 4.3 Block diagram of a 27-tap decimate/interpolate-by-two half-band FIR filter

4.2. Design Method of a Digital Half-Band FIR Filter

4.2.1. Introduction

High speed digital filtering applications (sample rates in excess of 10 MHz) require the use of custom application specific integrated circuits (ASIC's). Programmable signal processing cannot accommodate such high sample rates without an excessive amount of parallel processing which is not cost effective. For dedicated applications, the flexibility of a multiplier is not necessary, hence a far more efficient filter structure can be obtained by implementing each filter tap with dedicated logic blocks, leading to a fixed filter coefficients realization [13, 25, 26].

Multiplication by power-of-two coefficients can be realized in a dedicated hardware structure by simply shifting the data bus to the left or right by an appropriate number of bits, therefore a fairly broad set of filter coefficients can be realized by incorporating a small number of adders/subtractors in each filter tap along with the appropriate shifting operations. The resulting hardware is a small fraction of the hardware required to imple-

ment a general purpose multiplier and thus a significantly larger number of taps can be implemented on a single chip [26].

4.2.2. Design Method

The half-band filter coefficients have been generated using the Parks-McClellan algorithm [7, 8, 12]. For odd values of N of the form $N = 4n + 3$, where $n = 0, 1, 2, 3, \dots$, the Parks-McClellan method can be used to generate half-band filters coefficients. The resulting 27-tap half-band FIR filter exhibits a behavior in the frequency response which is different from that of an equiripple filter. There is an order of 15 dB deviation from an ideal half-band filter design

In order to reduce the computational cost in terms of hardware complexity and improve the filter performance, these coefficients are scaled and rounded to integer values. As a result, discrete power-of-two coefficients are obtained, and all the multiplication operations are implemented by simple shift and add operations, where every shift operation is hard-wired. Table 4.1 shows the filter coefficients resulting from the Parks-McClellan algorithm and the corresponding scaled and rounded values. With power-of-two coefficients, the filter complexity and power consumption are reduced and the throughput is increased as compared to a filter architecture using a multiplier for each tap [18, 27]. Figures 4.4a, b, and c show the filter characteristics, the passband exhibiting less than 0.15 dB of ripple from 0 to $0.2F_s$ with a stopband attenuation about 41dB from $0.3F_s$ to $0.5F_s$ (Nyquist frequency), with F_s being the sampling frequency. At $0.25F_s$ the filter provides 6 dB attenuation. The departure of the frequency characteristics of the half-band filter due to scaling and rounding of its coefficients from an ideal equiripple design is shown in Figure 4.5.

In case of sampling rate conversion by a ratio of two, equation 3.7 describes the filter transfer function as a function of its two polyphase components $H_0(z)$ and $H_1(z)$.

Table 4.1: The coefficients of the 27-tap half-band FIR filter

Coefficient No.	Decimal Value Parks-McClellan	Scaled and Rounded Value
h_1	0.0039	1
h_2	0.0000	0
h_3	-0.0079	-2
h_4	0.0000	0
h_5	0.0156	4
h_6	0.0000	0
h_7	-0.0282	-8
h_8	0.0000	0
h_9	0.0502	13
h_{10}	0.0000	0
h_{11}	-0.0976	-25
h_{12}	0.0000	0
h_{13}	0.3154	82
h_{14}	0.5000	130
h_{15}	0.3154	82
h_{16}	0.0000	0
h_{17}	-0.0976	-25
h_{18}	0.0000	0
h_{19}	0.0502	13
h_{20}	0.0000	0
h_{21}	-0.0282	-8
h_{22}	0.0000	0
h_{23}	0.0156	4
h_{24}	0.0000	0
h_{25}	-0.0079	-2
h_{26}	0.0000	0
h_{27}	0.0039	1

Figure 4.4.a Transfer function of a 27-tap half-band FIR filter

Figure 4.4.b Stopband detail of 27-tap half-band FIR filter

Figure 4.4.c Passband detail of 27-tap half-band FIR filter

Figure 4.5 Frequency response deviation from an equiripple design

Supposing $H(z)$ corresponds to the 27-tap half-band filter transfer function, it can be then expressed as

$$H(z) = 1 - 2z^{-2} + 4z^{-4} - 8z^{-6} + 13z^{-8} - 25z^{-10} + 82z^{-12} + 130z^{-13} \\ + 82z^{-14} - 25z^{-16} + 13z^{-18} - 8z^{-20} + 4z^{-22} - 2z^{-24} + z^{-26}. \quad (4.1)$$

Thus, the two polyphase components $H_0(z)$ and $H_1(z)$ are written as

$$H_0(z) = 1 - 2z^{-1} + 4z^{-2} - 8z^{-3} + 13z^{-4} - 25z^{-5} + 82z^{-6} + \\ + 82z^{-7} - 25z^{-8} + 13z^{-9} - 8z^{-10} + 4z^{-11} - 2z^{-12} + z^{-13}. \quad (4.2)$$

$$H_1(z) = 130z^{-6}.$$

Figure 4.6 and 4.7 show the magnitude response and the phase responses of the polyphase filters $H_0(z)$ and $H_1(z)$. The resulting polyphase subfilters exhibit a linear phase with respect to the frequency. As can be observed in Figure 4.6, the frequency response range in the band $0 < F/F_s < 0.5$ as shown in Figure 4.5 for the overall half-band FIR filter, scales to the frequency response range in the band $0 < F/F_s < 0.25$ for the polyphase half-band filters $H_0(z)$ and $H_1(z)$. Therefore, $|H_0(e^{j\omega})|$ and $|H_1(e^{j\omega})|$ are all-pass frequency functions.

Figure 4.6 Frequency response of the polyphase half-band filters (a) $|H_0(e^{j\omega})|$, and (b) $|H_1(e^{j\omega})|$

Figure 4.7 Phase response of the polyphase half-band filters (a) $|H_0(e^{j\omega})|$, and (b) $|H_1(e^{j\omega})|$

4.3. Filter Functional Block Diagram Overview

Figure 4.3 shows a functional block diagram of a multiplier-free 27-tap decimate and interpolate by two half-band FIR filter, with 12 bit 2's complement data input and up to 22 bit data output. Each of the filter functional blocks is described in the following sections.

4.3.1. Filter Controller Unit

This unit generates control signals in accordance with whether the circuit is programmed to operate as a decimator or as an interpolator. These signals ensure appropriate operation of the half-band filter in each mode. Among the input signals to the control unit are the master clock and the mode select signals. In a decimate-by-two mode, a control signal drives the input register clock at full master clock speed, and the internal and output register clocks at half the master clock speed. Thus, for every two consecutive inputs, only one output is obtained by summing the outputs of the even and odd taps subfilters. In interpolate-by-two mode, the filter control unit slows down the effective input register clock rate so as to match the internal register clock rate, while the output register clock is driven at full master clock speed. Hence, each output is obtained by interlacing the even and odd taps filter's outputs. The timing control unit also generates signals that control the operation of the data flow controller, and the interpolation and mode selection multiplexers. Figures 4.8 and 4.9 summarize the control issues in both decimation and interpolation by two modes [28, 29].

