THE UNIVERSITY OF CALGARY

Design and Bit-Serial Implementation of LDI Jaumann Digital Filters

by

Lorne Michael Smith

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND

COMPUTER ENGINEERING

CALGARY, ALBERTA

SEPTEMBER, 1993

ISBN 0-315-88631-5

Canada

Name Lorne M. Smith

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.

Engineering, Electronics and Electrical — SUBJECT TERM

0544 — 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 0998
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

Folklore 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
- Limnology 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
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 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 0986
- 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 0537
- 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

Nom _____

Dissertation Abstracts International est organisé en catégories de sujets. Veuillez s.v.p. choisir le sujet qui décrit le mieux votre thèse et inscrivez le code numérique approprié dans l'espace réservé ci-dessous.

_____ ☐☐☐ U·M·I

SUJET CODE DE SUJET

Catégories par sujets

HUMANITÉS ET SCIENCES SOCIALES

COMMUNICATIONS ET LES ARTS

Architecture 0729
Beaux-arts 0357
Bibliothéconomie 0399
Cinéma 0900
Communication verbale 0459
Communications 0708
Danse 0378
Histoire de l'art 0377
Journalisme 0391
Musique 0413
Sciences de l'information 0723
Théâtre 0465

ÉDUCATION

Généralités 515
Administration 0514
Art 0273
Collèges communautaires 0275
Commerce 0688
Économie domestique 0278
Éducation permanente 0516
Éducation préscolaire 0518
Éducation sanitaire 0680
Enseignement agricole 0517
Enseignement bilingue et multiculturel 0282
Enseignement industriel 0521
Enseignement primaire. 0524
Enseignement professionnel 0747
Enseignement religieux 0527
Enseignement secondaire 0533
Enseignement spécial 0529
Enseignement supérieur 0745
Évaluation 0288
Finances 0277
Formation des enseignants 0530
Histoire de l'éducation 0520
Langues et littérature 0279
Lecture 0535
Mathématiques 0280
Musique 0522
Orientation et consultation 0519
Philosophie de l'éducation 0998
Physique 0523
Programmes d'études et enseignement 0727
Psychologie 0525
Sciences 0714
Sciences sociales 0534
Sociologie de l'éducation 0340
Technologie 0710

LANGUE, LITTÉRATURE ET LINGUISTIQUE

Langues
- Généralités 0679
- Anciennes 0289
- Linguistique 0290
- Modernes 0291

Littérature
- Généralités 0401
- Anciennes 0294
- Comparée 0295
- Médiévale 0297
- Moderne 0298
- Africaine 0316
- Américaine 0591
- Anglaise 0593
- Asiatique 0305
- Canadienne (Anglaise) 0352
- Canadienne (Française) 0355
- Germanique 0311
- Latino-américaine 0312
- Moyen-orientale 0315
- Romane 0313
- Slave et est-européenne 0314

PHILOSOPHIE, RELIGION ET THÉOLOGIE

Philosophie 0422
Religion
- Généralités 0318
- Clergé 0319
- Études bibliques 0321
- Histoire des religions 0320
- Philosophie de la religion 0322

Théologie 0469

SCIENCES SOCIALES

Anthropologie
- Archéologie 0324
- Culturelle 0326
- Physique 0327

Droit 0398
Économie
- Généralités 0501
- Commerce-Affaires 0505
- Économie agricole 0503
- Économie du travail 0510
- Finances 0508
- Histoire 0509
- Théorie 0511

Études américaines 0323
Études canadiennes 0385
Études féministes 0453
Folklore 0358
Géographie 0366
Gérontologie 0351
Gestion des affaires
- Généralités 0310
- Administration 0454
- Banques 0770
- Comptabilité 0272
- Marketing 0338

Histoire
- Histoire générale 0578
- Ancienne 0579
- Médiévale 0581
- Moderne 0582
- Histoire des noirs 0328
- Africaine 0331
- Canadienne 0334
- États-Unis 0337
- Européenne 0335
- Moyen-orientale 0333
- Latino-américaine 0336
- Asie, Australie et Océanie 0332

Histoire des sciences 0585
Loisirs 0814
Planification urbaine et régionale 0999
Science politique
- Généralités 0615
- Administration publique 0617
- Droit et relations internationales 0616

Sociologie
- Généralités 0626
- Aide et bien-âtre social 0630
- Criminologie et établissements pénitentiaires 0627
- Démographie 0938
- Études de l' individu et de la famille 0628
- Études des relations interethniques et des relations raciales 0631
- Structure et développement social 0700
- Théorie et méthodes. 0344
- Travail et relations industrielles 0629

Transports 0709
Travail social 0452

SCIENCES ET INGÉNIERIE

SCIENCES BIOLOGIQUES

Agriculture
- Généralités 0473
- Agronomie. 0285
- Alimentation et technologie alimentaire 0359
- Culture 0479
- Élevage et alimentation 0475
- Exploitation des pâturages 0777
- Pathologie animale 0476
- Pathologie végétale 0480
- Physiologie végétale 0817
- Sylviculture et faune 0478
- Technologie du bois 0746

Biologie
- Généralités 0306
- Anatomie 0287
- Biologie (Statistiques) 0308
- Biologie moléculaire 0307
- Botanique 0309
- Cellule 0379
- Écologie 0329
- Entomologie 0353
- Génétique 0369
- Limnologie 0793
- Microbiologie 0410
- Neurologie 0317
- Océanographie 0416
- Physiologie 0433
- Radiation 0821
- Science vétérinaire 0778
- Zoologie 0472

Biophysique
- Généralités 0786
- Medicale 0760

SCIENCES DE LA TERRE

Biogéochimie 0425
Géochimie 0996
Géodésie 0370
Géographie physique 0368
Géologie 0372
Géophysique 0373
Hydrologie 0388
Minéralogie 0411
Océanographie physique 0415
Paléobotanique 0345
Paléoécologie 0426
Paléontologie 0418
Paléozoologie 0985
Palynologie 0427

SCIENCES DE LA SANTÉ ET DE L'ENVIRONNEMENT

Économie domestique 0386
Sciences de l'environnement 0768
Sciences de la santé
- Généralités 0566
- Administration des hôpitaux 0769
- Alimentation et nutrition 0570
- Audiologie 0300
- Chimiothérapie 0992
- Dentisterie 0567
- Développement humain 0758
- Enseignement 0350
- Immunologie 0982
- Loisirs 0575
- Médecine du travail et thérapie 0354
- Médecine et chirurgie 0564
- Obstétrique et gynécologie 0380
- Ophtalmologie 0381
- Orthophonie 0460
- Pathologie 0571
- Pharmacie 0572
- Pharmacologie 0419
- Physiothérapie 0382
- Radiologie 0574
- Santé mentale 0347
- Santé publique 0573
- Soins infirmiers 0569
- Toxicologie 0383

SCIENCES PHYSIQUES

Sciences Pures

Chimie
- Généralités 0485
- Biochimie 487
- Chimie agricole 0749
- Chimie analytique 0486
- Chimie minérale 0488
- Chimie nucléaire 0738
- Chimie organique 0490
- Chimie pharmaceutique 0491
- Physique 0494
- PolymÇres 0495
- Radiation 0754

Mathématiques 0405
Physique
- Généralités 0605
- Acoustique 0986
- Astronomie et astrophysique 0606
- Electronique et électricité 0607
- Fluides et plasma 0759
- Météorologie 0608
- Optique 0752
- Particules (Physique nucléaire) 0798
- Physique atomique 0748
- Physique de l'état solide 0611
- Physique moléculaire 0609
- Physique nucléaire 0610
- Radiation 0756

Statistiques 0463

Sciences Appliqués Et Technologie

Informatique 0984
Ingénierie
- Généralités 0537
- Agricole 0539
- Automobile 0540
- Biomédicale 0541
- Chaleur et ther modynamique 0348
- Conditionnement (Emballage) 0549
- Génie aérospatial 0538
- Génie chimique 0542
- Génie civil 0543
- Génie électronique et électrique 0544
- Génie industriel 0546
- Génie mécanique 0548
- Génie nucléaire 0552
- Ingénierie des systâmes 0790
- Mécanique navale 0547
- Métallurgie 0743
- Science des matériaux 0794
- Technique du pétrole 0765
- Technique minière 0551
- Techniques sanitaires et municipales 0554
- Technologie hydraulique 0545

Mécanique appliquée 0346
Géotechnologie 0428
Matières plastiques (Technologie) 0795
Recherche opérationnelle 0796
Textiles et tissus (Technologie) 0794

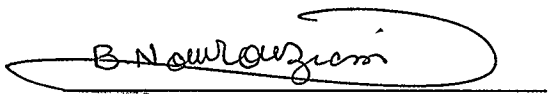
PSYCHOLOGIE

Généralités 0621
Personnalité 0625
Psychobiologie 0349
Psychologie clinique 0622
Psychologie du comportement 0384
Psychologie du développement 0620
Psychologie expérimentale 0623
Psychologie industrielle 0624
Psychologie physiologique 0989
Psychologie sociale 0451
Psychométrie 0632

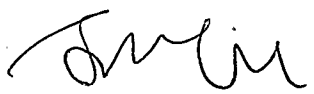
THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

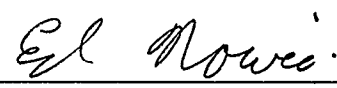
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Design and Bit-Serial Implementation of LDI Jaumann Digital Filters" submitted by Lorne Michael Smith in partial fulfillment of the requirements for the degree of Master of Science.



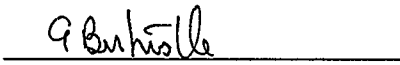
Supervisor, Dr. B. Nowrouzian
Department of Electrical and
Computer Engineering



Dr. J. Gu
Department of Electrical and
Computer Engineering



Dr. E. Nowicki
Department of Electrical and
Computer Engineering



Dr. G. Birtwistle
Department of Computer Science

Sept. 13, 1993
Date

ABSTRACT

This thesis presents the design and optimization of a multirate digital bandpass filter that satisfies the voice shaping requirements of the commercial digital CODECs, together with a corresponding field programmable gate array bit-serial implementation.

The above digital bandpass filter is designed to simultaneously satisfy a set of magnitude/frequency and group-delay/frequency response specifications subject to multiple equality and inequality constraints, and subject to ensuring an area efficient implementation. This bandpass filter is realized as a tandem connection of a 5th order lowpass and a 3rd order highpass digital filter, where the constituent digital filters themselves are realized as LDI Jaumann digital filters.

A min-max type optimization satisfaction routine is used to determine the required multiplier coefficient values. The digital filter implementation employs two's complement bit-serial arithmetic and a single multiplexed modified Booth multiplier architecture to ensure an area efficient realization.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. B. Nowrouzian, for his guidance and encouragement throughout the course of this research, and for his advice and constructive criticism offered during the writing of this thesis.

I would also like to thank the Alberta Microelectronic Centre for their financial support in terms of a graduate student scholarship, for permission to use their design facilities, and for the assistance offered by their staff.

I gratefully acknowledge the support provided by Micronet Network of Centres of Excellence, and the support provided by NSERC.

Finally, I would like to thank all of the staff within the Department of Electrical and Computer Engineering for their support and assistance.

In memory of my Dad,

Leigh Ellwood Smith,

and to

my family

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BP - bandpass

BS - bandstop

CODEC - coder-decoder

DSP - digital signal processing

FPGA - field programmable gate array

HP - highpass

IIR - infinite impulse response

LDI - lossless discrete integrator

LP - lowpass

LSB - least significant bit

LSW - least significant word

MSB - most significant bit

MSW - most significant word

SFG - signal flowgraph

SWL - signal wordlength

TC - two's complement

VLSI - very large scale integration

CHAPTER 1

INTRODUCTION

Digital signal processing (DSP) applications abound in the field of communications. These applications range from speech processing characterized by sample rates in the low kHz range, through to radar processors characterized by sample rates in the hundreds of MHz range. Other DSP applications include telecommunication, image processing, instrumentation, as well as biomedical, seismic, and geophysical data processing, to name just a few [1]-[7].

Digital filters play an important part in modern DSP. Such filters may be realized in many different forms, from a software program running on a general purpose computer in the case of a non real-time implementation (such as batch processing of seismic data), to a dedicated high-speed custom VLSI hardware implementation (real-time radar or image processing).

This thesis is concerned with the design and bit-serial implementation of a practical multirate lossless discrete integrator (LDI) [8] Jaumann [9] bandpass (BP) digital filter [10], [11]. This type of filter finds applications within the existing commercial digital coder-decoders (CODECs) used by the telecommunications industry [12]. In these applications, it is required that both speech signals and digital data be transmitted along the same communications channel. As such, the BP digital filter is required to satisfy a certain voice band magnitude/frequency response characteristic. Moreover, the phase or group-delay/frequency response characteristic is required to be such that it does not cause distortion in the transmitted digital data.

The above LDI Jaumann BP digital filter belongs to the category of infinite impulse response (IIR) digital filters. In IIR digital filters, the output signal $y(n)$ is related to the input signal $x(n)$ through a linear difference equation of the form

$$y(n) = \sum_{i=0}^{M} \alpha_i x(n-i) - \sum_{j=1}^{N} \beta_j y(n-j), \tag{1.1}$$

where n denotes the sampling instant, and where α_i and β_j are constant coefficients. In practice, the output signal samples in Eqn. (1.1) are evaluated using finite-precision arithmetic which leads inevitably to quantization errors [3]-[6], [13]-[15] in $y(n)$.

In the past, a variety of digital filter structures have been developed in an attempt to reduce the sensitivity of $y(n)$ to the above quantization errors. The resulting filter structures are based on the calculation of the output signal samples indirectly through the use of the transfer function of the filter. By applying the Z-transformation [3], Eqn. (1.1) can be recast in the form

$$Y(z) = \sum_{i=0}^{M} \alpha_i z^{-i} X(z) - \sum_{j=1}^{N} \beta_j z^{-j} Y(z), \tag{1.2}$$

where zero initial conditions have been assumed. Then, the transfer function of the digital filter is obtained as

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{i=0}^{M} \alpha_i z^{-i}}{\sum_{j=0}^{N} \beta_j z^{-j}}, \tag{1.3}$$

where $\beta_0 = 1$. The conventional digital filter structures for the realization of the transfer function $H(z)$ in Eqn. (1.3) include the direct form, the cascade form, the parallel form, and their combinations [3]. Many other classes of filter structures have been developed in

an ongoing attempt to achieve highly desirable properties such as low sensitivity to quantization errors and minimal hardware requirements. These include wave-digital [16] ladder [17] and lattice [18] filters, Gray and Markel digital lattice [19], as well as LDI ladder [20],[21] and lattice [22] digital filters.

Recently, a method was developed for the design of novel bilinear-LDI [23] Jaumann digital filters having Foster configurations [24]. This method was later extended to the exact design of LDI Jaumann digital filters having Cauer (leapfrog) and other configurations [9]. The resulting Jaumann digital filters have many practical features which make them attractive for high-quality high-performance applications in real-time DSP. In particular, they have the salient feature of exhibiting very low passband sensitivity to multiplier coefficient quantization errors. Moreover, they require the theoretical minimum number of multiply operations for the realization of lowpass (LP) and BP transfer functions of a given order, making a corresponding area-efficient implementation feasible [25].

The main contributions of this thesis include the introduction of a new structure for the realization of stable highpass (HP) and bandstop (BS) LDI Jaumann digital filters, a comprehensive discussion of the modified Booth multiplier, and the design and bit-serial implementation of a practical multirate BP LDI Jaumann digital filter which meets or exceeds the design requirements of the bandpass filter used within the commercial digital CODEC applications.

A brief overview of LDI Jaumann digital filters will be given in Chapter 2. Then, a new LDI Jaumann digital filter structure will be presented for the realization of HP and BS transfer functions. This is followed by a discussion of a gradient based min-max optimization routine [26] for the design of Jaumann digital filters capable of satisfying magnitude/

frequency and group-delay/frequency specifications simultaneously. The calculations of the required gradients is facilitated by the explicit expressions for the magnitude/frequency and group-delay/frequency response characteristics of IIR digital filters with respect to the constituent multiplier coefficients, together with explicit expressions for the corresponding derivatives presented.

Many DSP systems, particularly those used in speech processing, operate at relatively low sample rates, making a corresponding bit-serial implementation attractive [1]. Bit-serial architectures transmit digital signals represented as sequential successions of bits on single dedicated data paths as opposed to parallel architectures which transmit words of data on parallel buses. This leads to efficient communication within the bit-serial system in the form of reduced interconnection area and ease of routability. In addition, bit-serial arithmetic operations require less area intensive hardware cells for their implementation than their parallel counterparts. Bit-serial architectures seem to offer a better relationship between area and speed than traditional parallel architectures, and this has led to their extensive use in VLSI design [1].

Chapter 3 will begin with a review of bit-serial DSP systems including their corresponding hardware cells and the control signals required for operation. Then, the hardware allocation considerations necessary in bit-serial implementations will be discussed. Finally, an approach for the development of general order bit-serial LDI Jaumann digital filter architectures will be presented.

Many methods for realizing a multiplication operation in hardware have been studied in the past. These range from multipliers which multiply two positive binary numbers in a straightforward direct manner, to those which rely on complex algorithms to perform

two's complement (TC) multiplication operations [27]-[30]. The most commonly used TC bit-serial multiplier employs the modified Booth [28] recoding technique [29] due to the superior area and speed characteristics it offers.

In Chapter 4, the arithmetic operations of TC bit-serial addition, subtraction, and multiplication will be given together with their corresponding hardware cell implementations. The multiplication cells will include direct TC, Booth, and modified Booth variants. Then, a suitable method for the return of a rounded [31] product will be presented. Moreover, the shift register cells required to facilitate the operations of parallel-to-serial conversion, serial-in/serial-out data storage, downsampling, delay, state signal storage and update operations, and serial-to-parallel conversion will be given. Finally, the development of a bit-serial digital filter control signal generator will conclude the chapter.

In Chapter 5, the design and bit-serial field programmable gate array (FPGA) implementation of a multirate LDI Jaumann BP digital filter satisfying specifications similar to those required within the commercial digital CODECs is presented. The design employs a combination of a 5th order LP Jaumann digital filter operating at a sample frequency of 32 kHz, and a 3rd order HP Jaumann digital filter operating at a sample frequency of 8 kHz. The min-max optimization routine in Chapter 2 is applied to these Jaumann digital filters in order to obtain the constituent multiplier coefficient values. The bit-serial implementation of the resulting multirate digital filter uses the Actel 1.2μ FPGA technology. A single multiplexed modified Booth multiplier is employed within each of the constituent LP and HP Jaumann digital filters to ensure an area efficient implementation while achieving the required sample rate. The measured magnitude/frequency and group-delay/frequency response values will be compared to the corresponding theoretical characteristics.

Finally, the main conclusions of the thesis and suggestions for future research are summarized in Chapter 6.

CHAPTER 2

LDI JAUMANN DIGITAL FILTERS

This chapter will begin with a brief overview of Cauer-type LDI Jaumann digital filters suitable for the realization of LP and BP transfer functions. Then, a new structure is introduced for the realization of Cauer-type LDI Jaumann digital filters having HP and BS transfer functions. This is followed by a discussion of a gradient based min-max type optimization procedure which allows the design of LDI Jaumann digital filters satisfying general magnitude/frequency and group-delay/frequency specifications simultaneously. Explicit expressions are presented for the magnitude/frequency and group-delay/frequency response characteristics of IIR digital filters in terms of the constituent multiplier coefficients, together with explicit expressions for the corresponding derivatives for the calculation of the necessary gradients. These expressions are used within the software developed for the optimization of the corresponding Jaumann digital filters.

2.1 Cauer-Type LDI Jaumann Digital Filters

Two categories of design techniques for the synthesis of LDI Jaumann digital filter structures were proposed in [9], [22], and [24]. The first category proceeds by assuming the existence of a corresponding analog prototype filter having a voltage transfer function which is known to meet the required discrete-time frequency domain characteristics after a suitable transformation from the s to the z domain, where s denotes the continuous-time and z denotes the discrete-time frequency variable. The second category proceeds directly in terms of a z-domain description of the required transfer function, without any recourse to the concept of an analog prototype reference filter (or its transfer function).

The above LDI Jaumann digital filters exhibit low sensitivity to multiplier coefficient quantization errors and good dynamic range properties. Furthermore, their structure requires the theoretical minimum number of multipliers for the realization of LP and BP transfer functions of a given order. In addition, they possess a highly parallel structure which permits a fast two-cycle state update operation in which all even filter states are computed in the first cycle, and all odd filter states are computed in the second cycle. Because of these high-quality characteristics, a Cauer-type LDI Jaumann digital filter structure has been chosen as a candidate for the realization and bit-serial implementation of the multirate BP digital filter to be designed in this thesis.

The signal flow-graph (SFG) of an LDI Jaumann digital filter is shown in Fig. 2.1, where Z_1 and Z_2 are LDI reactances of order n_1 and n_2, respectively, and where *Output1*, and *Output2* are used to produce the desired transfer functions $H(z)$.

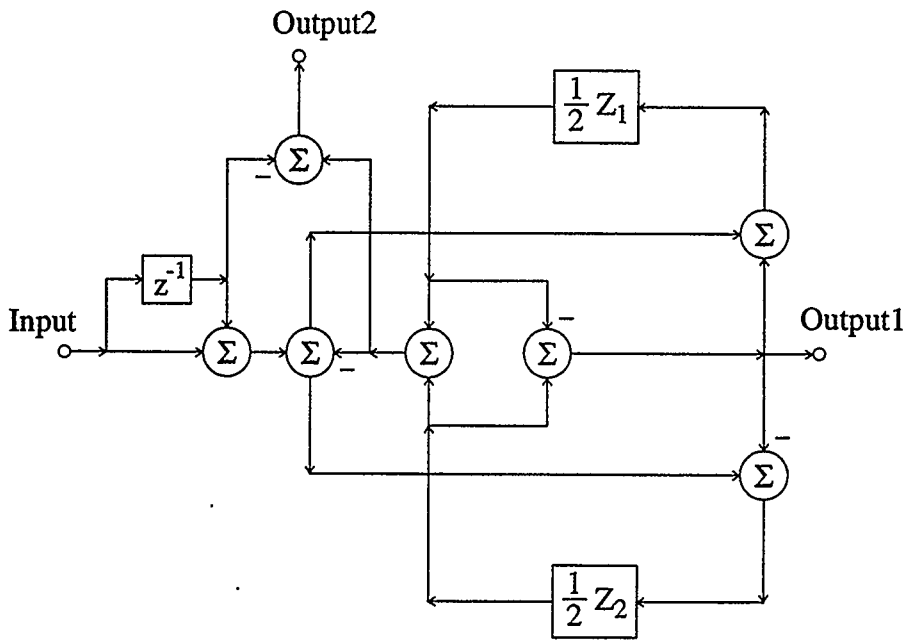


Fig. 2.1 LDI Jaumann Digital Filter Signal Flow-Graph

In the existing LDI Jaumann digital filter synthesis techniques [9],[24], the transfer function $H(z)$ is produced at *Output1* in accordance with

$$H(z) = z^{-1}(1+z)\left[\frac{1}{2(1+Z_1)} - \frac{1}{2(1+Z_2)}\right], \tag{2.1}$$

where the term $1 + z$ is a constituent of these techniques (arising from source precompensation [23], [32]). Clearly, if the transfer function $H(z)$ to be realized is a LP or a BP transfer function, then it contains a factor $1 + z$ producing the required transmission zero at $z = -1$. In this case, the factor $1 + z$ can be produced as the constituent of the LDI synthesis techniques. However, if the transfer function $H(z)$ to be realized is a HP or BS transfer function, then it does not contain the factor $1 + z$. In this case, the constituent of the LDI synthesis techniques must be eliminated internally through the creation of an extra pole at $z = -1$. Unfortunately, this additional pole increases the order of the resulting filter to be implemented to $n = n_1 + n_2 + 1$. Moreover, due to the non-ideal finite-precision arithmetic errors inherent in an actual implementation, the pole at $z = -1$ may move to a location slightly outside the unit circle in the z-plane thus making the digital filter unstable. In addition, any attempt to keep this pole within the unit circle will normally require highly accurate coefficient values (corresponding to long multiplier coefficient wordlengths). Finally, experience has shown that the internal signal wordlength (SWL) in this case turns out to be prohibitively long, rendering a corresponding implementation impractical.

In order to remedy the above problems, the fact is taken into account that LDI Jaumann digital filter structures are most suited to the realization of high-quality stable LP and BP transfer functions. Therefore, it is proposed that a desired HP transfer function $H_{HP}(z)$ be realized indirectly in terms of a corresponding LP transfer function $H_{LP}(z)$ in accordance with

$$H_{HP}(z) = H_{LP}(z) - 1, \tag{2.2}$$

where the transfer function $H_{LP}(z)$ can now be conveniently realized by a Jaumann digital filter. Similarly, it is proposed that a desired BS transfer function $H_{BS}(z)$ be realized indirectly in terms of a corresponding BP transfer function $H_{BP}(z)$ in accordance with

$$H_{BS}(z) = H_{BP}(z) - 1, \tag{2.3}$$

where the transfer function $H_{BP}(z)$ can now be conveniently realized by a Jaumann digital filter.

If the output signal is taken at *Output2* in Fig. 2.1, then the transfer function $H(z)$ may be obtained as

$$H(z) = z^{-1}\left[(1+z) \left(\frac{1}{2(1+Z_1)} + \frac{1}{2(1+Z_2)}\right) - 1 \right]. \tag{2.4}$$

This transfer function is of the exact form required in Eqn. (2.2) and Eqn. (2.3). The salient feature of this method for the realization of HP and BS transfer functions $H(z)$ is that no pole-zero cancellation is required at $z = -1$, resulting in a stable LDI Jaumann digital filter having one less multiplication operation and one less state storage register compared to the case when the transfer function is taken from *Output1*.

In accordance with the above discussions, the SFG of a general-order Cauer-type LDI Jaumann digital filter can be obtained as shown in Fig. 2.2, where the filter order is $n = n_1 + n_2$ not only for LP and HP, but also for BP and BS transfer functions. In this SFG, the even states are represented by X_{ij} for $i = 1, 2$ and $j = 2, 4, ..., \text{Even}\{n_i\}$, the odd states are represented by X_{ij} for $i = 1, 2$ and $j = 1, 3, ..., \text{Odd}\{n_i\}$, where $\text{Even}\{n_i\}$ denotes the largest even integer less than or equal to n_i, where $\text{Odd}\{n_i\}$ denotes the largest odd integer less than or equal to n_i, and where X_{10} and X_{20} represent intermediate signals.

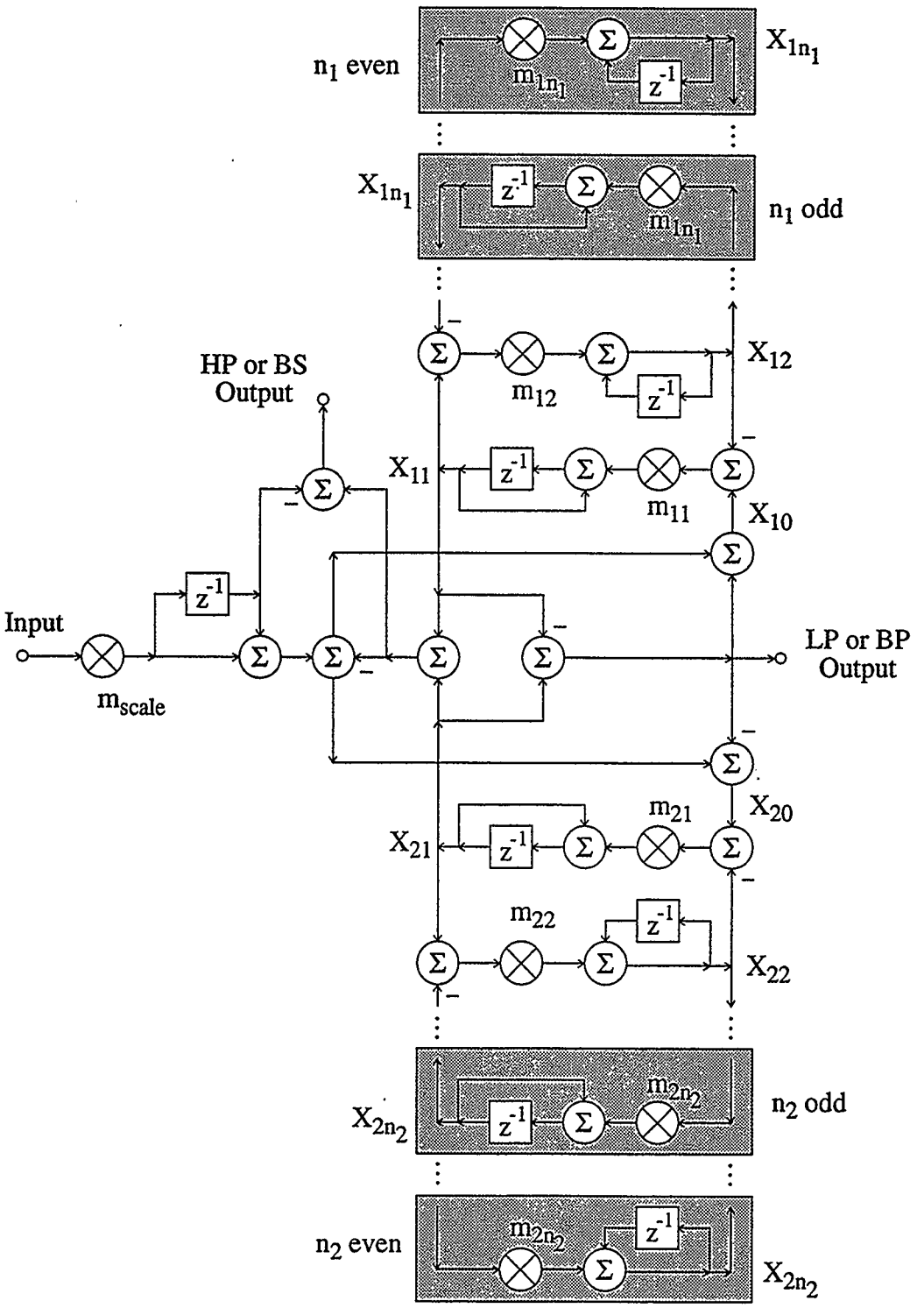


Fig. 2.2 General Order LDI Jaumann Digital Filter Signal Flow-Graph

2.2 Min-Max Constrained Optimization Satisfaction Procedure

The design and synthesis of LDI digital filters having ladder, lattice, and other practical configurations may be performed analytically in the realization of the classical transfer functions having Butterworth, Tschebyscheff, inverse Tschebyscheff, and elliptic magnitude/frequency response characteristics.