Figure 4.8 Decimation mode control

Figure 4.9 Interpolation mode control

4.3.2. Datapath Controller (DPC)

The DPC circuit routes the input data samples from the input port to the internal processing elements of the half-band filter according to the mode of operation. The data routing paths are based on the mode of operation; in decimate-by-two mode, the DPC demultiplexes the input data into even and odd sample streams which are processed by the poly-phase subfilters. Consequently, input samples are alternatively latched in the even and odd taps subfilters every second clock cycle. Figure 4.10 illustrates the operation of DPC in decimation mode. In interpolate-by-two mode, DPC feeds the input data to both subfilters. Figure 4.11 illustrates the operation of DPC in interpolation mode. In decimation, this operation is guaranteed by controlling DPC by a signal which runs at the same rate as the internal clock rate. While in interpolation, the control signal changes and becomes identical to the internal clocking signal [28].

Figure 4.10 DPC operation in decimate-by-two mode

Figure 4.11 DPC operation in interpolate-by-two mode

4.3.3. Even and Odd Delay Lines

In hardware implementation of digital filters, the z^{-1} terms in Equation 4.1 correspond to temporary (duration of a single clock cycle) data storage elements. These elements are often implemented as shift registers. In FIR filters, the registers form one or more tapped delay lines in which data is transferred from one register to the next at the completion of current output computation. To compute each output, the input data samples stored in the registers are shifted to a common point where they are multiplied by the appropriate coefficients [30]. According to Equation 4.2, the half-band filter is split into two subfilters consisting of odd numbered coefficients (odd taps) ($h_1, h_3, h_5, \dots, h_{27}$) and even numbered coefficients (even taps) ($0, 0, \dots, 0, h_{14}, 0, \dots, 0, 0$) respectively, and together they perform sum-of-products required to implement a 27-tap half-band filter in an architecture capable of supporting decimation and interpolation filtering modes.

In polyphase implementation, when the sampling rate conversion is equal to two, an N -tap half-band filter can be decomposed into two polyphase subfilters, where the odd taps subfilter consists of $(N+1)/2$ delay units and the even taps subfilter consists of $(N+1)/4$ delay units. Therefore, in the actual design, the odd taps subfilter contains 14 registers that form the odd tapped delay line, and the even taps subfilter has 7 registers that form the even tapped delay line. These registers are simply D-type flip-flops, each having two output ports that provide the data and its inverted form (Q and QB). The data is transferred from one register to the next along these delay lines. The half-band decimation and interpolation filter accepts 12-bit 2's complement input data. Multiplication of the data samples by positive coefficients requires the data bus to be taken from the data port Q of the registers, while multiplication by negative coefficients requires the data bus to be taken from the inverted data port QB of the registers. This is because in 2's complement, multiplication by a negative number requires bit inversion and addition of 1 to the least significant bit (LSB). The addition of multiple 1's is optimized before being incorporated into the bit pattern of partial-products. The overall addition is then performed by the multiplier-accumulator structure.

4.3.4. Multiplier-Accumulator (MAC1 and MAC2)

4.3.4.a. Introduction

Fast digital multipliers belong to the most important building blocks in VLSI chips for advanced digital processing. DSP applications place great demands on the multiplier cores for the following reasons. First, they are large circuits, where the effect of the circuit as well as the properties of the interconnects affect circuit performance [31]. There are many techniques that may be used to perform multiplication. The choice is based on factors such as speed, throughput, numerical accuracy, and area. As a rule, digital multipliers are classified by the format in which data words are accessed. Generally, multiplier circuits must perform two operations: partial-products generation and partial-products row

reduction, and final product addition[32].

4.3.4.b. Concept of Architecture

The 27-tap decimate/interpolate-by-two half-band FIR filter is split into two subfilters, one consisting of odd taps implementing the following taps (1, -2, 4, -8, 13, -25, 82, 82, -25, 13, -8, 4, -2, 1), and the second consisting of even taps implementing the only non-zero even tap, with a value of 130. Note that each integer coefficient can be written either as a power-of-two or as a sum of powers-of-two form (i.e., 2^k or $\sum 2^k$, where $k > 0$). Thus, the partial-products terms are generated by simply shifting the data sample bits by k positions to the left. The multiplier-accumulator accepts 12-bits 2's complement data, and as a result the odd taps subfilter generates a tree of rows of bits of height 26, while the even taps subfilter generates a tree of rows of bits of height 2. The total number of rows is 28, representing the number of power-of-two numbers used in the decomposition of the filter coefficients. The total number of partial-products bits is $28 \times 12 = 336$ bits. Figure 4.12 shows the bit pattern of the generated even and odd tree; the fixed pattern of 1's generated from multiplication of data words by negative coefficients is also included.

After partial-products generation, adders are used to reduce the tree height from 26 to 2 rows [33, 34]. Following the terminology introduced in [34] a name counter is used. An (n, m) counter is a combinational circuit with n inputs and m outputs. A full-adder is a $(3,2)$ counter, which is the key macrocell used in arithmetic multiplication. Figure 4.13 shows a gate level schematic of a full-adder with three inputs.

Figure 4.12 Bit pattern of generated partial products

Figure 4.13 A gate level schematic of a full-adder

The architecture of the multiplier-accumulator achieves parallel partial-products reduction in two stages which consist of, a combinational tree circuit followed by a fast adder. The first stage reduces the partial-products to two rows which are summed by the second stage to yield the final result. The partial-products reduction circuit is implemented as a regular array of vertical slices, each slice containing counter cells. It is important to realize that the total delay in the multiplier-accumulator is not only related to the number of stages used to reduce the tree height but also depends strongly on the delay associated with the interconnection wiring capacitance. High-order counters are used to ease the routing of signals throughout the multiplier-accumulator structure, as well as minimizing the wiring delay. Figure 4.14 shows the overall architecture of the multiplier-accumulator MAC1.

4.3.4.c. High-order Counter Design

The counter (5,3) is realized by applying one (3,2) counter to three inputs (namely x_2, x_1, x_0) to yield two contemporary outputs, the temporary carry-out (t_1) and the temporary sum (t_0). The temporary carry-out (t_1) will be passed to the left stage. The carry-in (T_1) is taken from the right stage with the remaining input x_3 , together with the temporary sum (t_0) to generate the final outputs y_0 and y_1 as shown in Figure 4.15. Each (5,3) counter contains the equivalent of two (3,2) counter delays [34]. The same technique can be applied to design a (9,5) counter using two (3,2) counters and one (5,3) counter as shown in Figure 4.16. The inputs are partitioned into two groups (x_5, x_4, x_3 as group 1, and x_2, x_1, x_0 as group 0), and a (3,2) is applied to each group. Thus, four signals are obtained, two of which are of weight 2^n and two of which have a weight of 2^{n+1} . Two others signals of weight 2^n are carried in from the right stage. Thus, a total of four signals of weight 2^n will be fed to the (5,3) counter as explained above. Three equivalent (3,2) counter delays stages are counted for each (9,5) counter. Figure 4.17 shows a slice of the partial-product reduction array implemented with (9,5) and (5,3) counters [34].