However, a general synthesis technique that allows the direct design of a digital filter realizing a transfer function which satisfies arbitrary magnitude/frequency and/or group-delay/frequency specifications is not available. In these situations, an optimization approach is usually adopted to satisfy these specifications. Moreover, only certain transfer functions are realizable as LDI Jaumann digital filters. Therefore, any optimization of a Jaumann digital filter must be applied to the filter structure rather than its transfer function. This ensures the realizability of the optimized result, and corresponds to the optimization of the constituent multiplier coefficient values.

Let the frequency response of the digital filter to be designed be represented as

$$H(e^{j\Omega}) = M(\underline{x}, \Omega)\, e^{j\phi(\underline{x}, \Omega)} \tag{2.5}$$

where

$$M(\underline{x}, \Omega) = \left| H(e^{j\Omega}) \right| \tag{2.6}$$

represents the magnitude/frequency response,

$$\phi(\underline{x}, \Omega) = Arg\{H(e^{j\Omega})\} \tag{2.7}$$

represents the phase/frequency response, and

$$\tau(\underline{x}, \Omega) = -\frac{1}{f_s}\frac{d\phi(\underline{x}, \Omega)}{d\Omega} \tag{2.8}$$

represents the absolute group-delay/frequency response of the filter. In these formulations,

$\Omega = 2\pi f/f_s$ represents the normalized real frequency-variable, $\underline{x}$ represents the vector of the constituent multiplier coefficients, and f_s represents the sampling rate.

In the most general situations, the magnitude/frequency response is required to fall within a certain tolerance region characterized by a lower and an upper bound, in addition to satisfying certain equality and/or inequality constraints. These design requirements can be recast into a min-max type optimization satisfaction problem [26] as follows:

$$\begin{aligned} &\text{minimize} \quad E = \text{MAX}\,\{e_i \,|\, i \in I\} \\ &\text{subject to} \quad G = \text{MAX}\,\{g_j \,|\, j \in J\} \le 0 \end{aligned} \tag{2.9}$$

where E represents the overall objective function and G represents the overall constraint.

In this thesis, the error components e_i take either the form

$$e_i = k_i M(\Omega_i) + k'_i \tag{2.10}$$

which represent the magnitude response errors at specific frequencies Ω_i, or the form

$$e_i = \underset{0 \le \Omega \le f_s/2}{\text{Max}} \begin{cases} \dfrac{M_L(\Omega) - M(\underline{x},\Omega)}{M_L(\Omega)} & \text{if } M(\underline{x},\Omega) \le \dfrac{M_L(\Omega) + M_U(\Omega)}{2} \\ \dfrac{M(\underline{x},\Omega) - M_U(\Omega)}{M_U(\Omega)} & \text{if } M(\underline{x},\Omega) \ge \dfrac{M_L(\Omega) + M_U(\Omega)}{2} \end{cases} \tag{2.11}$$

which represent the magnitude response errors throughout the tolerance region. Here, k_i and k'_i are constants, and $M_L(\Omega)$ and $M_U(\Omega)$ represent the lower and upper bounds of the prescribed magnitude/frequency response tolerance region. Similarly, the error components g_j can take on either the form given in Eqn. (2.10), or the form

$$g_j = k_j \tau(\Omega_j) - k'_j, \tag{2.12}$$

which represent the absolute group-delay errors at specific frequencies Ω_i. In this way, an equality constraint of the form

$$k_i M(\Omega_i) + k'_i = 0 \tag{2.13}$$

can be effectively handled by forming an error component e_i given by

$$e_i = k_i M(\Omega_i) - k'_i, \tag{2.14}$$

in the specification of the overall objective function E, together with an inequality constraint given by

$$g_j = k_j \tau(\Omega_j) - k'_j \leq 0, \tag{2.15}$$

in the specification of the overall constraint G.

In the design process advocated in this thesis, the initial step in the optimization is to select an LDI Jaumann digital filter of appropriate order which approximately satisfies the desired magnitude/frequency response specifications. Then, the above optimization procedure is used to improve on the satisfaction of the magnitude/frequency response specifications while at the same time attempting to satisfy the additional equality and inequality constraints on the magnitude and group-delay response characteristics.

2.3 Magnitude and Group-Delay Expressions

The optimization procedure discussed in the previous section requires the gradient of the magnitude/frequency and group-delay/frequency response with respect to the multiplier coefficient values of the LDI Jaumann digital filter. Explicit expressions for the calculation of these gradients are presented below.

As discussed in Chapter 1, the transfer function of an IIR digital filter is of the general form

$$H(z) = \frac{\sum_{i=0}^{m} \alpha_i z^{-i}}{\sum_{i=0}^{n} \beta_i z^{-i}}. \tag{2.16}$$

By setting $z = e^{j\Omega}$, and using the magnitude squared function

$$M^2(\Omega) = H(e^{j\Omega}) H^*(e^{j\Omega}), \tag{2.17}$$

the magnitude/frequency response of the digital filter can be obtained as

$$M(\Omega) = \sqrt{\frac{\sum_{i=0}^{m} \lambda_i \cos(i\Omega)}{\sum_{i=0}^{n} \sigma_i \cos(i\Omega)}}, \tag{2.18}$$

where $H^*(e^{j\Omega})$ represents the complex conjugate of $H(e^{j\Omega})$, and where

$$\lambda_0 = \sum_{j=0}^{m} \alpha_j^2, \qquad \lambda_i = 2\sum_{j=i}^{m} \alpha_j \alpha_{j-i} \qquad i = 1,2,...,m, \tag{2.19}$$

and

$$\sigma_0 = \sum_{j=0}^{n} \beta_j^2, \qquad \sigma_i = 2\sum_{j=i}^{n} \beta_j \beta_{j-i} \qquad i = 1,2,...,n. \tag{2.20}$$

From Eqn. (2.18), the square of the magnitude/frequency response can be expressed as

$$M^2(\Omega) = \frac{M_N(\Omega)}{M_D(\Omega)}, \tag{2.21}$$

where

$$M_N(\Omega) = \sum_{i=0}^{m} \lambda_i \cos(i\Omega) \quad \text{and} \quad M_D(\Omega) = \sum_{i=0}^{n} \sigma_i \cos(i\Omega). \tag{2.22}$$

Then, the derivative of the magnitude/frequency response with respect to the constituent

multiplier coefficients m_p may be obtained in accordance with

$$\frac{dM(\Omega)}{dm_p} = \frac{1}{2M_D(\Omega)}\left[\frac{1}{M(\Omega)}\frac{dM_N(\Omega)}{dm_p} - M(\Omega)\frac{dM_D(\Omega)}{dm_p}\right] \tag{2.23}$$

where

$$\frac{dM_N(\Omega)}{dm_p} = 2\sum_{i=0}^{m}\cos(i\Omega)\left[\begin{array}{ll}\displaystyle\sum_{j=0}^{m}\alpha_j\frac{d\alpha_j}{dm_p}, & i=0 \\ \displaystyle\sum_{j=i}^{m}\left(\alpha_j\frac{d\alpha_{j-i}}{dm_p}+\alpha_{j-i}\frac{d\alpha_j}{dm_p}\right), & i=1,2,\ldots,m\end{array}\right] \tag{2.24}$$

and

$$\frac{dM_D(\Omega)}{dm_p} = 2\sum_{i=0}^{n}\cos(i\Omega)\left[\begin{array}{ll}\displaystyle\sum_{j=0}^{n}\beta_j\frac{d\beta_j}{dm_p}, & i=0 \\ \displaystyle\sum_{j=i}^{n}\left(\beta_j\frac{d\beta_{j-i}}{dm_p}+\beta_{j-i}\frac{d\beta_j}{dm_p}\right), & i=1,2,\ldots,n\end{array}\right]. \tag{2.25}$$

The phase/frequency response associated with the transfer function $H(z)$ in Eqn. (2.16) can be obtained in accordance with

$$\phi(\Omega) = \phi_N(\Omega) - \phi_D(\Omega) \tag{2.26}$$

where

$$\phi_N(\Omega) = \operatorname{atan}\left[\frac{\displaystyle\sum_{i=0}^{m}\alpha_i\sin(i\Omega)}{\displaystyle\sum_{i=0}^{m}\alpha_i\cos(i\Omega)}\right] \tag{2.27}$$

represents the phase of the numerator, and where

$$\phi_D(\Omega) = \text{atan}\left[\frac{\sum_{j=0}^{n} \beta_j \sin(j\Omega)}{\sum_{j=0}^{n} \beta_j \cos(j\Omega)}\right] \tag{2.28}$$

represents the phase of the denominator. The corresponding group-delay is given by

$$\begin{aligned}\tau(\Omega) &= -\frac{1}{f_s}\frac{d\phi(\Omega)}{d\Omega} = -\frac{1}{f_s}\left[\frac{d\phi_N(\Omega)}{d\Omega} - \frac{d\phi_D(\Omega)}{d\Omega}\right] \\ &= -\frac{1}{f_s}\left[\tau_N(\Omega) - \tau_D(\Omega)\right]\end{aligned} \tag{2.29}$$

where

$$\tau_N(\Omega) = \frac{\sum_{i=1}^{m} i\alpha_i^2 + \sum_{i=0}^{m-1}\sum_{j=2i+1}^{m+i} j\alpha_i\alpha_{j-i}\cos\left[(j-2i)\,\Omega\right]}{\sum_{i=0}^{m}\alpha_i^2 + 2\sum_{i=0}^{m-1}\sum_{j=i+1}^{m}\alpha_i\alpha_j\cos\left[(j-i)\,\Omega\right]} = \frac{\tau_{NN}(\Omega)}{\tau_{ND}(\Omega)} \tag{2.30}$$

and

$$\tau_D(\Omega) = \frac{\sum_{i=1}^{n} i\beta_i^2 + \sum_{i=0}^{n-1}\sum_{j=2i+1}^{n+i} j\beta_i\beta_{j-i}\cos\left[(j-2i)\,\Omega\right]}{\sum_{i=0}^{n}\beta_i^2 + 2\sum_{i=0}^{n-1}\sum_{j=i+1}^{n}\beta_i\beta_j\cos\left[(j-i)\,\Omega\right]} = \frac{\tau_{DN}(\Omega)}{\tau_{DD}(\Omega)}. \tag{2.31}$$

Then, if the group-delay is represented as

$$\tau(\Omega) = -\frac{1}{f_s}\left[\frac{\tau_{NN}(\Omega)}{\tau_{ND}(\Omega)} - \frac{\tau_{DN}(\Omega)}{\tau_{DD}(\Omega)}\right], \tag{2.32}$$

the derivative of the group delay with respect to the individual multiplier coefficients may be obtained from

$$\frac{d\tau(\Omega)}{dm_p} = -\frac{1}{f_s \tau_{ND}(\Omega)}\left[\frac{d\tau_{NN}(\Omega)}{dm_p} - \tau_N(\Omega)\frac{d\tau_{ND}(\Omega)}{dm_p}\right] + \frac{1}{f_s \tau_{DD}(\Omega)}\left[\frac{d\tau_{DN}(\Omega)}{dm_p} - \tau_D(\Omega)\frac{d\tau_{DD}(\Omega)}{dm_p}\right] \tag{2.33}$$

where

$$\frac{d\tau_{NN}(\Omega)}{dm_p} = 2\sum_{i=1}^{m} i\alpha_i\left(\frac{d\alpha_i}{dm_p}\right) + \sum_{i=0}^{m-1}\sum_{j=2i+1}^{m+i} j\cos\left[(j-2i)\,\Omega\right]\left(\alpha_i\frac{d\alpha_{j-i}}{dm_p} + \alpha_{j-i}\frac{d\alpha_i}{dm_p}\right) \tag{2.34}$$

$$\frac{d\tau_{ND}(\Omega)}{dm_p} = 2\sum_{i=1}^{m}\alpha_i\frac{d\alpha_i}{dm_p} + 2\sum_{i=0}^{m-1}\sum_{j=i+1}^{m} j\cos\left[(j-2i)\,\Omega\right]\left(\alpha_i\frac{d\alpha_j}{dm_p} + \alpha_j\frac{d\alpha_i}{dm_p}\right) \tag{2.35}$$

and

$$\frac{d\tau_{DN}(\Omega)}{dm_p} = 2\sum_{i=1}^{n} i\beta_i\frac{d\beta_i}{dm_p} + \sum_{i=0}^{n-1}\sum_{j=2i+1}^{n+i} j\cos\left[(j-2i)\,\Omega\right]\left(\beta_i\frac{d\beta_{j-i}}{dm_p} + \beta_{j-i}\frac{d\beta_i}{dm_p}\right) \tag{2.36}$$

$$\frac{d\tau_{DD}(\Omega)}{dm_p} = 2\sum_{i=1}^{n}\beta_i\frac{d\beta_i}{dm_p} + 2\sum_{i=0}^{m-1}\sum_{j=i+1}^{m} j\cos\left[(j-2i)\,\Omega\right]\left(\beta_i\frac{d\beta_j}{dm_p} + \beta_j\frac{d\beta_i}{dm_p}\right). \tag{2.37}$$

By using the above expressions for the calculation of the required gradients, a software package [33] implementing the min-max procedure in [26] for the optimization of the transfer function coefficients has been modified and extended to the corresponding optimization of a multirate BP digital filter structure corresponding to a combination of a 5th order LP and a 3rd order HP LDI Jaumann digital filter.

2.4 Chapter Summary

A brief overview of LDI Jaumann digital filters has been given in this chapter. Jaumann digital filters have been classified according to whether they realize LP and BP, or HP and BS transfer functions. The high-quality characteristics of the existing LP or BP Jaumann digital filter structures have been discussed, and new HP or BS Jaumann digital filter structures have been proposed. The resulting HP or BS Jaumann digital filters have the same high-quality characteristics as their LP or BP counterparts and can realize highly stable transfer functions.

The main important features of a constrained min-max optimization method have been discussed together with its application to the design of digital filters satisfying simultaneous magnitude/frequency and/or group-delay/frequency specifications. This optimization method can be applied to the LDI Jaumann digital filters to determine the constituent multiplier coefficient values. Explicit expressions have been presented for obtaining the magnitude/frequency and group-delay/frequency response of IIR digital filters, together with explicit expressions for the derivatives of magnitude/frequency and group-delay/frequency response with respect to the constituent multiplier coefficients.

CHAPTER 3

BIT-SERIAL ARCHITECTURES FOR LDI JAUMANN DIGITAL FILTERS

This chapter presents bit-serial architectures for the implementation of the LDI Jaumann digital filter structures introduced in Chapter 2.

An overview of bit-serial DSP will be given together with a characterization of the corresponding control signals. This will be followed by a brief discussion of the most important factors that should be taken into account in the implementation of bit-serial digital filters. These results will then be applied to the development of bit-serial architectures for general-order LDI Jaumann digital filters.

3.1 Bit-Serial Digital Signal Processing

A bit-serial DSP system is an interconnected network of hardware cells, with the interconnections being dedicated bit-wide data paths which transmit and receive serial digital signals at a rate synchronized with the system bit-clock. These bit-wide data paths occupy a small amount of interconnection area and are easily routed, leading to efficient communications within the bit-serial DSP system. The high levels of pipelining achievable within these DSP systems lead to short critical-path delays which make possible the use of fast bit-clock rates.

The operations of bit-serial addition and multiplication are performed most naturally when the constituent digital signals are processed starting with the least significant bit (LSB) first. Moreover, by using fixed-point TC number format to represent these digital signals, the required bit-serial arithmetic cells can be implemented without any additional

hardware for sign correction. Due to the cyclical nature of the TC number system, the arithmetic sum operation leads to a correct final result even in the presence of intermediate overflows, provided that the final result is representable in the available wordlength. In addition, $-X$ may be obtained as $-X = \overline{X} + 1$, facilitating the hardware realization of the TC subtraction operation indirectly via an addition operation.

The hardware cells in a bit-serial system are of the general form shown in Fig. 3.1, where each cell is associated with one or more bit-wide input signals, and one or more corresponding output signals. Since the various digital signals flow within the system as continuous streams of bits, a method is required to identify the start of each new data word. This identification is achieved through the use of a control signal also provided as an input to the cell. This control signal is used to initiate or terminate an operation within the cell in synchronization with the arrival of the LSBs of the input signals. Throughout this thesis, the individual bits of a digital signal are said to be high or asserted if their logic values are 1, and low or unasserted if their logic values are 0.

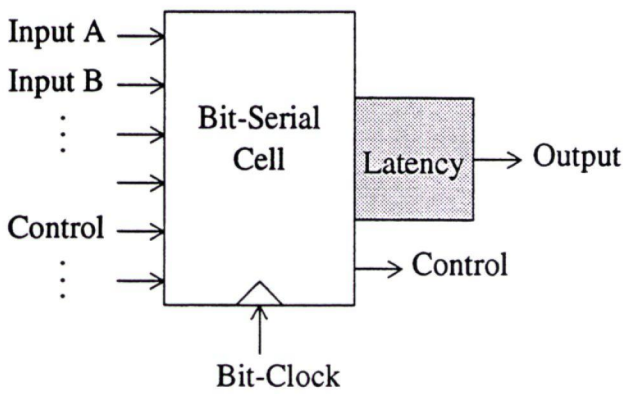


Fig. 3.1 Generic Bit-Serial Cell

A bit-serial hardware cell has a specific delay associated with it which is referred to as the latency of the cell. This latency is measured as an integral number of bit-clock periods, and signifies the time elapsed from the arrival of the LSBs of the input signals and the departure of the LSB of the output signal.

Digital filters targeted to bit-serial implementations on FPGAs require hardware cells including arithmetic cells such as adders, subtracters, and multipliers, memory cells such as shift registers and bit-delay chains, and signal selection cells such as multiplexors. The bit-serial hardware cells used in the development of LDI Jaumann digital filter architectures are shown in Fig. 3.2, where, for the sake of clarity, the associated bit-clock input is not indicated.

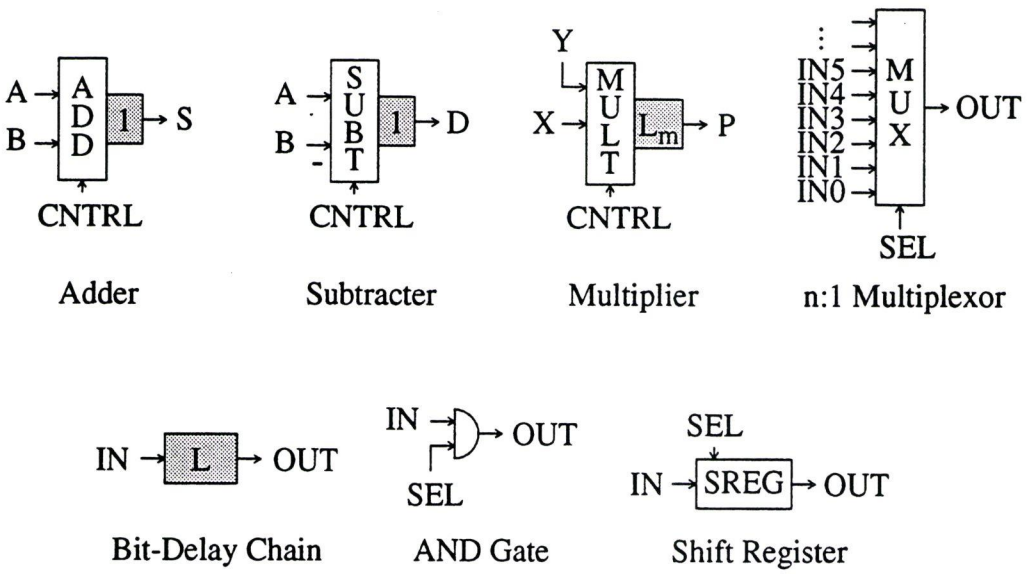


Fig. 3.2 Bit-Serial Cell Representations

The arithmetic operations of addition and subtraction are performed by the *ADD* and *SUBT* cells, respectively (the signal to be subtracted appears at the input labelled with a

minus sign), and the operation of multiplication is preformed by the *MULT* cell. Furthermore, shift register operations are performed by the *SREG* cell (when the control signal SEL is high, the data currently stored in the shift register is shifted out while the input data is shifted in). Finally, signal alignment operations are performed by bit-delay chains (where the latency is indicated by a number or expression within a shaded box), and signal selection operations are performed by the *MUX* cell and the *AND* gate.

3.2 Control Signals in Bit-Serial DSP Systems

In a bit-serial DSP system, a subsystem is required to control the timing of the constituent periodic chain of events. In addition, in an implementation employing multiplexed hardware cells, the control signals are required to direct the data flow along the correct data paths.

In the above DSP systems, a bit-clock control signal is used to generate all the other necessary control signals. This bit-clock control signal is used as a reference for the timing of all the events within the system. Moreover, the duration of any event is measured in terms of the bit-clock period.

In Fig. 3.3 the two basic forms of control signals are shown, namely the LSB pulse (c.f. *C0*, *C_0*) and the selection signal (c.f. *S0*, *S0_1*), where in the case of LDI Jaumann digital filters, the sample period is an integral multiple of the SWL period. In this figure, the sample period is 12 bit-clock periods, and the SWL period is 4 bit-clock periods.

There are two types of LSB pulse control signal, one of which is logic high for one bit-clock period during each sample period (represented by the notations *C0*, *C1*, ...), while the other is high for one bit-clock period during each SWL period (represented by the notations *C_0*, *C_1*, ...), where the numbers in these notations refer to the bit-clock

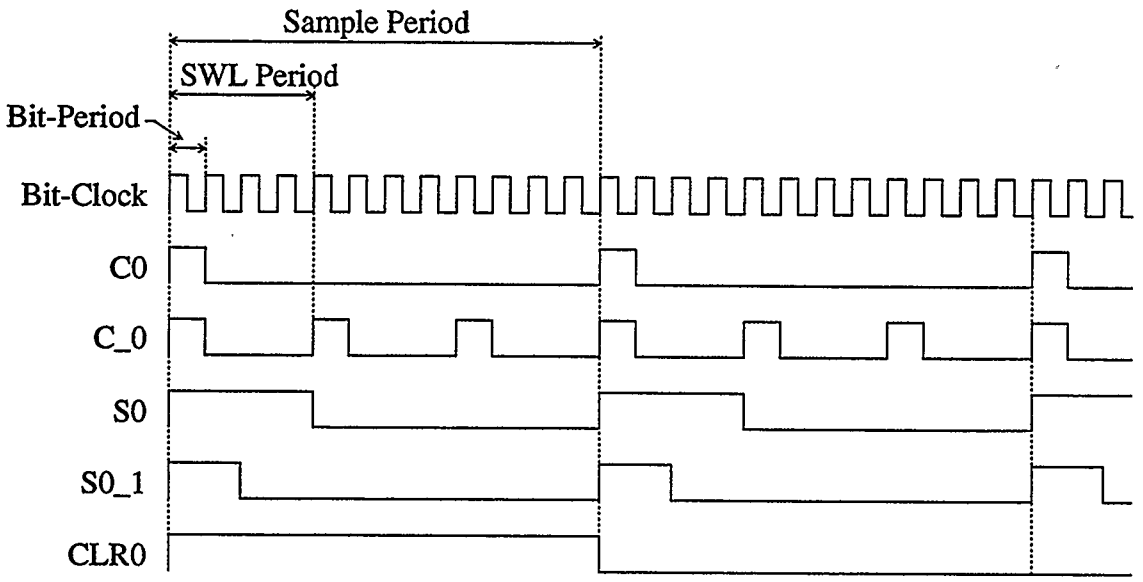


Fig. 3.3 Bit-Serial Control Signal Examples

period during which the respective signal is first asserted. In this way, each sample period will start at a relative time of t = 0, e.g. the control signal *C0* is asserted at t = 0 of the first sample period, at t = 0 of the second sample period, and so on. The primary use of these control signals is to force the bit-serial arithmetic cells to perform an operation once the LSB of a new signal-word has arrived, or to latch data into registers.

Similarly, there are two types of selection control signals, one of which is logic high for one SWL period during each sample period (represented by the notations *S0*, *S1*, ...), where the numbers in these notations refer to the bit-clock period during which the respective signal is first asserted. The other selection control signal is logic high for a certain portion of each sample period (represented by the notations *S0_1*, *S3_6*, ...), where the first number in the notation refers to the bit-clock period during which the respective signal is first asserted, and the second number indicates the bit-clock period after which the signal

becomes logic low.

A non-periodic control signal designated as *CLR0* is also required to facilitate the clearing of all data path storage registers at the start-up time. This control signal is asserted upon a system reset and will remain asserted for a duration of time no less than one sample period, after which it will become logic low until a future system reset.

In addition to the above control signals, which are generated by an on chip control generator cell, the multiplier hardware cell generates a control signal output. The multiplier receives an LSB pulse control signal and generates a corresponding LSB pulse control signal aligned with the LSB of the output product. The resulting control signal can in turn be used by the downstream hardware cells.

3.3 Hardware Allocation Considerations

The bit-serial multiplication hardware cell is typically more than an order of magnitude larger than the bit-serial addition and subtraction hardware cells. A comparison of the hardware requirements for these cells is illustrated in Table 3.1, where a modified Booth multiplier has been chosen for comparison (the multiplier coefficient wordlength m is either even or is made to be even through a sign-extension).

	Adder/Subtracter	Multiplier
Full Adders	1	$m/2$
D Flip-Flops	2	$13(m/2) + 2$
2:1 Multiplexors	0	$4(m/2) + 1$
Gates	1	$7(m/2) + 3$

Table 3.1 Hardware Requirement Comparison

Because of the hardware requirements associated with the above cells, the initial design

involves the consideration of the trade-offs amongst a number of factors. These factors include the number of physical multipliers N_m, the sample rate f_s, the bit-clock frequency f_{clk} (technology dependent), and the level of component multiplexing and its associated control hardware requirement.

In an n-th order LDI Jaumann digital filter

$$f_s = \frac{1}{W}\left\lceil \left(\frac{n}{N_m}\right) \right\rceil^{-1} f_{clk}, \tag{3.1}$$

where W represents the SWL in bits, and where $\lceil\ \rceil$ represents the "ceiling" function. The most viable choice in the implementation of bit-serial digital filters is to ensure that the constituent hardware multipliers are allocated such that they are never idle, which in the case of the corresponding LDI Jaumann digital filters can be achieved if $n/N_m \in \{1, 2, 3, \ldots\}$. In Jaumann digital filters, the minimum achievable sample period is constrained to be an integer multiple of W, with the minimum SWL obtained in accordance with

$$W_{min} = \mathrm{Max}\{W_l, W_s\}, \tag{3.2}$$

where W_l represents the minimum SWL as imposed by the latency within the digital filter structure and its associated data paths, and where W_s represents the minimum SWL as required by signal scaling constraints. The signal scaling constraints include overflow guard bits and round-off guard bits (see Sec. 5.2), as well as multiplier sign-extension guard bits (see Sec. 4.2.6). The format the SWL takes on is shown in Fig. 3.4. Furthermore, in such situations when W_s is less than W_l, one must ensure that the input data signal is positioned within the system word such that the scaling constraints remain satisfied.

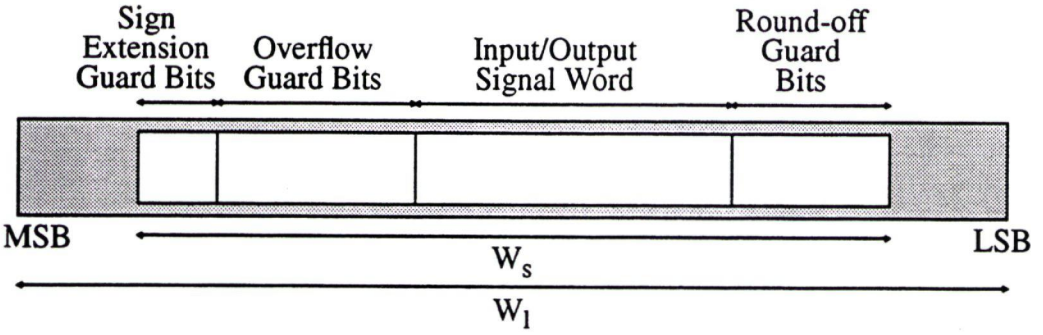


Fig. 3.4 Internal System Word

3.4 State Update Operations

The processing of each input sample by a digital filter involves a complete set of state update operations, where these update operations must be performed in a certain chronological order. Since LDI Jaumann digital filters are highly parallel structures, the only constraint on the state update operations is that any even state which contributes to the value of an odd state must be updated prior to updating the respective odd state, where the even and odd states were defined in Sec. 2.1. The pseudocode in Fig. 3.5 will generate a complete set of equations representing the required sequential state update operations, where the suffix *new* refers to the updated state value, and the suffix *old* refers to the state value prior to being updated.