Figure 4.14 Overall architecture of the 27-taps half-band filter multiplier-accumulator MAC1

Figure 4.15 Constructing a (5,3) counter from (3,2) counters

Figure 4.16 Constructing a (9,5) counter from (3,2) counters

4.3.4.d. Implementation Procedure

As shown in Figure 4.3, MAC1 performs height reduction only on the tree of rows of bits of height 26 resulting from the multiplication of input data samples by odd numbered coefficients. The tree resulting from the multiplication of data samples by even numbered coefficients is of height 2. Starting with 26 partial-product rows, and after applying the (9,5) counter (four (9,5) counters and two rows must be carried in) at the first stage, the tree height is reduced to 10 rows. One (9,5) counter and (5,3) counter are applied at the second stage, and the tree is reduced to 4 rows. By applying a (5,3) counter at the third stage the tree is finally reduced to 2 rows. Since there are counters with different orders within the same stage, the delay through the stage is calculated taking into consideration only the counters with the highest order. Thus, in this case, an equivalent of 8 (3,2) counter delays was found through the whole MAC1 structure. Note that the same delay was found using only (3,2) counters, and the only advantage of using higher order counters is the ease of wiring and structure regularity.

The presented architecture has been extended to support sign bit processing. The sign bit processing is performed according to the rules of calculation with 2's complement numbers. By duplicating the most significant bit (MSB) of each row of bits, extra slices of XOR gates are added at the furthest left of the multiplier-accumulator structure to perform addition of the MSB's without a carry-out bit. To prevent overflow from occurring, computation based on the dynamic range of the filter allocates a 22-bit result to the filter output, so that even the maximum expected output will fit in this given wordlength [32].

After reducing the even tree to two rows, the multiplier-accumulator MAC2 performs row reduction on both the even and odd tree. MAC2 consists of only one stage of (5,3) counters and it is dedicated to decimate-by-two mode only. Finally the resulting tree has a height of two rows. The delay through the whole MAC2 structure is equivalent to two (3,2) counter delays

Figure 4.17 Slice of MAC1 array where counters are used for partial-products reduction

4.3.5. Interpolation and Mode Selection Multiplexers

4.3.5.a. Interpolation Multiplexer (MUX1)

In interpolation mode, every output sample is derived by multiplexing the outputs of the two polyphase subfilters into a single data stream with twice the input sample rate. MUX1 is dedicated to interpolate-by-two mode. The operation of MUX1 is shown in Figure 4.18 and is controlled by the select signal. In decimate-by-two mode this signal is set low, and in interpolate-by-two mode it runs at the master clock speed [28].

Figure 4.18 Operation of interpolation multiplexer MUX1

4.3.5.b. Mode Selection Multiplexer (MUX2)

Since the half-band filter has two modes of operation, the function of the MUX2 is to select the appropriate mode of operation. Another important feature of the MUX2 is resource sharing: both decimation and interpolation share the last stage of the multiplier accumulator which is the vector merge adder (VMA). The mode selection is achieved by setting the control signal (*mode_select* = 1, decimate-by-two mode and *mode_select* = 0, interpolate-by-two mode).

4.3.6. Vector Merge Adder (VMA)

The VMA is the final stage of the multiplier-accumulator structure. In decimation as well as in interpolation the VMA sums two rows of the reduced tree of bits. The implementation of VMA can be accomplished in several possibilities of design. In order to achieve an optimal implementation and meet the system speed requirements, the VMA is implemented as a fast carry-look-ahead adder. This is a 19 bit adder that adds two numbers to produce a 19 bit result with no carry out. The adder is composed of identical groups of 4 bit carry-look-ahead adders

4.3.7. Output Data Flow Controller (ROUNDER)

The ROUNDER sets the precision of the output data by controlling the position of the effective least significant bit of the output port. This is achieved by adding a rounding

bit to the next LSB bit and zeroing all bits below the rounding bit. The bus width is controlled via a 3-bit control bus *RND* and an enable signal *R_EN*, and the output precision ranges from 14 to 22-bits, making the output port of the filter circuit more flexible in interfacing with other DSP components.

4.4. Implementation Considerations

Among the major global issues that have been considered in the design of the chip are modularity of the design, clock distribution, clock skew control and power consumption. These issues can significantly affect the entire design at all levels including algorithm, architecture, circuit and layout. In order to achieve high speed signal operation, issues of increasing the MAC1 throughput must be considered. As an alternative, pipeline stages may be added in the MAC1 structure to meet the target throughput. The cost of pipelining is latency. The need to incorporate levels of pipeline stages comes at the expense of an increase in area, and power consumption [32, 33]. Figure 4.19 shows the pipeline stages inserted in the MAC1 structure. To avoid truncation errors, the output data is always internally rounded before it is latched into the output register. Rounding is user selectable, and can be programmed to round the output data from 22 down to 14-bit values.

During the design cycle, a great deal of attention was given to the routing of the critical signals such as the master clock and the master reset to ensure the consistency of the circuits with MOSIS CMOS 1.2 μm technology design rules. The design rule of distributing the global clock network allows a maximum skew of 2 ns. Clock skew occurs from two sources, one is the latency skew which results from similar size drivers with different capacitive loads, the other is the clock routing caused by different path lengths from a driver to two clocked elements. Throughout the design process, the clock skew was kept to a minimum by matching the ratio of the capacitive load to the driver size [35] (e.g., an

inverter INV1 can drive only 4 clocked elements), and the skew associated with routing the clock was minimized by a geometrically balanced clock tree.

Figure 4.19 Block diagram of the multiplier-accumulator structure with pipeline registers

4.5. Summary

The design of a dedicated 27-tap half-band decimation and interpolation by two filter was presented. This is a linear-phase, lowpass, mode programmable interpolating and decimating single-chip digital filter that requires no tuning. The filter can significantly reduce the complexity of traditional analog anti-aliasing and anti-imaging filters in a radio transceiver without compromising the signal bandwidth or attenuation. This can be achieved by using the filter as a decimating post-filter, or as an interpolating pre-filter. The VLSI implementation is based on polyphase architecture consisting of even and odd tap subfilters running at a lower data rate. The filter coefficients were generated using the Parks-McClellan algorithm, with minimum ripples in the filter passband and stopband. The coefficients are fixed, and the only user programming required is the selection of the mode of operation (decimation or interpolation). The filter coefficients were subject to scaling and rounding which yielded a set of power-of-two numbers. As a result, the filter multiply and add operations based on finite wordlength were implemented by hardwired shift and add operations. With power-of-two coefficients, the filter complexity and power consumption were reduced and the throughput was increased, as compared to implementing a multiplier for each tap. Note also that the symmetry property was not exploited in this design due to the simplicity of the coefficients. The pre-addition of two data samples before a multiplication by a common coefficient would offset the advantage of power-of-two coefficients by adding extra hardware overhead.