In an implementation which performs two or more state update operations concurrently, the state update equations are assigned to multipliers and SWL periods according to the process shown in Fig. 3.6.

It should be pointed out that if a scaling multiplier is incorporated at the input to the digital filter, then the corresponding multiplication operation must be scheduled first.

Having scheduled the multipliers to specific multiplication operations and SWL peri-

ods, a bit-delay equalization must be performed on the digital filter. This equalization involves the insertion of delay-chains to ensure the proper bit-alignment of the internal signals. Although the SFG of a digital filter includes unit-delay operators in predefined

```
n = n1 + n2;
for (j = 2; j <= n; j = j + 2)
{
if (j < n1)
  Xnew1j = [Xold(j-1) - Xold1(j+1)] * m1j + Xold1j;
else if (j = n1)
  Xnew1j = [Xold1(j-1) - GND] * m1j + Xold1j;
if (j < n2)
  Xnew2j = [Xold2(j-1) - Xold2(j+1)] * m2j + Xold2j;
else if (j = n2)
  Xnew2j = [Xold2(j-1) - GND] * m2j + Xold2j;
if (j <= n1)
  Xnew1(j-1) = [Xnew1(j-2) - Xnew1j] * m1(j-1) + Xold1(j-1);
else if (j = n1+1)
  Xnew1(j-1) = [Xnew1(j-2) - GND] * m1(j-1) + Xold1(j-1);
if (j <= n2)
  Xnew2(j-1) = [Xnew2(j-2) - Xnew2j] * m2(j-1) + Xold2(j-1);
else if (j = n2+1)
  Xnew2(j-1) = [Xnew2(j-2) - GND] * m2(j-1) + Xold2(j-1);
}
```

Fig. 3.5 State Update Pseudocode

While any even state update equation is unassigned

Assign N_m even state update equations per SWL period (one to each of the N_m multipliers)

If the current SWL period does not have all multipliers assigned

Assign odd state update equations to the unassigned multipliers, subject to the previously discussed the state update constraint

While any odd state update equation is unassigned

Assign N_m odd state update equations per SWL period (one to each of the N_m multipliers)

Fig. 3.6 State Update Equation Assignments

locations, once the design is mapped to hardware the unit-delays are replaced by D-flip/ flops inserted at strategic locations within the filter. In addition, in systems incorporating multiplexed arithmetic hardware cells, certain signals are required more than once, and must be stored in recirculating registers such that their LSB's are available when needed.

3.5 Bit-Serial Architectures for LDI Jaumann Digital Filters

A fully pipelined bit-serial LDI Jaumann digital filter operates such that an input signal-bit enters the digital filter and an output signal-bit exits the digital filter during each bit-clock period. Such an implementation would entail a sample period equal to the SWL period. In this case, each addition, subtraction, multiplication, and unit-delay operation in the bit-serial LDI Jaumann digital filter SFG will have a corresponding hardware cell in the bit-serial implementation, with each hardware cell processing data continuously. The fully pipelined bit-serial LDI Jaumann digital filter architecture will be identified as the

$N_m = n$ case, since the number of SFG multiplications is the same as the filter order.

In situations when $n/N_m \in \{2, 3, \ldots\}$, some level of multiplier multiplexing is required. Due to the parallel nature of the LDI Jaumann digital filter, the most efficient implementations are those which ensure that each multiplier is multiplexed in such a manner that it is assigned to the same number of even and odd state update operations, with the one exception being the case when $N_m = 1$.

In the development of the bit-serial LDI Jaumann digital filter architectures, only the cases in which the constituent multipliers are fully pipelined will be considered. These cases include the architectures corresponding to the $N_m = n$ and $n/N_m \in \{2, 3, \ldots\}$ cases.

Case 1. $N_m = n$

In this case, all of the state update operations are performed concurrently. Then, each arithmetic operation in the LDI Jaumann digital filter SFG will map directly to a hardware cell in the corresponding bit-serial architecture as shown in Fig. 3.7, where the parenthesized terms in the control signal notations indicate the bit-clock period when the respective control signal is logic high. Furthermore, in the case where this architecture is augmented by a scaling multiplier, the multiplier within the dashed box is present.

The main consideration in the architecture shown in Fig. 3.7 is that of bit-delay equalization. The following discussion will be confined to the consideration of the LSBs of the various serial data signals, keeping in mind the fact that if the operations are performed correctly for the LSBs of these signals, they will be necessarily performed correctly for the remaining bits in these signals.

In the above architecture, starting at a reference time of t = 0, all of the odd state signals will have their LSBs available at the locations labelled by the odd state signal nota-

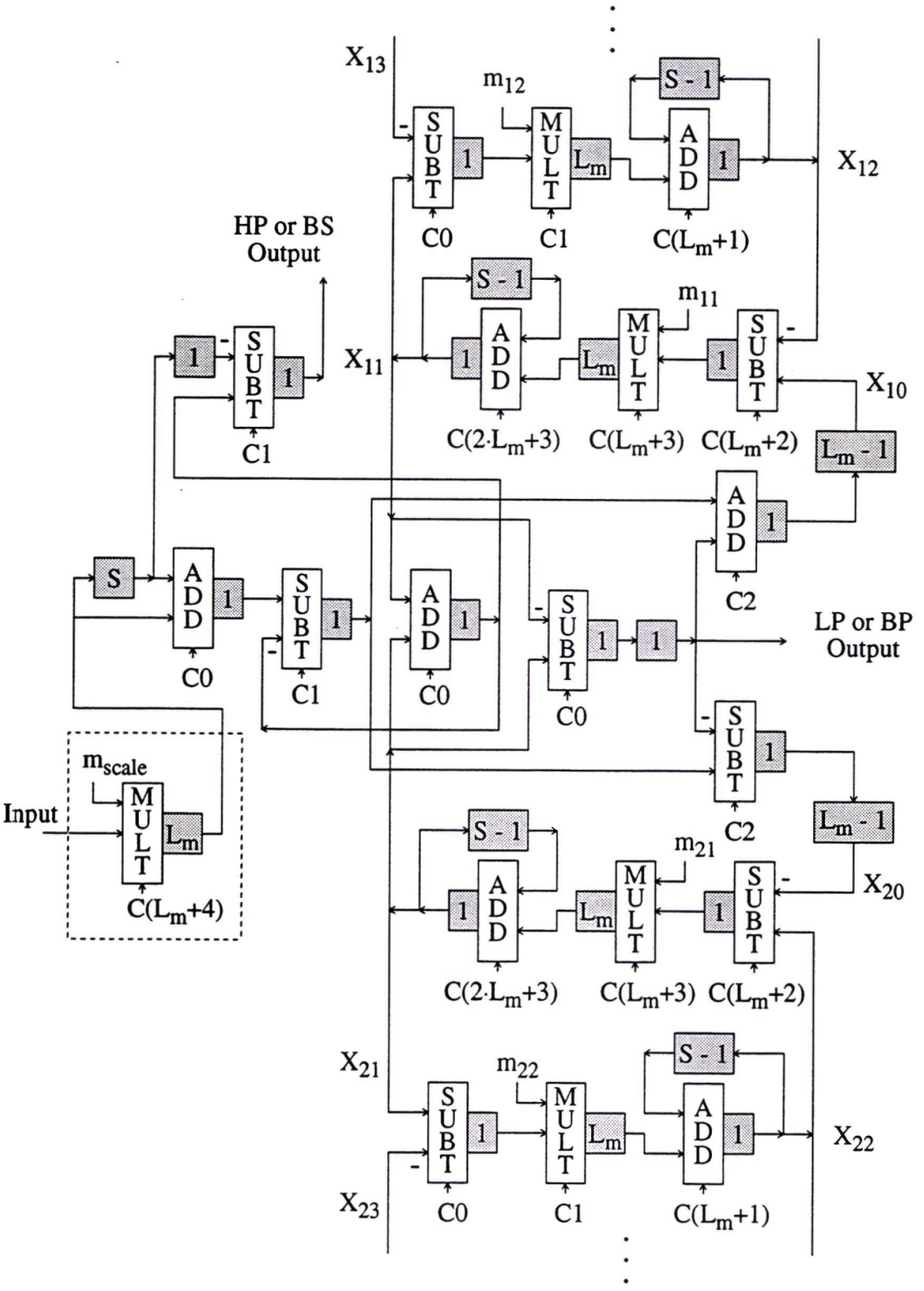


Fig. 3.7 $N_m = n$ Bit-Serial Architecture for LDI Jaumann Digital Filter

tions in the figure. Then, the state update operation

$$Xnew_{ij} = [Xold_{i(j-1)} - Xold_{i(j+1)}]\, m_{ij} + Xold_{ij}$$

is performed, where the LSB of the even state $Xnew_{ij}$ will be available following a latency of $L_m + 2$ (corresponding to the latency of one addition, one subtraction, and one multiplication operation). This latency represents the delay associated with the composite even filter state branch along which the LSB has travelled. Having the LSBs of the even state signals available at $t = L_m + 2$, the state update operation

$$Xnew_{ij} = [Xnew_{i(j-1)} - Xnew_{i(j+1)}]\, m_{ij} + Xold_{ij},$$

is performed, where the LSBs of the intermediate signals X_{10} and X_{20} are required to be available at $t = L_m + 2$. The update operation of the odd state signal $Xnew_{ij}$ will involve a latency of $L_m + 2$, resulting in the LSBs of the updated odd state signals being available at $t = 2 \cdot L_m + 4$. Since the LSB of any odd state signal has now travelled around a closed loop data path with a minimal amount of delay, the minimum SWL due to hardware and data path latency for an $N_m = n$ architecture is obtained as

$$W_l = 2 \cdot L_m + 4. \tag{3.3}$$

The input/output operations in the LDI Jaumann digital filter architecture are considered next. Since the LSBs of the odd state signals are available at the reference time $t = 0$, the LSB of the input signal (following the scaling multiplier) must also be available at this time. This will allow the output signal of the *ADD* cell at the filter input to be available at $t = 1$, and to enter the *SUBT* cell in alignment with the LSB of the output signal arriving from the *ADD* cell computing $X_{11} + X_{21}$. The LSB of the output signal from this *SUBT* cell then arrives at the *ADD* and *SUBT* cells at $t = 2$ to begin the computation of the signals X_{10} and X_{20}. By delaying the LSB output signal from the *SUBT* cell computing

$X_{21} - X_{11}$ by one bit-clock period, this signal will arrive at the *ADD* and *SUBT* cells computing X_{10} and X_{20} synchronized with the other input. Furthermore, the output signals X_{10} and X_{20} of these *ADD* and *SUBT* cells are available at t = 3 but are not required until t = $L_m + 2$ (a delay of $L_m - 1$ is then used to make this signal available at the time required). Finally, the LSBs of the LP or BP output signal and the HP or BS output signal are available at t = 2.

Although the architecture in Fig. 3.7 allows a fully pipelined bit-serial implementation, it involves a substantial cost associated with the SWL. With $W_l = 2 \cdot L_m + 4$, the minimum SWL requirement for the LDI Jaumann digital filter is usually dominated by W_l (c.f. Eqn. (3.2)). As an example, a Jaumann digital filter incorporating a modified Booth multiplier cell would incur a multiplication latency of $L_m = 3 \cdot m/2 + 1$, where m represents the number of multiplier coefficient bits. Even for a relatively short multiplier coefficient wordlength, W_l would typically be much greater than W_s (c.f. Eqn. (3.2)). Therefore, such an implementation would be practical only in situations incorporating short coefficient wordlengths and requiring high sample rates.

Case 2. $n/N_m \in \{2, 3, \ldots\}$

In this case, the nth order LDI Jaumann digital filter architecture will be designed as two separate modules, namely an input/output module and a multiplexed multiplier module performing the state update operations.

The input/output module is shown in Fig. 3.8. If the LDI Jaumann digital filter incorporates a scaling multiplier at its input, then $D = L_m + 1$ in the assertion times of the selection control signals and in the delay-chains. If the Jaumann digital filter does not incorporate a scaling multiplier at its input, then $D = 0$.

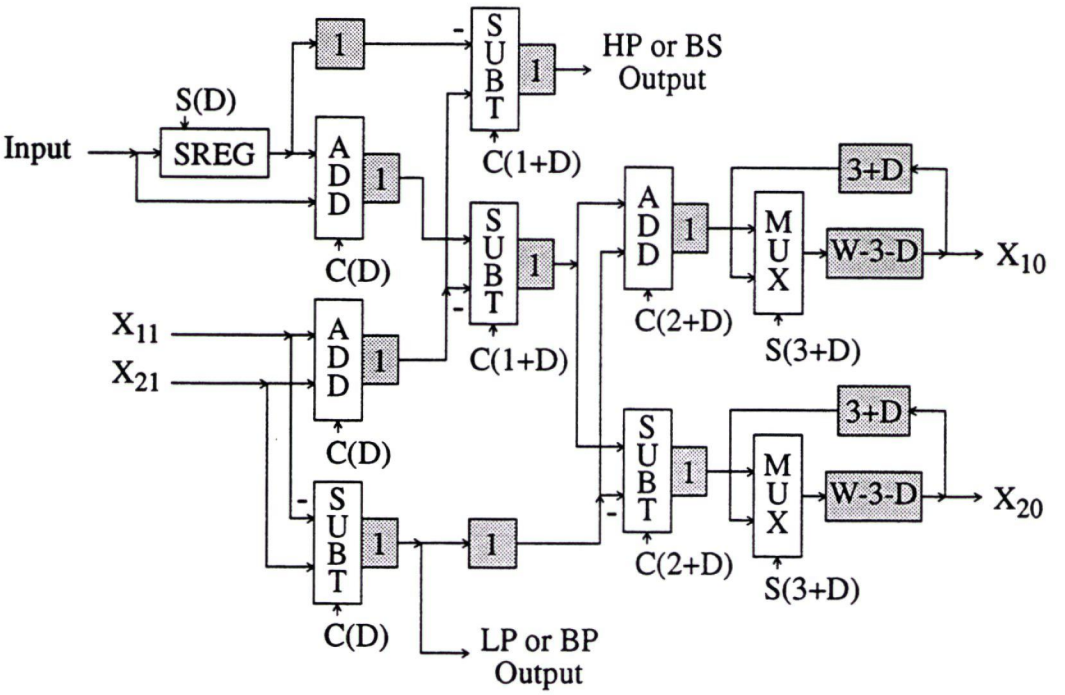


Fig. 3.8 Input/Output Module for Multiplexed Architecture

In the input/output module shown in Fig. 3.8, the input signal will arrive at a reference time of t = *D*, where *D* is as defined above. At this time, the shift register will receive a control signal S(*D*) causing the previous input signal sample currently stored in the shift register cell *SREG* to begin shifting out, while allowing the present signal sample to start shifting in. This process will continue for a time equal to the SWL period. Also at t = *D*, the LSBs of the odd states X_{11} and X_{21} will have arrived. The present input signal sample, the past signal sample, and the state signals X_{11} and X_{21} will be processed as shown in Fig. 3.8, to produce the LP or BP output signal (LSB at t = 1 + *D*), the HP or BS output signal (LSB at t = 2 + *D*), as well as the intermediate signals X_{10} and X_{20} (LSBs at t = *W*). Furthermore, these intermediate signals will recirculate in such a manner as to continually make their LSBs available at integral multiples of the SWL period

(i.e. at t = W, $2W$, $3W$, ...). In the case of the Jaumann digital filter incorporating a scaling multiplier, the input/output module shown in Fig. 3.8 will be receiving the actual scaled input signal *Input'*, as well as X'_{11} and X'_{21} (c.f. Fig. 3.9, Fig. 3.10) at the inputs labelled as *Input*, X_{11} and X_{21}.

The above mentioned multiplexed multiplier module is responsible for performing the state update operations as well as performing any scaling multiplication operations required. By selecting $n_1 = 2$ and $n_2 = 2$, the pseudocode in Fig. 3.5 can be used to generate the corresponding set of state update equations as

$$Xnew_{12} = [Xold_{11} - GND] * m_{12} + Xold_{12}$$

$$Xnew_{22} = [Xold_{21} - GND] * m_{22} + Xold_{22}$$

$$Xnew_{11} = [Xnew_{10} - Xnew_{12}] * m_{11} + Xold_{11}$$

$$Xnew_{21} = [Xnew_{20} - Xnew_{22}] * m_{21} + Xold_{21}.$$

Although the following discussion will deal with the assignments of these equations to the hardware, the same process applies equally well to a general-order LDI Jaumann digital filter.

Two examples of a multiplexed multiplier module are discussed, one of which employs one hardware multiplier, while the other employs two hardware multipliers.

Example 1: $N_m = 1$

A multiplexed multiplier module employing one hardware multiplier is shown in Fig. 3.9. In this module, each state signal update operation will occur during a processing interval whose length equals the SWL, starting at a reference time of t = 0. In this example, the state equations are assigned to SWL periods in the same order as generated by the pseudocode.

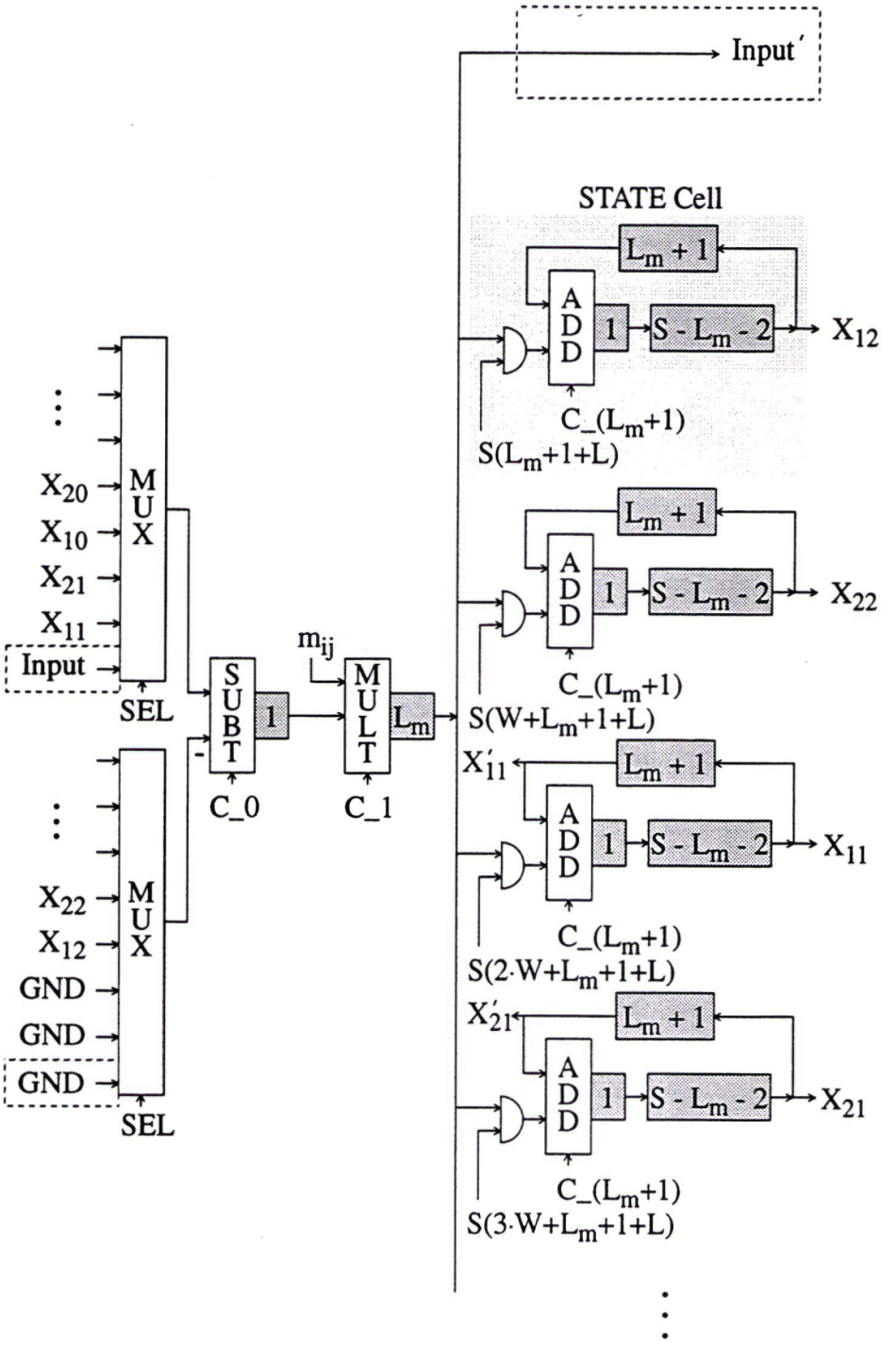


Fig. 3.9 $N_m = 1$ Multiplexed Multiplier Module

If the LDI Jaumann digital filter incorporates a scaling multiplier at its input, then the components in Fig. 3.9 within the dashed lines are present, and $L = W$ in the assertion times of the selection control signals. Moreover, during the processing interval $t \in [0,W)$ in the first SWL period, the top multiplexor will select the input signal, the bottom multiplexor will select GND, and the required multiplication by the scaling multiplier coefficient will be performed. The output signal will appear at Input' at $t = L_m + 1$, and it will then be used by the input/output module previously shown in Fig. 3.8. If the Jaumann digital filter does not incorporate a scaling multiplier at its input, then the components in Fig. 3.9 within the dashed lines are not present, and $L = 0$ in the assertion times of the selection control signals. The mapping of the state update equations to the architecture is discussed in the following.

The first state equation in the above list relates to updating the state signal X_{12} and requires the state signal X_{11}. During the processing interval $t \in [0+L,W+L)$ in the first SWL period, the signal X_{11} will be selected as the input to the top *MUX*, and the signal GND will be selected as the input to the bottom *MUX*, facilitating the operation $X_{11} - \text{GND}$. This will be followed by a multiplication by the coefficient m_{12}, with the resulting product entering a *STATE* cell (represented, e.g., by the lightly shaded box in Fig. 3.9), where it is added to the currently stored signal $Xold_{12}$ to obtain the signal $Xnew_{12}$. This updated signal will then have its LSB available at correct times as required by other state update operations. The remaining equations are each assigned to hardware in the same manner, with the second equation being assigned to the second SWL period, the third equation to the third SWL period, and so on.

All of the *STATE* cells receive the same initialization control pulse $C\_(L_m+1)$. This

ensures that during the bit-period immediately following a corresponding state update operation, the most significant carry-out is prevented from becoming a carry-in to the continuously recirculating registers. To allow the product of the multiplication to be added to the currently stored state values, the *AND* gate selection signals will be asserted at the times indicated. The *SEL* signal causes the selection of the 0th *MUX* input during the 0th SWL period, the selection of the 1st *MUX* input during the 1st SWL period, and so on, where *SEL* is provided through $\lceil log_2(n) \rceil$ selection lines.

Example 2: $n/N_m = 2$

A multiplexed multiplier module employing two multipliers (in addition to the scaling multiplier, if required) is shown in Fig. 3.10. In this module, two state signal update operations will occur during each SWL period, starting at a reference time of t = 0. In this case, the state update equations are assigned to SWL periods in accordance with the process given in Fig. 3.6.

If the LDI Jaumann digital filter incorporates a scaling multiplier at its input, then the components in Fig. 3.10 within the dashed lines are present. Moreover, during $t \in [0,S)$ in the first SWL period, the top multiplier will multiply the input signal by the scaling multiplier coefficient m_{scale}. The scaled signal will appear as *Input'* at $t = L_m + 1$, and it will then be used by the input/output module previously shown in Fig. 3.8. The mapping of the state update equations to the architecture is discussed below.

The first even state equation in the above list relates to updating the state signal X_{12} and requires the state signal X_{11}. During the processing interval $t \in [0,W)$ in the first SWL period, the signal X_{11} will be selected as the input to the top *MUX* associated with the top multiplier, and the signal GND will be selected as the input to the bottom *MUX* associated

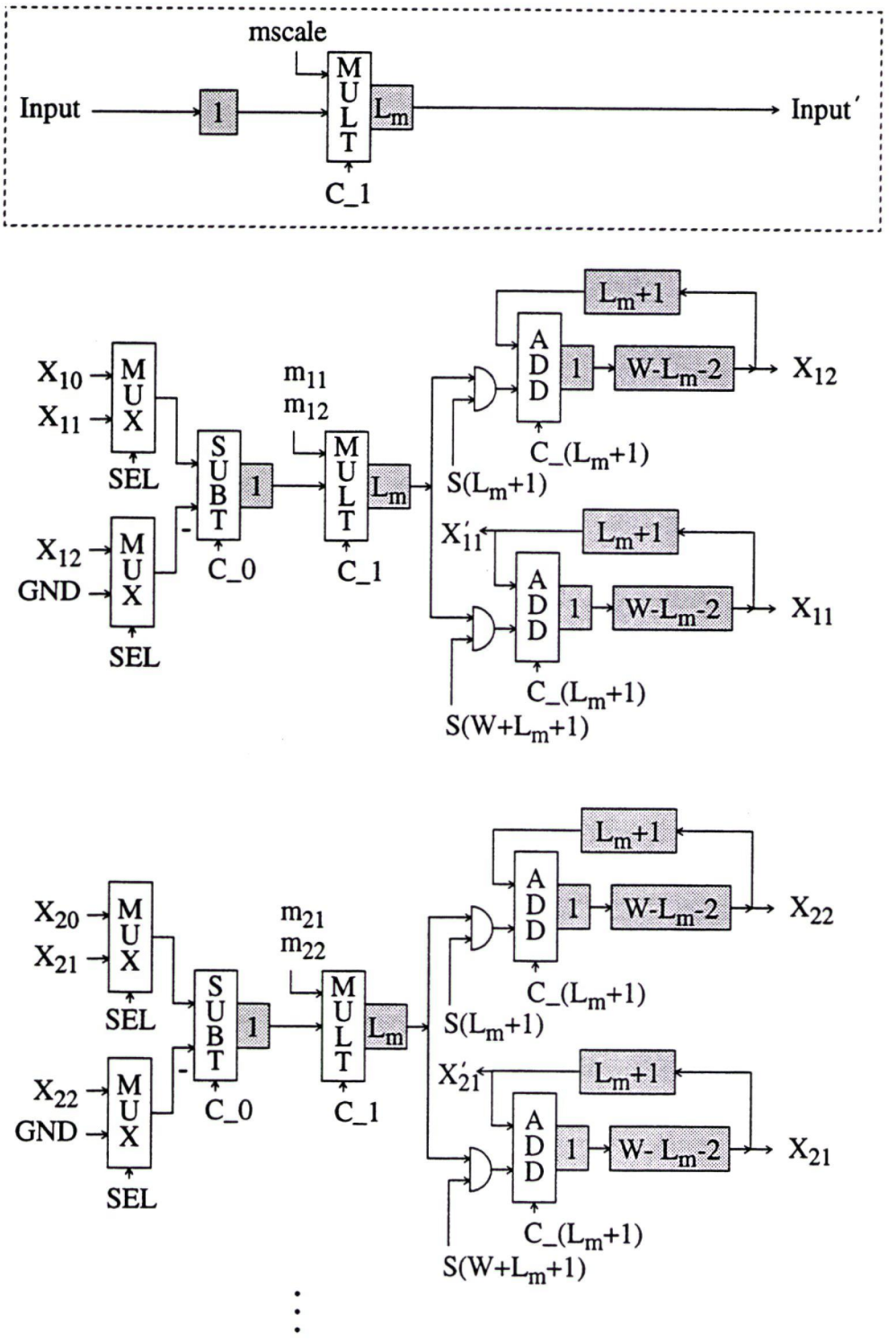


Fig. 3.10 $n/N_m = 2$ Multiplexed Multiplier Module

with the top multiplier, facilitating the operation $X_{11} - \mathrm{GND}$. This will be followed by a multiplication by the coefficient m_{12}, with the resulting product entering a *STATE* cell where it is added to the currently stored signal $Xold_{12}$ to obtain the signal $Xnew_{12}$. This updated signal will then have its LSB available at the correct times as required by other state update operations. Similarly, the second even state equation in the list relates to updating the state signal X_{22} and requires the state signal X_{21}. The signal X_{21} will be selected as the input to the top *MUX* associated with the bottom multiplier, and the signal GND will be selected as the input to the bottom *MUX* associated with the bottom multiplier, facilitating the operation $X_{21} - \mathrm{GND}$. This will be followed with a multiplication by the coefficient m_{22}, with the resulting product entering a *STATE* cell where it is added to the currently stored signal $Xold_{22}$ to obtain the signal $Xnew_{22}$. This updated signal will then have its LSB available at correct times as required by other state update operations.

Having assigned the even state equations to specific multipliers and SWL periods, the odd state equations can then be assigned to hardware in a similar manner. The first odd state equation in the above list relates to updating the state signal X_{11} and requires the signals X_{10} and X_{12} which were computed during the first SWL period. During the processing interval $t \in [W,2W)$ in the second SWL period, the signal X_{10} will be selected as the input to the top *MUX* associated with the top multiplier, and the signal X_{12} will be the selected as the input to the bottom *MUX* associated with the top multiplier, facilitating the operation $X_{10} - X_{12}$. This will be followed by a multiplication by the coefficient m_{11}, with the resulting product entering a *STATE* cell where it is added to the currently stored signal $Xold_{11}$ to obtain the signal $Xnew_{11}$. This updated signal will then have its LSB available at the correct times as required by other state update operations. Similarly, the second odd

state equation in the list relates to updating the state signal X_{22} and requires the signals X_{20} and X_{22}. The signal X_{20} will be selected as the input to the top *MUX* associated with the bottom multiplier, and the signal X_{22} will be selected as the input to the bottom *MUX* associated with the bottom multiplier, facilitating the operation $X_{20} - X_{22}$. This will be followed with a multiplication by the coefficient m_{21}, with the resulting product entering a *STATE* cell where it is added to the currently stored signal $Xold_{22}$ to obtain the signal $Xnew_{22}$. This updated signal will then have its LSB available at correct times as required by other state update operations.