Chapter 5

Hardware Synthesis of a Programmable Decimation and Interpolation FIR Digital Filter

5.1. Introduction

As the application specific integrated circuit (ASIC) design process becomes increasingly complex due to the design density and shorter design cycles, logic synthesis and optimization tools become more important in the design process. These tools offer a reliable method of keeping pace with the demands of the marketplace by shortening the time-to-market for the design. At the gate level, synthesized circuits are competitive with hand-crafted circuits, since optimization algorithms can further improve the circuits in terms of area and performance. VHSIC hardware description language (VHDL) can be efficiently used for the specification, modelling, simulation, and hardware synthesis of ASIC's. VHDL models have the advantage over schematics in retargeting an ASIC, field programmable gate array (FPGA) or programmable logic devices (PLD) technology, in reusing parametrized modules, in setting optimization constraints in terms of what area/

speed trade-off is desired, and in debugging at a higher level [36]. VHDL is a high level description language for system and circuit design. The language supports various levels of abstraction. These include structural, dataflow and behavioral descriptions. In the next sections, hardware synthesis with VHDL design methodology will be used for modelling and implementing a programmable decimation and interpolation-by-two half-band FIR digital filter.

5.2. Generic Implementation Issues

5.2.1. Design Specification

The first step in designing ASIC's using hardware description languages such as VHDL is the creation of a behavioral code representing circuit functionality [27, 38]. This section describes the steps involved in the specification stage of the design of a digital FIR half-band decimation/interpolation-by-two filter. In the initial phase of the design, an algorithm that specifies the input-output relationship of the circuit is tested. A half-band FIR filter can be described by the convolution equation given by

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \quad (5.1)$$

where $x(n)$ and $y(n)$ are the input and output samples at time n , $h(k)$ are the filter coefficients and N is the filter order. A 7-tap half-band FIR filter has been taken as a design example, with a set of coefficients $(-1, 0, 8, 14, 8, 0, -1)$. Note that the order N and the impulse response $h(k)$ are taken as design parameters. The VHDL model created to implement Equation 5.1 is shown in Appendix A-1. The entity **HB_FIR** consists of ports X and Y respectively for the input and output, a generic array that defines the set of the FIR half-band filter coefficients, and a clock CLK . An integer data type can be used for the inputs and outputs at this level.

In order for the filter to operate in real time, it is necessary to perform all filtering operations before the arrival of the next input sample. The time required to perform these operations places a constraint on the minimum time between consecutive samples and the corresponding maximum sampling frequency. A target clock speed of 50 MHz (sampling period of 20 ns) has been specified as the expected operating speed of the system. The architecture consists of a single process model activated by the clock *CLK*. The process contains a variable *REG_VAL* which acts as an input register. For each clock, the input data is shifted into the register, specifically into *REG_VAL(0)*, and the past inputs get shifted to higher locations of the register. A sum of the term by term product of the register contents with the filter coefficients is stored in a temporary variable *REG_TEMP*. The temporary value is then assigned to the output and this implements Equation 5.1. Functional simulation is performed in order to validate the model, and the same set of input values is used as stimulus of the VHDL model. Timing simulation results are shown in Figure 5.1. The filter outputs follow exactly the results predicted by Equation 5.1.

Figure 5.1 Simulation results of VHDL behavioral model of half-band FIR filter

5.2.2. Architectural Considerations

The programmable half-band filter tap cells take advantage of the symmetry in the filter coefficients by pre-adding data samples prior to multiplication. This allows a 7-tap filter to be implemented using only three multipliers. The coefficients are loaded into the

filter static memory via a programmable interface, where each coefficient is multiplied by the sum of two corresponding data samples; the multiplier output is then registered to form a tap cell output. The chip is serially programmed through the control of two input pins, the choice of a serial programming method for the filter chip was made mainly to reduce the number of I/O pins. Figure 5.2 shows a functional block diagram of a programmable 7-tap decimate/interpolate-by-two half-band FIR filter. The design is implemented using polyphase architecture. By adding I/O control circuitry, a single chip design is used to implement both decimation and interpolation. Both the impulse response $h(n)$ and the filter's operation mode are user-selectable, hence allowing chip programmability and reconfigurability.

Figure 5.2 Block diagram of a programmable decimate/interpolate-by-two half-band FIR filter

5.3. Programmable Filter Functional Block Overview

The programmable 7-tap decimate and interpolate-by-two half-band FIR filter is implemented in a parallel processing configuration optimized for sums-of-products operations. The internal VLSI architecture of the even and odd taps subfilter is shown in Figures 5.3(a) and (b) respectively. Each tap consists of a 12x8 bit array multiplier, a 12 bit adder,

a coefficient memory register and a 20 bit adder along with two pipeline registers inserted in the computation path to achieve the desired throughput. The number of delay elements in the even and odd taps subfilter are the result of the polyphase decomposition of the transfer function of the original half-band filter. An external DSP processor would control the loading/unloading of the filter coefficients, the input and output data bus [39].

Figure 5.3 The internal VLSI architecture of the odd taps (a) and even taps (b) subfilters

The inputs that are connected to the input port O are the odd data samples, and those connected to the input port E are the even data samples. In decimation mode, the input data bus is connected to port O during one cycle and to port E during the next cycle. In interpolation mode, the input data bus ties the ports E and O together. Q1 and Q2 are the outputs from the odd and even taps subfilter respectively. In decimation mode, the outputs Q1 and Q2 are summed to form the decimated output stream running at half of the input rate, while in interpolation mode, the outputs Q1 and Q2 are multiplexed into a single interpolated output running at twice the input rate.

The symmetry adders are used to exploit the symmetry property only in the odd taps subfilter, since in the even taps subfilter there is only a single non-zero tap. Let us assume that $(h_0, h_1, h_2, h_3, h_4, h_5, h_6)$ is the set of the 7-tap half-band FIR filter coefficients. Figure 5.3 shows that only three multipliers *ML1*, *ML2*, and *ML3* are needed to implement the filter. In the odd taps subfilter, Tap_h₀ replaces the multipliers h_0 and h_6 (h_0 and h_6 are of equal value), and Tap_h₂ replaces the multipliers h_2 and h_4 (h_2 and h_4 are of equal value). In the even taps subfilter, Tap_h₃ is the same as the multiplier h_3 , while the multipliers h_1 and h_5 are zero. The filter taps (Tap_h₀, Tap_h₂, Tap_h₃) are loaded via the input port *COBUS* as shown in Figure 5.2. When cascading more than one filter module, the same input port can be used to write coefficients to the taps of different filter chips. Figure 5.3 also shows some pipeline registers that were internally inserted in each filter tap to increase the throughput.