3.6 Chapter Summary

This chapter has reviewed various concepts associated with the development of bit-serial DSP architectures. The general form of a bit-serial hardware cell has been reviewed together with the various forms of the required control signals. The main considerations to take into account in the development of these DSP architectures have been discussed in connection with digital filter SFGs. These included the trade-offs between the required sample rate and the number of physical hardware multipliers employed, as well as the SWL and the level of hardware cell multiplexing. Finally, an approach for realizing a general-order LDI Jaumann digital filter as a bit-serial architecture has been presented. This allows practical LDI Jaumann digital filters to be realized in a corresponding bit-serial architecture in a straightforward manner. Such an architecture will be adopted for the realization of a corresponding multirate BP digital filter as discussed in Chapter 5.

CHAPTER 4

BIT-SERIAL HARDWARE CELLS FOR THE IMPLEMENTATION OF LDI JAUMANN DIGITAL FILTERS

The FPGA implementation of the bit-serial LDI Jaumann digital filter architectures presented in the previous chapter requires the development of a corresponding set of gate-level hardware cells. In this chapter, the arithmetic operations of TC bit-serial addition, subtraction, and multiplication will be presented together with their corresponding hardware cell implementations. In addition, the shift register cells required to facilitate the operations of parallel-to-serial conversion, serial-in/serial-out data storage, downsampling, delay, state storage and update, and serial-to-parallel conversion will be given. The chapter will conclude with the development of a bit-serial digital filter control signal generator.

4.1 Two's Complement Bit-Serial Addition and Subtraction

This section is concerned with the development of dedicated hardware cells for the operations of TC bit-serial addition and subtraction, and a hardware cell allowing selectable addition/subtraction operations.

4.1.1 Bit-Serial Addition and Subtraction

Let A and B denote two n-bit TC binary numbers represented as $A = a_{n-1}a_{n-2}\ldots a_1a_0$ and $B = b_{n-1}b_{n-2}\ldots b_1b_0$. Then, if both A and B are transmitted as bit-serial data, the process of forming the sum $S = A + B$ is known as bit-serial

addition, and the process of forming the difference $D = A - B$ is known as bit-serial subtraction. Furthermore, the difference D may be obtained as $D = A + (-B)$, where in the TC binary number system, $-B = \overline{B} + 1$ with the overbar representing complement.

Bit-serial addition and subtraction operations are performed in a bit-wise manner from the LSB to the most significant bit (MSB). The addition of the i-th bits of A and B results in a two-bit number given by $cout_i s_i = a_i + b_i + cin_i$, where s_i is the i-th bit of the sum S, and cin_i and $cout_i$ are the corresponding carry-in and carry-out bits, respectively. Similarly, the subtraction of the i-th bits of A and B results in a two-bit number given by $cout_i d_i = a_i + \overline{b}_i + cin_i$, where d_i is the i-th bit of the difference D, and cin_i and $cout_i$ have the same meaning as before. For addition, the carry-in bit is defined as $cin_i = cout_{i-1}$, with $cin_0 = 0$, while for subtraction it is defined as $cin_i = cout_{i-1}$, with $cin_0 = 1$. The process of bit-serial addition is as indicated in Fig. 4.1, and the corresponding process of bit-serial subtraction is as indicated in Fig. 4.2.

	a_{n-1}	a_{n-2}	...	a_2	a_1	a_0
	b_{n-1}	b_{n-2}	...	b_2	b_1	b_0
cin_n	cin_{n-1}	cin_{n-2}	...	cin_2	cin_1	0
s_n	s_{n-1}	s_{n-2}	...	s_2	s_1	s_0

Fig. 4.1 Bit-Serial Addition

	a_{n-1}	a_{n-2}	...	a_2	a_1	a_0
	$\overline{b}_{n-1}$	$\overline{b}_{n-2}$	...	$\overline{b}_2$	$\overline{b}_1$	$\overline{b}_0$
cin_n	cin_{n-1}	cin_{n-2}	...	cin_2	cin_1	1
d_n	d_{n-1}	d_{n-2}	...	d_2	d_1	d_0

Fig. 4.2 Bit-Serial Subtraction

It should be pointed out that the results of the n-bit addition and subtraction operations may require up to n+1 bits. In practical situations, the SWL is chosen judiciously to ensure that these results are represented correctly.

4.1.2 Hardware Cells for Bit-Serial Addition and Subtraction

The main processing element within each of the above hardware cells is a single-bit full-adder as shown in Fig. 4.3, where *F* and *COUT* represent the two-bit sum resulting from the addition of the inputs *X*, *Y*, and *CIN*.

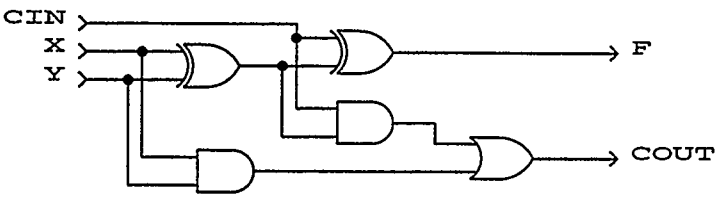


Fig. 4.3 Single-Bit Full-Adder Cell (*FULL ADDER*)

The hardware cell for bit-serial addition $S = A + B$ is shown in Fig. 4.4. In the *ADD* cell, the control signal connected to the *CNTRL* input is asserted at the arrival time of the LSBs a_0 and b_0 of *A* and *B*. This will force the LSB carry-in signal cin_0 to be logic 0. For the remaining n-1 bits to be added, the control signal will remain low, allowing the carry-out to recirculate through the D flip-flop and to become the carry-in for the next single-bit addition. The output sum bit s_i resulting from each single-bit addition is latched with a D flip-flop, giving the addition cell a latency of 1 bit. This output latching will prevent long chains of gates (and their associated gate delays) from occurring, thus allowing the realization of a tightly pipelined structure. In addition, the latch will facilitate a glitch free output signal for use by any downstream hardware cells.

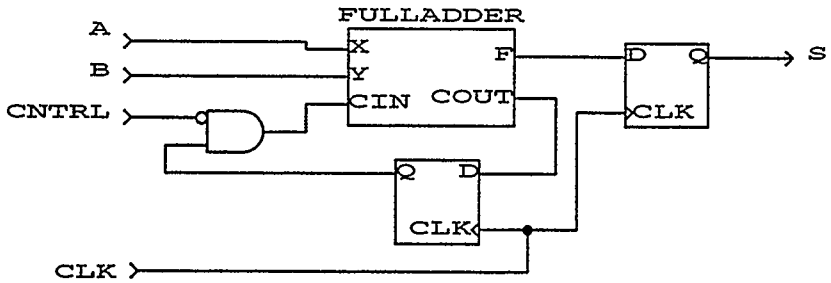


Fig. 4.4 Bit-Serial Addition Cell (*ADD*)

The hardware cell for bit-serial subtraction $D = A - B$ is shown in Fig. 4.5. As before, the control signal connected to the *CNTRL* input of the *SUBT* cell is asserted at the time the LSBs of A and B arrive, forcing the LSB carry-in signal cin_0 to be logic 1. For the remaining n-1 bits to be subtracted, the control signal will remain low, allowing the carry-out to recirculate through the D flip-flop and to become the carry-in for the next single-bit addition. Moreover, the individual bits of B are complemented by an inverter to facilitate subtraction as the addition operation $D = A + (-B)$. As in the addition cell, each difference bit d_i is latched with a D flip-flop, giving the subtraction cell a latency of 1 bit.

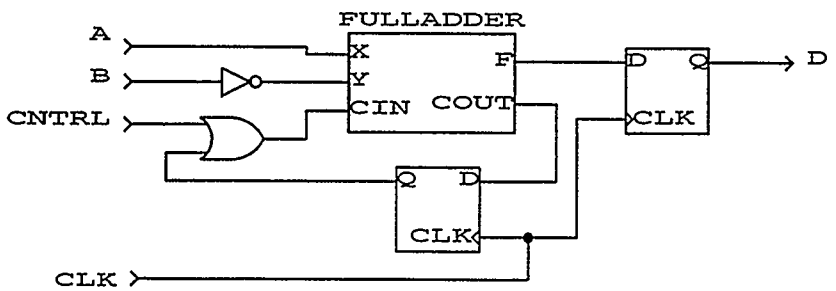


Fig. 4.5 Bit-Serial Subtraction Cell (*SUBT*)

A hardware cell which can be controlled to perform either bit-serial addition or subtraction is shown in Fig. 4.6. The signal applied to the *SIGN* input remains at a certain value for the length of the input data and selects the operation of addition when it is low, and subtraction when it is high. As in the addition and subtraction cells, the control signal

connected to the *CNTRL* input is asserted at the time the LSBs of *A* and *B* arrive, forcing the multiplexor to select the value of the LSB carry-in as the value of the *SIGN* input. This will result in a LSB carry-in of 0 for addition and 1 for subtraction, as required. For the operation of addition, the signal arriving on the *B* input line will be left unchanged by the exclusive-or gate. For the operation of subtraction, the logic high value of the *SIGN* input will cause the exclusive-or gate to act as a programmable inverter, causing the input *B* to be complemented.

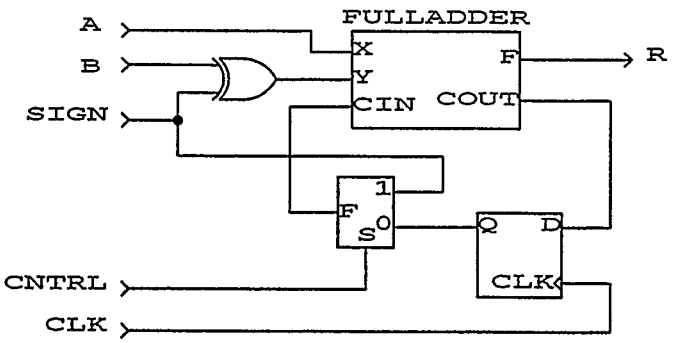


Fig. 4.6 Bit-Serial Selectable Addition/Subtraction Cell (*ADDER*)

The hardware cell in Fig. 4.6 will be referred to as an *ADDER* cell, and will be used within the multiplier cells to be discussed in the following section. The output *R* need not be latched with a D flip-flop as it is latched within the multiplier cell itself.

4.2 Two's Complement Bit-Serial Multiplication

This section is concerned with the design of hardware cells for the operation of TC bit-serial multiplication, where not only the data but also the multiplier coefficient are applied to the multiplier serially. A number of approaches to TC bit-serial multiplication are discussed, including direct TC multiplication, the Booth multiplication algorithm, and the modified Booth multiplication algorithm. Each of these is then followed by a discus-

sion of their corresponding hardware cells.

A bit-serial hardware multiplier is designed as a set of cells connected in tandem as indicated in Fig. 4.7, where the number of the required cells and their respective design will be discussed later. In this way, the inputs to the first cell include the multiplicand *X*, the multiplier coefficient *Y*, a control signal *CNTRL* which is asserted upon the arrival of the LSBs of *X* and *Y*, and ground connected to the *MSWI* and *LSWI* inputs. The output from the last cell in the tandem connection is the product *P*, where the *n*-bit most significant word (MSW) of the product *PM* appears on the *MSWO* line, and the *m*-bit least significant word (LSW) of the product *PL* appears on the *LSWO* line. Finally, the control signal *CNTRL* will leave the last cell synchronized with the LSB of the *MSWO* signal.

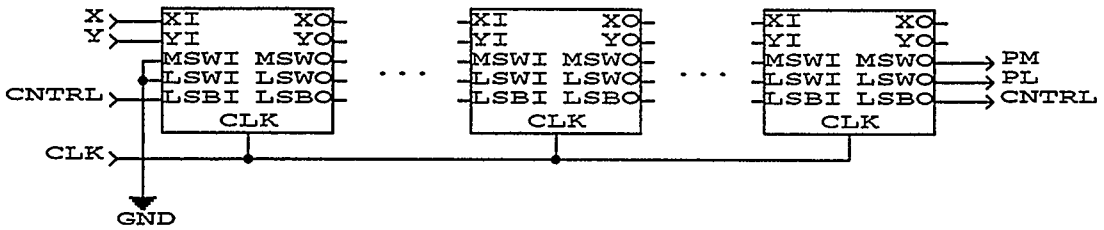


Fig. 4.7 Series Connection of Bit-Serial Multiplier Cells

In Fig. 4.7, each individual cell interprets one or more multiplier coefficient bits and based on the logic values of these bits forms a partial product. This partial product is then added to the input signal entering the multiplier cell on the *MSWI* line. The output from each of these cells includes *X* on the *XO* line, *Y* on the *YO* line, and the control signal *CNTRL* on the *LSBO* line, each of which is delayed by a certain number of bit-clock cycles. Moreover, in each individual cell, one or more of the least significant partial product sum bits will be appended to the LSW of the product arriving on the *LSWI* line, and the result will then leave on the *LSWO* line. The remainder of the partial product sum will leave on the *MSWO* line.