5.3.1. Filter Controller Circuit

The main function of this circuit is the provision of timing control signals for different logic blocks of the filter chip. The VHDL behavioral description of the control circuit is listed in Appendix A-2. The control signals are implemented as activated processes in the VHDL description, where the sensitivity lists contain the process triggering signals which are the clock (*clk*), the reset signal (*reset*), the mode selection signal (*mode_select*),

the clock divided-by-two (*clk2_int*), and its inverted version (*clk2b_int*). The filter controller consists mainly of a clock divider that generates two internal clock signals running at half the master clock rate with opposite phases, and multiplexers which select the appropriate control signals (*in_reg_cnt*, *out_reg_cnt*, *fer_reg_cnt*, and *mux_eo_cnt*) for the input and output registers, the data flow controller (DFC), and the interpolation multiplexer respectively. Figure 5.4 shows a gate level schematic of the control circuit which resulted from hardware synthesis of a VHDL register-transfer-level (RTL) description of the controller circuit. Table 5.1 summarizes the states of the control signals according to the mode of operation of the filter and relatively to the master clock rate. Figures 5.5 and 5.6 show the timing diagrams for the control circuit in decimate-by-two and interpolate-by-two modes.

Figure 5.4 Gate level schematic of the programmable filter control circuit

Table 5.1: Control signals states in decimation and interpolation mode

Circuit Block	Signal	Decimate-by-two mode	Interpolate-by-two mode
Input register	<i>in_reg_cnt</i>	<i>clk</i>	<i>clk2</i>
Data flow controller	<i>fer_reg_cnt</i>	<i>clk2</i>	<i>clk2b</i>
Output register	<i>out_reg_cnt</i>	<i>clk2</i>	<i>clk</i>
Interpolation multiplexer	<i>mux_eo_cnt</i>	inactive	<i>clk2</i>

Figure 5.5 Control signals timing diagram in decimate-by-two mode **Figure 5.6 Control signals timing diagram in interpolate-by-two mode**

5.3.2. Coefficients Loading Interface

The serial loading of the 7-tap decimate/interpolate-by-two half-band FIR filter coefficients is controlled by a programmable interface as depicted in Figure 5.2. The interface has two control inputs, *Write_Enable* and *Tap_Select(1:0)*. When the *Write_Enable* input is high, one out of the three taps is written into on the rising edge of the clock. The *Tap_Select* bus selects which one of the three taps is to be written, then the coefficients bus places the available user-selectable data into the corresponding coefficient memory register. The filter coefficients are loaded by setting the *Tap_Select* input as follows:

Tap_Select(0)=0, Tap_Select(1)=0, CLK 0->1

Tap_Select(0)=1, Tap_Select(1)=0, CLK 0->1

Tap_Select(0)=0, Tap_Select(1)=1, CLK 0->1.

A synthesizable VHDL description created to design the programmable interface is shown in Appendix A-3. This is an RTL description of the circuit that can be simulated and synthesized into a gate-level schematic. Figure 5.7 shows a gate level schematic of the programmable interface circuit produced by the hardware synthesis. The timing diagram of the circuit operation is also shown in Figure 5.8. The programmable interface is clocked by a signal that runs at half the master clock rate. Thus, for the first six master clock cycles, the filter coefficients are loaded into the memory registers from the *COBUS* bus. The filter can process input data after the loading of the last filter coefficient. The timing diagram in Figure 5.8 shows three coefficients, *ML1*, *ML2* and *ML3*, chosen with hexadecimal values of 49, 4E and 52. After setting the *Write_Enable* and the *Tap_Select* pins appropriately, these coefficients are loaded into the memory registers of the filter after six clock cycles.

Figure 5.7 Gate level schematic of the programmable coefficients loading interface

Figure 5.8 Timing diagram for the programmable half-band filter coefficients loading operation

5.3.3. The Symmetry adders

These are 12-bit carry-lookahead adders. Carry-lookahead circuits are special circuits that can reduce the time to perform addition [40, 41]. The symmetry adders add two 12-bit data samples without the carry-in bit to produce a 12-bit result without the carry-out bit. The adders are designed to exploit the symmetry property in the half-band decimate/interpolate-by-two FIR filter, so as to reduce the number of multipliers. The 12-bit adder is composed of three 4-bit carry-lookahead adders. The basic idea of carry-lookahead logic is to express the carry-out of each stage in terms of all the inputs $A_i, A_{i-1}, \dots, A_0, B_i, B_{i-1}, \dots, B_0$, and the carry-in C_0 [40]. Each adder has its own generate, propagate, and carry functions, where the carry generate G_i , and propagate P_i are given by:

$$\begin{aligned} G_i &= A_i B_i \\ P_i &= A_i \oplus B_i \end{aligned} \quad (5.2)$$

The carry-out logic can be expressed in terms of carry propagate and carry generate as follows:

$$\begin{aligned} C_1 &= G_0 + P_0 C_0 \\ C_2 &= G_1 + P_1 G_0 + P_1 P_0 C_0 \\ C_3 &= G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0 \\ C_4 &= G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0 \end{aligned} \quad (5.3)$$

Figure 5.9 shows the schematic for the adder stage with propagate and generate outputs. The carry-lookahead circuits for a 4-bit adder are given in Figure 5.10. Note that it takes one gate delay to compute the propagate and generate signals and two gate delays to compute the sum. When the P_i and G_i are available, the subsequent carries, C_1, C_2, C_3 , and C_4 , are computed after two more gate delays (three gate delays in total). The sum bits can be computed in just one more gate delay, the final sum bit being available after four gate

delays, compared with seven gate delays in adders without carry-lookahead. Therefore, the addition time is cut by half.

Figure 5.9 Add with generate and propagate outputs

Figure 5.10 Four-bit carry-lookahead logic

A 12-bit carry-lookahead adder can be implemented with three 4-bit adders, each employing its own internal 4-bit carry look-ahead logic. Each 4-bit computes its own group carry propagate and generate: the group propagate is the AND of P_3, P_2, P_1, P_0 , while the group generate is the expression, $G_3 + G_2P_3 + G_1P_3P_2 + G_0P_3P_2P_1$. A second-stage circuit computes the lookaheads and the output carries between first-stage 4-bit adders. The logic is identical to that of Figure 5.10. Figure 5.11 shows the block diagram of a 12-bit carry-lookahead adder.

Figure 5.11 12-bit adder using hierarchical carry-lookahead logic

5.3.4. The 20-bit Adder

The adder is composed of five 4-bit carry-lookahead adders and is architecturally similar to the 12-bit carry-lookahead adder shown in Figure 5.11. This adder is used to add the internal outputs in the odd taps subfilter to produce the 20-bit output Q1 as shown in Figure 5.3. In decimation mode another 20-bit adder is used to sum the outputs Q1 and Q2, the result forms the decimation output.

5.3.5. The 12x8-bit Array Multiplier

Multiplication is one of the basic operations in digital signal processing systems

providing filtering, correlation, and frequency analysis. High-speed multiplication is required for real time signal processing. The Booth algorithm based design is used to implement high-speed parallel multiplication in the digital FIR half-band filter. The method uses a bit-recoding technique in a planar array to implement 2's complement multiplication combinational [31, 41]. Table 5.2 illustrates the multiplier bit-pair recoding technique, where different combinations of positive and negative multiplicands and multipliers are presented.

Table 5.2: Multiplier bit-pair recoding in Booth algorithm

Multiplier bit-pair $i+1$	i	Multiplier bit-pair $i-1$	Multiplicand multiples to be added	Status
0	0	0	$0x$	No string
0	0	1	$+1x$	End of string
0	1	0	$+1x$	Single 1(+2 -1)
0	1	1	$+2x$	End of string
1	0	0	$-2x$	Begin of string
1	0	1	$-1x$	End/Begin of string (+1 -2)
1	1	0	$-1x$	Begin of string
1	1	1	$0x$	String of 1's

A Booth-recoded multiplier scans three bits of the multiplier at a time to determine whether to add $0xM_c$, $1xM_c$, $-1xM_c$, $2xM_c$, $-2xM_c$ (M_c denotes the multiplicand) to that rank of the multiplicand.

The multiplier structure is modular and consists of nine identical 4-bit slice modules. Each module has one Booth decoder, four 5-input multiplexers, and one 4-bit carry-lookahead adder. The Booth-multiplier uses decode cells, each cell observes 3-bits of the multiplier and generates the control signals which are used in the array adders. This deliv-

ers the appropriate version of the multiplicand to the adder, when added with the incoming partial-products. Figure 5.12 shows an individual Booth-decode cell schematic.

Figure 5.12 Gate level schematic of Booth decode cell

The organization of an individual multiplexer is shown in Figure 5.13. One of the five inputs to each multiplexer is selected by the decoder to be added to the corresponding partial-product bit. Note that during $2x$ and $-2x$ operations the multiplicand bits are shifted one bit position to the left before being added to the partial-product.

Figure 5.14 shows an overall structure of a Booth module. The inputs to the module are the incoming partial-products from a previous stage pp_{k+3} , pp_{k+2} , pp_{k+1} , pp_k (initially all zero), and 4-bit slice a_{j+3} , a_{j+2} , a_{j+1} , a_j , which represent the multiplicand. The 4-bit carry-lookahead adder is similar in architecture to the one described in section 5.3.3. However, the carry-in to the first section is the OR of the group carry-out of $i-1^{th}$ module

(logic 0 if the i^{th} module is the rightmost module in a row), and logic 1 when $-1xM_c$ is selected. The carry-in to the second section is the OR of the carry-out from the first section (logic 1 when multiplication by -2 is selected), and logic 1 when $-2xM_c$ is selected. This ensures that a 1 is always added when multiplying by a negative number according to the 2's complement representation rules.

Figure 5.13 Gate level schematic of Booth multiplexer

Figure 5.14 Module structure in a 12x8-bit Booth array multiplier

5.4. Summary

Top-down VHDL based design methodology was used for the hardware synthesis and functional simulation of a programmable half-band decimation and interpolation-by-two digital filter. By creating a high-level functional description of the half-band filter circuit, logic synthesis tools were used to automatically translate the models into gate-level implementation. The use of VHDL allows more focus on the exploration of various IC's technologies, as well as the improvement of circuit design.

The half-band decimator and interpolator filter design was captured by modelling the filter behavior. The partitioning of the filter design into smaller and manageable hardware blocks resulted in a shorter design cycle and simplified development process. The design and VLSI implementation of a 18k transistor, programmable 7-tap decimation and interpolation by two half-band filter, with 12-bit 2's complement data input, 8-bit coefficients and 20-bit data output was presented. The filter was designed with the polyphase representation as a target implementation. The half-band filter input interface enables a serial loading of user-selectable coefficients which can be generated using various design methods. The ability to change many of the chip spectrum characteristics makes it a suitable macro for a wide range of DSP applications.

2 of/de 2

PM-1 3 1/2" x 4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

Chapter 6

Verification Procedures and Simulation Results

6.1. Introduction

In this chapter, results of simulations of the fixed and programmable coefficients half-band decimation and interpolation-by-two filters are presented. Simulations to verify the hardware functionality of the circuits, are performed in both decimation and interpolation by two modes. These filters are simulated in stand-alone mode, and in a chain of two modules, in order to verify their cascadability and operation at higher decimation and interpolation ratios. In digital filtering applications, frequency spectrum analysis is very important because it depicts the response of DSP systems to real-time signals. For this reason, MATLAB and QUICKSIM simulators were used to validate the VLSI implementation [42]. Real-time input signals with various frequency tones are generated and fed to the filter circuits, and frequency domain verification is performed to check for any discrepancy between the hardware performance and the expected frequency response. Spectral analysis is performed in both modes of operation on both the fixed-coefficients and programmable decimate/interpolate-by-two half-band FIR filters.

6.2. Design Tools and Technology

6.2.1. Introduction

The introduction of advanced CAD tools has vastly reduced design cycle time and improved the quality of products. CAD tools allow designers starting from a given design specification, to capture its schematic and perform extensive verification and simulation at different levels. Also, the design layout can be carried out in order to manufacture the IC. All CAD tools used for the design decimation/interpolation digital filters are supplied by Mentor Graphics Corporation (MGC) and run on HP stations with the Unix operating system.

6.2.2. Schematic Capture Tools

Schematic capture is the process of drawing a schematic with a computer and storing it so that it can be used in other processes. Design Architect was the schematic capture application. It is a multi-level environment that includes a schematic editor, a symbol editor, and VHDL code Editor. Design Architect offers many design options which include schematic capture, symbol creation, digital and analog component library access, property annotation, back annotation, and VHDL code creation.

The 27-tap decimate/interpolate-by-two half-band FIR filter circuit was implemented using the Mentor Graphics, MOSIS 1.2 μm CMOSN process standard cell library. The programmable 7-tap decimate/interpolate-by-two half-band filter was synthesized first using components from a generic library. Technology mapping was successfully executed by targeting physical gates from MOSIS library. Figure 6.1 illustrates the ASIC design flow in the MGC environment that was followed by the hardware synthesis.

Figure 6.1 ASIC design flow for hardware synthesis using Mentor Graphics tools

6.2.3. Simulation Tools

Simulation is the modelling and behavior analysis of an electronic design without the ownership costs of the physical hardware. Timing verifications were performed using the Quicksim logic simulator. Quicksim is an interactive 12-state timing simulator that can be used to simulate designs implemented in technologies such as CMOS, TTL, ECL [35].

Timing violations were checked using QuickPath. QuickPath is a critical path timing analyzer that can analyze all the portions of the schematic for timing errors without the need to perform extensive logic simulation. It performs two types of timing checks: delay checking (propagation delays, path length analysis), and constraints checking (setup and hold violations, slack analysis). These checks are useful for both PCB and ASIC designs to assure timing violation free hardware before the costly process of manufacturing.

6.2.4. Hardware Synthesis and Optimization Tools

HDL-based design methodology can be efficiently used to implement top-down design. The functionality of the design can be captured at an abstract-level. Simulation, synthesis and optimization tools are then used to transform the abstract HDL description into a physical design specifically targeted to an ASIC or FPGA technology [43].

System-1076 is the hardware description language, based on IEEE STD 1076-1987, that was used to debug, compile, and simulate the VHDL code for hardware synthesis. Synthesis and optimization were performed using Autologic VHDL. Autologic VHDL is an integral part of MGC multi-level design and analysis solution which includes design creation, mixed level simulations, and HDL-based synthesis.

6.3. Design Verification

6.3.1. Introduction

The functionality of both decimation and interpolation half-band filters was simulated using MATLAB and QUICKSIM simulators. Matlab simulation is used for the theoretical verification of the filter's spectral behavior. While Quicksim simulation verifies and checks the practical hardware implementation of the filter's circuits. For timing analysis, QUICKSIM is used to test the functionality of the design, by providing inputs stimulus.

6.3.2. Functional Simulation of the 27-Tap Decimate/Interpolate By Two Half-Band FIR Filter

Functional verification through frequency spectrum and timing analysis was carried out on the 27-tap decimate/interpolate-by-two half-band filter. Functional simulations were run with a throughput of 50 Msamples/sec under typical operating conditions. While the decimation and interpolation ratio was fixed to a factor of two, higher ratios can be

obtained by cascading more than one filter structure. The filter was simulated as a stand alone module (decimation/interpolation ratio of two) and as a chain of two modules (decimation/interpolation ratio of four) with a clock frequency of 50 MHz. Quicksim simulation results of the half-band FIR filter operating as a decimator and interpolator are shown in Figures 6.2 and 6.3 respectively. Figure 6.2(a) illustrates an input signal spectrum, with two in-band sine waves with frequencies of 4 MHz and 9.5 MHz sampled at 50 MHz, that were combined with white noise and with multiple high frequency components with frequencies greater than $f_s/2$, where f_s is the filter output sampling rate. Figures 6.2(b) and (c) show the output signal being decimated by 2 and by 4 to sampling rates of 25 MHz and 12.5 MHz respectively. Figure 6.3(a) depicts a 12.5 MHz sampled input signal composed of two sine waves at 0.62 MHz and 1.87 MHz. This signal is interpolated by 2 and by 4 to sampling frequencies of 25 MHz and 50 MHz respectively, as shown in Figures 6.3(b) and (c).

6.3.3. Functional Simulation of the Programmable Half-Band FIR Filter

The same functional verification procedure was also carried out on the programmable 7-tap decimate/interpolate-by-two FIR half-band filter. A set of coefficients was loaded into the filter memory registers with the hexadecimal values (49, 00, 4E, 52, 4E, 00, 49). Timing and spectrum analysis were performed with a clock period of 20 ns. VHDL timing simulations of the programmable filter simulated in decimate-by-two and interpolate-by-two modes are respectively shown in Figure 6.4(a) and (b). The programmable filter was also verified through Matlab analysis of Quicksim simulation vectors. The input test signal consisted of multiple frequency tones. Two sine wave signals with frequencies of 5 MHz and 10 MHz respectively, were combined and additive noise was introduced. Then the signal was digitized and sampled at a sampling frequency of 50 MHz. Figure 6.5(a) depicts the input signal spectrum of the in-band signals and the quantization effects due to the finite wordlength of the input data bus. Figure 6.5(b) shows the frequency spectrum of the programmable filter operating as a decimator by two as ana-

Figure 6.2 Power spectra of the 27-tap decimation half-band filter
(a) Input signal spectrum, $F_s = 50$ MHz
(b) Decimated-by-two output signal spectrum, $F_s = 25$ MHz
(c) Decimated-by-four output signal spectrum, $F_s = 12.5$ MHz

Figure 6.3 Power spectra of the 27-tap interpolation half-band filter

- (a) Input signal spectrum, $F_s = 12.5$ MHz
- (b) Interpolated-by-two output spectrum, $F_s = 25$ MHz
- (c) Interpolated-by-four output spectrum, $F_s = 50$ MHz

lyzed using Matlab simulation; this represents the theoretical result. It consists of the input signal responses at frequencies of 5 MHz and 10 MHz sampled at a sampling frequency of 25 MHz. Figure 6.5(c) depicts the output frequency spectrum from the VLSI implementation using Quicksim, and as can be noted, the programmable filter frequency response satisfactorily agrees with the theoretical results. Figure 6.6(a) illustrates the input signal spectrum to the programmable filter operating as an interpolator by two. The spectrum consists of two sine waves with frequencies of 5 MHz and 10 MHz sampled at a frequency of 25 MHz. Figure 6.6(b) shows the output spectrum of the signal after it has been interpolated by two to a sampling frequency of 50 MHz

6.4. Summary

Functional verification through frequency spectrum analysis has been carried out on the decimating and interpolating half-band filters with fixed and programmable taps. In both decimation and interpolation modes, Matlab and Quicksim simulations were performed. Matlab simulation was used for theoretical verification, while Quicksim simulation verified the hardware implementation. The operation of the both ASICs was verified in the time and frequency domains by subjecting the circuits to test vectors consisting of sampled signals at various frequencies with additive noise. Data samples can be input into the filters at rates of up to 50 million samples per second, and the output sample rates can be one-half, two times, or four times the input sample rates. The simulation results based on MOSIS 1.2 μm CMOS technology, indicate that the fixed 27-tap and programmable 7-tap half-band filter throughput to be 50 Msamples/sec under typical operating conditions.

Figure 6.4 Timing simulations of the programmable half-band FIR filter in decimate-by-two (a) and interpolate-by-two (b) modes

Figure 6.5 Power spectra of decimate-by-two programmable half-band filter

(a) Input signal spectrum, $F_s = 50$ MHz

(b) Matlab output signal spectrum, $F_s = 25$ MHz

(c) Quicksim output signal spectrum, $F_s = 25$ MHz

Figure 6.6 Power spectra of interpolate-by-two programmable half-band filter

(a) Input signal spectrum, $F_s = 25$ MHz

(b) Output signal spectrum, $F_s = 50$ MHz

Chapter 7

Conclusion

A number of VLSI implementations of half-band filters have been reported in the literature [13, 25, 30]. These implementations which support decimation and interpolation, include programmable as well as dedicated architectures with fixed and generic coefficients with performances that range from 10 to 100 MHz. The effectiveness of half-band filters in multirate processing systems depend on the judicious selection of filter characteristics and architectures as well as efficient VLSI implementation to satisfy the throughput, power and area requirement. In this thesis, two different VLSI filter design methodologies were exercised and as a result two filter chips intended for decimation and interpolation by a factor of two were obtained, these filters can be cascaded to achieve higher power-of-two rate conversion ratios. Functional simulation results have shown the throughput of both filters to be 50 Msamples/s under typical operating conditions.

The first design was that of an integer fixed-coefficient 27-tap half-band FIR filter operating as a decimator and an interpolator by a factor of two. The filter chip uses power-

of-two coefficients, and by tailoring the architecture for a multiplier-free algorithm, both the arithmetic operations and the sensitivity of the coefficients were minimized. This makes it possible to design high throughput and low cost filter with a minimum number of multipliers, storage elements, and with no restrictions on the frequency response. This design has 23k transistors, with 12-bit 2's complement data input and 22-bit data output, with a user programmable rounding circuit which truncates the result internally and supports output data precision from 14 bits to 22 bits.

The second design was that of a programmable 7-tap decimate and interpolate-by-two half-band FIR filter, the filter chip was implemented using only three multipliers. This represents only one quarter the number of multipliers required to implement a typical FIR filter because of the symmetry in the frequency response. This design has 18k transistors, with 12-bit 2's complement data input, 8-bit 2's complement user-selectable integer coefficients which are loaded into the filter static memory via a programmable serial interface, and 20-bit data output. The high sampling rate and the ability to change many characteristics of the design makes this programmable filter a versatile macro with a diverse range of applications. The overall goal of this research project was to develop high-speed and low-power half-band digital decimation and interpolation filters as low cost building blocks for digital radio transceivers. The realization of such designs has indeed been achieved by taking advantage of the polyphase representation which allow the partition of the filter block into low-order with simpler architectures, and operating at a lower data rate.

The architectural and design techniques presented in this thesis involve the use of a DSP architecture which allows decimation and interpolation to be selectively performed by the same filter chip. Polyphase representation did not offer any advantage over conventional DSP implementations in terms of the number of arithmetic operations. Flexibility to realize a variety of the filter configurations and minimization of the computational load was the major gain from using such implementation.

Future work that could be undertaken with respect to the implementation of half-band decimation and interpolation FIR filters, presents a potential avenue of research in the area of communication and digital mobile radio. Improvements to this implementation can be made in several ways. To achieve a gain in speed and reduction in gate count, the encoding section of each multiplier in the programmable filter can be removed. This because the operating speed of a Booth multiplier is data dependent. The design can be expanded through the incorporation of more digital signal processing functions. In the cascaded configuration, more filters would be added to form a chain with the control circuitry being modified as necessary. This design approach can be used to implement integer and fractional sampling rate systems, where half-band filters can be used as filtering stages. There are a large number of parameters and trade-offs involved in the design of multistage decimator and interpolator systems, including the choice of the number of stages, the decimation or interpolation ratios, the filter requirements, and the actual filter designs at each stage. To achieve an efficient implementation, a thorough investigation of several suggested optimization procedures for design parameters selection is required.

To assure the necessary throughput of the filter, several functional blocks require internal rounding when the signal wordlength grows too large during filtering operations. While the results are satisfactory, the significance of the truncation has not been determined. This presents another option, where the filters could be optimized as a function of word size. In addition, hardware synthesis algorithms are not yet efficient in optimizing at very high abstract level. Therefore, the result of synthesizing efficient VHDL designs depends entirely on the coding style and on the setting of logic optimization constraints.

Appendix

Hardware Description of the Programmable FIR Filter

This section provides hardware description of some of the programmable half-band decimation and interpolation by two FIR filter blocks. Design re-use is one of the primary reasons to use VHDL. By maintaining a VHDL library of basic digital signal processing blocks, the development process of digital half-band FIR decimation and interpolation filters is greatly simplified. One example of building blocks that are used repeatedly in every filter design are adders and multipliers. The ability to use parametrized arithmetic modules made the design of the programmable decimation and interpolation filter easier to iterate without extensive hardware modifications. This design approach was accomplished by adopting a VHDL coding style which supports expansion of logic blocks. Design characteristics such as the input port width, filter order (number of taps), output port width, and maximum clock frequency are specified as generic parameters. The variable-width ports must be constrained when using generics. This provides better documentation of the relationship between the size of different ports, and allows the VHDL environment to per-

form a complete width consistency check automatically. Downstream applications, such as synthesis will require port size information and all user-selected values for defined generics.

Filter tap is considered to be the basic processing cell for filtering operations. By using this design approach, a generic tap entity was created from low-level generic logic blocks such as multiplier, adder, and pipeline registers. In top-level architecture, a generic structural VHDL entity was created, where all the tap cells were instantiated and tied together to perform the required sum-of-products operations.

Hardware description of the half-band decimation and interpolation filter blocks was captured with the goal of targeting polyphase filter implementations. This architectural issue is taken into account at the start of the design process so as to have a better control over the synthesis result. The overall VHDL design of the programmable decimator and interpolator filter includes a mix of algorithmic, RTL and structural description of hardware blocks, with the purpose of functional simulation and circuit synthesis. The top level structural description of the filter was created by instantiating all low-level blocks of the half-band filter and declaring all the physical interconnections. For functional simulation and verification, all of the filter sub-modules and top-level entity were instantiated as design-under-test entity in VHDL testbenches, where test vectors are applied as stimuli to all the filter input pins, and the output results were monitored from timing waveforms.

A-1

```

-----
-- Entity : hb_fir
-- Architecture : algorithmic
-- Model Description : Algorithmic VHDL Description of the Programmable 7-tap Half-Band
--                      Decimation and Interpolation Filter
-- Package Required : QSIM_LOGIC
-- Purpose : Simulation only
-----

```

```

library mgc_portable ;
use mgc_portable.qsim_logic.all ;

```

```

entity hb_fir is
port ( x      : in integer;
      clk     : in qsim_state;
      y      : out integer
      );
end hb_fir;

```

```

architecture algorithmic of hb_fir is
type array_of_integer is array (0 to 6) of integer;
begin
  process (clk)
    variable reg_val : array_of_integer := (0,0,0,0,0,0,0);
    variable reg_temp : integer := 0;
    constant coeff : array_of_integer := (-1, 0, 8, 14, 8, 0, -1);
    begin
      if (clk'event and clk = '1' and clk'last_value = '0') then
        for i in 6 downto 1 loop
          reg_val(i) := reg_val(i-1);
        end loop;
        reg_val(0) := x;
        temp := 0;
        for i in 0 to 6 loop
          reg_temp := reg_temp + coeff(i)*reg_val(i);
        end loop;
        y <= reg_temp;
      end if;
    end process;
  end algorithmic;

```

A-2

```
-- Entity: hblend1
-- Architecture: rtl
-- Model Description: Register-Transfer Level Description of the Programmable Filter
--                      Control Unit
-- Package Required: QSIM LOGIC
-- Purpose: Simulation and Synthesis
```

```
library ieee;
use ieee.std_logic_1164.all;

entity hblend1 is
    port (
        in1 : std_logic;
        in2 : std_logic;
        in3 : std_logic;
        in4 : std_logic;
        in5 : std_logic;
        out : std_logic;
        mode : std_logic
    );
end entity;

architecture rtl of hblend1 is

    signal q : std_logic;
    signal clk2_int : std_logic;
    signal clk2b_int : std_logic;

begin
    process (mode)
        begin
            if mode = '0' then
                q <= in1 and in2 and in3 and in4 and in5;
                q <= q;
            end if;
        end process;

        qb <= not q;
        clk2_int <= q;
        clk2b_int <= not q;

        process (mode, select, clk, clk2_int, clk2b_int)
            begin
                if mode = select then
                    if mode = '1' then
                        m_reg <= clk;
                        f_reg <= clk2_int;
                        out_reg <= clk2b_int;
                    else
                        m_reg <= clk2b_int;
                        f_reg <= clk2b_int;
                    end if;
                end if;
            end process;
        end process;
    end architecture;
```

```

end if
end if
end if
for x = 1:n
    for y = 1:n
end for
end for

```