4.2.1 Direct Bit-Serial Multiplication

The product P resulting from the binary TC multiplication of an n-bit multiplicand X and an m-bit multiplier Y may be obtained in accordance with

$$P = Y \cdot X = -PP_{m-1} + \sum_{i=0}^{m-2} PP_i \tag{4.1}$$

where $PP_i = y_i 2^i \cdot X$ represents the i-th partial product. The resulting n-bit by m-bit TC multiplication can be performed as indicated in Fig. 4.8, where all partial products require their sign to be extended to the $(n+m-1)$-th bit location in order to obtain a correct product P. It should be noted that the final partial product must be subtracted from the sum of the other PP_is.

				x_{n-1}	x_{n-2}	...	x_1	x_0
				y_{m-1}	y_{m-2}	...	y_1	y_0
y_0x_{n-1}	y_0x_{n-1}	y_0x_{n-1}	y_0x_{n-1}	y_0x_{n-1}	y_0x_{n-2}	...	y_0x_1	y_0x_0
y_1x_{n-1}	y_1x_{n-1}	y_1x_{n-1}	y_1x_{n-1}	y_1x_{n-2}	...	y_1x_1	y_1x_0	
.	.	.	.	.	.	.		
$y_{m-2}x_{n-1}$	$y_{m-2}x_{n-1}$	$y_{m-2}x_{n-2}$	...	$y_{m-2}x_1$	$y_{m-2}x_0$			
$-y_{m-1}x_{n-1}$	$-y_{m-1}x_{n-2}$	...	$-y_{m-1}x_1$	$-y_{m-1}x_0$				
P_{n+m-1}	P_{n+m-2}		...			P_2	P_1	P_0

Fig. 4.8 Standard Two's Complement Multiplication

4.2.2 Hardware Cells for Direct Bit-Serial Multiplication

The TC serial multiplication in Eqn. (4.1) can be implemented in hardware as a tandem connection of m cells (as shown in [27]), where the i-th cell forms the partial product PP_i, and then sums it to the $(i-1)$-th partial product sum. This process is as shown in

Fig. 4.9, where $PPS_{i-1} = pps_{i-1,n-1}pps_{i-1,n-2}\cdots pps_{i-1,1}pps_{i-1,0}$. Each of the first m-1 PP_is and PPS_is are computed in identical cells, while the final partial product and partial product sum require a modified version of these cells to allow for the sign bit considerations. All of these cells have a fixed latency, resulting in the LSB of the MSW of the product being available after $L = 2 \cdot m + 1$ bit-clock periods.

			x_{n-1}	x_{n-2}	...	x_1	x_0		X
				y_{m-1}	...	y_1	y_0		Y
			0	0	...	0	0		PPS_{-1}
			y_0x_{n-1}	y_0x_{n-2}	...	y_0x_1	y_0x_0		PP_0
		$pps_{0,n}$	$pps_{0,n-1}$	$pps_{0,n-2}$	...	$pps_{0,1}$	$pps_{0,0}$		PPS_0
		y_1x_{n-1}	y_1x_{n-2}	...	y_1x_1	y_1x_0			PP_1
	$pps_{1,n}$	$pps_{1,n-1}$	$pps_{1,n-2}$	...	$pps_{1,1}$	$pps_{1,0}$			PPS_1
			...						
			...						
			...						
$pps_{m-2,n}$	$pps_{m-2,n-1}$	$pps_{m-2,n-2}$	...	$pps_{m-2,1}$	$pps_{m-2,0}$				PPS_{m-2}
$-y_{m-1}x_{n-1}$	$-y_{m-1}x_{n-2}$	...	$-y_{m-1}x_1$	$-y_{m-1}x_0$					PP_{m-1}
P_{n+m-1}	P_{n+m-2}		...	P_m	P_{m-1}	...	P_1	P_0	P
		MSW				*LSW*			

Fig. 4.9 Serial Two's Complement Multiplication

4.2.3 Booth Multiplication

A.D. Booth [28] rearranged the binary partial products in Eqn. (4.1) in accordance with

$$P = Y \cdot X = \sum_{i=0}^{m-1} PP_i \tag{4.2}$$

in order to allow all of the partial products to be computed in an identical manner, where $PP_i = (-y_i + y_{i-1})X2^i$. This leads to a multiplier implementation requiring m identical cells, where the i-th cell interprets two successive multiplier coefficient bits y_i and y_{i-1} (with $y_{-1} = 0$), to perform a corresponding operation as indicated in Table 4.1.

Coefficient bits		Operation of i-th Cell	Operation selection signals	
y_i	y_{i-1}		ys	ya
0	0	add 0	0	0
0	1	add $X2^i$	0	1
1	0	subtract $X2^i$	1	1
1	1	subtract 0	1	0

Table 4.1 Booth's Multiplication Algorithm

4.2.4 Hardware Cell for Booth Multiplier

Each of the m identical Booth multiplier cells has a gate-level implementation as shown in Fig. 4.10, where the operation of each cell is as follows.

The signal x within the i-th cell represents the multiplicand X shifted left by i bits relative to the partial product sum PPS_{i-1} arriving on the *MSWI* input line. This facilitates the formation of the $X2^i$ component of PP_i. Furthermore, the signal ya (c.f. Table 4.1) will select the appropriate partial product (0 or $X2^i$) which will subsequently appear at the input to the *ADDER* cell for processing along with PPS_{i-1}. Then, the signal ys (c.f. Table 4.1) will determine whether the *ADDER* cell is to perform the operation of addition (for $ys = 0$) or subtraction (for $ys = 1$). The signals ys and ya will remain unchanged until a future assertion of the control signal *CNTRL*, associated with the arrival of new multiplier coefficient bits. The output of the *ADDER* cell will be PPS_i. The LSB of this result is not

necessary for the calculation of any future partial product sum, and will be appended to the LSW of the product being formed. This is facilitated by the signal *sel* being high for one-bit period, causing the *LSWI/LSWO* multiplexor to select the LSB of PPS_i. The assertion of the *sel* signal also facilitates the one bit sign extension of the previous partial product sum which was formed in this cell and which is currently leaving the cell on the *MSWO* line. When the *sel* signal is low, the partial product sum will leave the cell on the *MSWO* line and enter the next cell on its *MSWI* line.

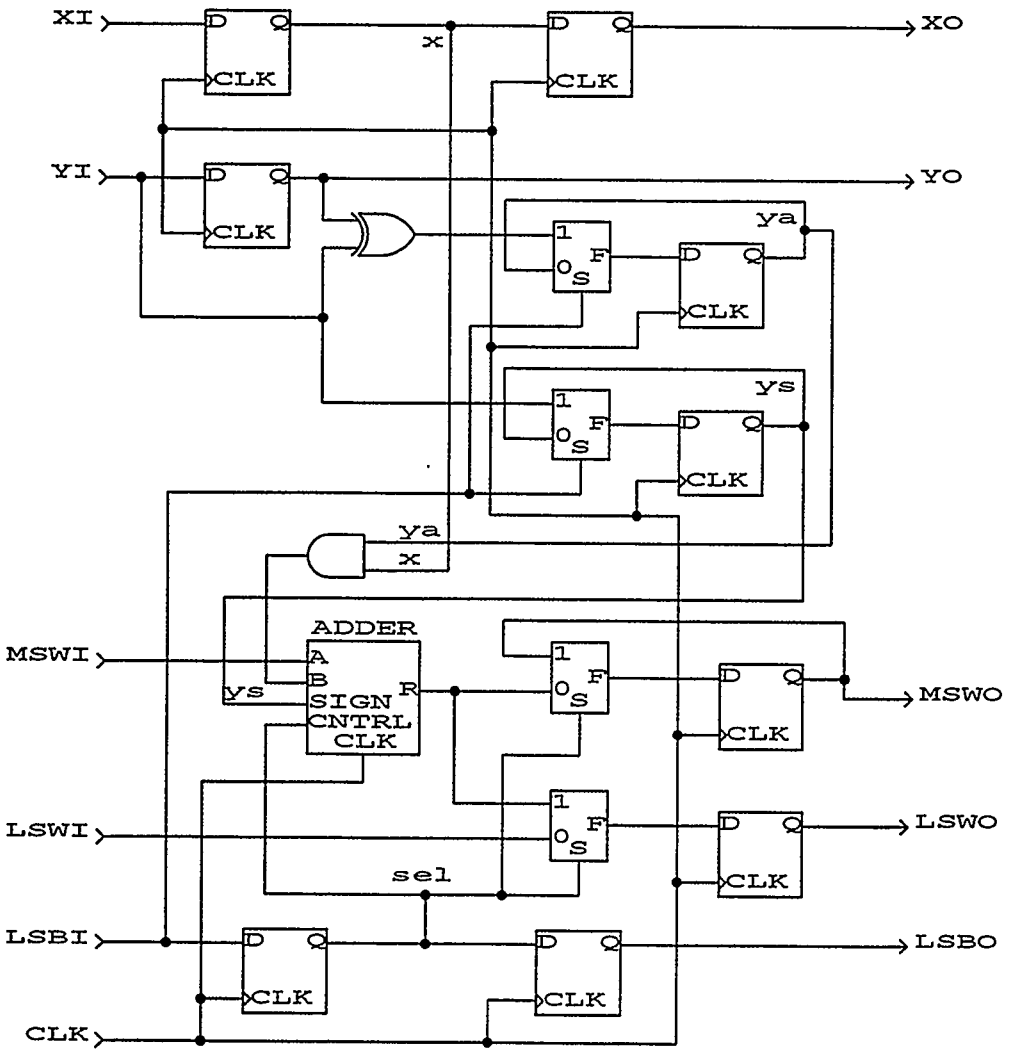


Fig. 4.10 *i*-th Booth Multiplier Cell

The waveforms shown in Fig. 4.11 illustrate the operation of the Booth multiplier for the case where the n-bit multiplicand X is an 11-bit signal (which includes 1 sign extension bit required to prevent overflow), and the m-bit multiplier Y is a 4-bit signal. In Frame 1, the multiplicand $X = -1$, and the multiplier $Y = 1$, yielding the 15-bit product $P = Y \cdot X = -1 = 111111111111111\text{b}$, where the 4-bit LSW is given by the signal *PL*, and the 11-bit MSW is given by the signal *PM*. In Frame 2, the multiplicand $X = -512$, and the multiplier $Y = -8$, yielding the 15-bit product $P = Y \cdot X = 4096 = 001000000000000\text{b}$, where again the 4-bit LSW is given by the signal *PL*, and the 11-bit MSW is given by the signal *PM*.

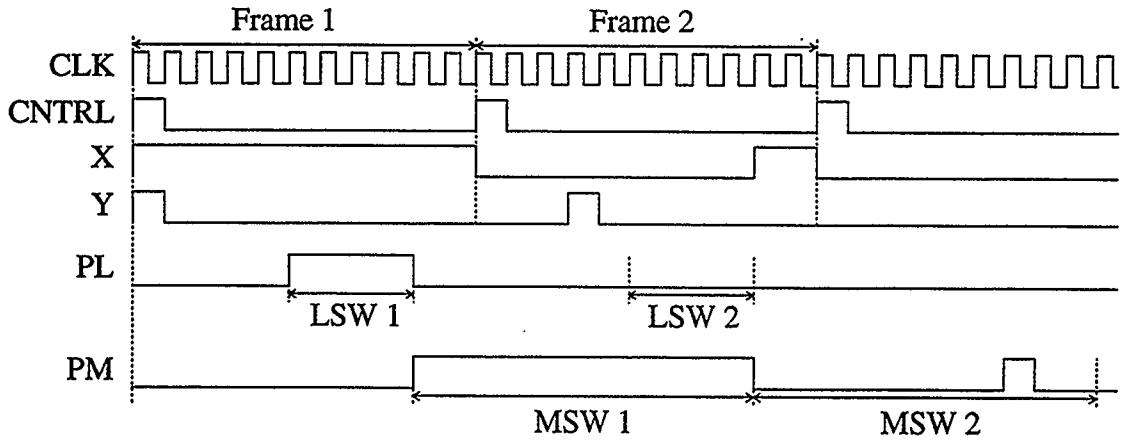


Fig. 4.11 Booth Multiplier Waveforms

Although the Booth multiplication algorithm leads to a regular multiplier structure with each hardware cell being identical, it does not decrease either the number of partial products or the number of corresponding hardware cells, and therefore does not reduce the latency of the operation from that of the direct TC multiplication. Moreover, the Booth multiplier cells are more complex than the direct TC multiplication cells. Therefore, any advantage secured by the regular structure of a Booth multiplier is more than offset by its increased hardware complexity rendering this multiplier impractical.

4.2.5 Two's Complement Modified Booth Multiplication

O.L. MacSorley [29] modified the Booth algorithm (c.f. Eqn. (4.2)) in such a manner as to reduce the number of partial product computations by half while maintaining the regularity of the algorithm for an actual implementation. In the resulting modified Booth algorithm, each partial product is computed based on three successive multiplier bits, y_{2i+1}, y_{2i}, and y_{2i-1}, in accordance with

$$P = Y \cdot X = \sum_{i=0}^{\frac{(m-2)}{2}} PP_i \tag{4.3}$$

where

$$PP_i = z_i 4^i X \tag{4.4}$$

represents the i-th partial product, where

$$z_i = (-2y_{2i+1} + y_{2i} + y_{2i-1}) \tag{4.5}$$

can take on values $z_i \in \{-2, -1, 0, 1, 2\}$, and where $y_{-1} = 0$. The process of TC modified Booth multiplication is illustrated in Fig. 4.12.

				x_{n-1}	x_{n-2}	...	x_1	x_0
				y_{m-1}	y_{m-2}	...	y_1	y_0
z_0x_{n-1}	z_0x_{n-1}	z_0x_{n-1}	z_0x_{n-1}	z_0x_{n-1}	z_0x_{n-2}	...	z_0x_1	z_0x_0
z_1x_{n-1}	z_1x_{n-1}	z_1x_{n-1}	z_1x_{n-2}	...	z_1x_1	z_1x_0		
.	.	.	.	.				
$z_{(m-2)/2}x_{n-1}$	$z_{(m-2)/2}x_{n-1}$	$z_{(m-2)/2}x_{n-2}$	...	$z_{(m-2)/2}x_0$				
P_{n+m-1}	P_{n+m-2}		...			P_2	P_1	P_0

Fig. 4.12 Two's Complement Modified Booth Multiplication

The hardware implementation of the modified Booth multiplication algorithm

involves a set of $m/2$ identical cells connected in tandem, where the operation of the i-th cell is as indicated in Table 4.2.

Coefficient bits			Operation of i-th Cell	Operation selection signals		
y_{2i+1}	y_{2i}	y_{2i-1}		ys	yb	ya
0	0	0	add 0	0	0	0
0	0	1	add $X4^i$	0	0	1
0	1	0	add $X4^i$	0	0	1
0	1	1	add $2X4^i$	0	1	0
1	0	0	subtract $2X4^i$	1	1	0
1	0	1	subtract $X4^i$	1	0	1
1	1	0	subtract $X4^i$	1	0	1
1	1	1	subtract 0	1	0	0

Table 4.2 Modified Booth Multiplication Algorithm

4.2.6 Hardware Cells for Modified Booth Multiplier

The individual modified Booth multiplier cells have a gate-level implementation as shown in Fig. 4.13. In order to perform the operations indicated in Table 4.2, the i-th cell in the modified Booth multiplier requires two versions of the multiplicand X to be available, namely $X4^i$ and $2X4^i$. The signal x within the i-th cell represents the individual bits of X shifted left by $2i$ bits relative to the partial product sum arriving on the *MSWI* input line, and provides the $X4^i$ component. The signal $2x$ represents the signal x shifted left by 1 bit, and thus provides the $2X4^i$ component. The three coefficient bits y_{2i+1}, y_{2i}, and y_{2i-1} are mapped to the latched signals *ys*, *yb*, and *ya* in accordance with the entries in Table 4.2. Therefore, if $ya = 1$, then the signal x will arrive at the *ADDER* cell input. Furthermore, if $yb = 1$, then the signal $2x$ will arrive at the *ADDER* cell input with its LSB set to zero by the control signal *c2* (asserted at the inverting terminal of the corresponding

AND gate). Finally, if $ya = 0$ and $yb = 0$, then the *ADDER* cell input will be 0. The other *ADDER* cell input will be the partial product sum PPS_{i-1} arriving at the multiplier cell on the *MSWI* line. The signal *ys* (c.f. Table 4.2) will determine the operation that the *ADDER* cell is to perform, selecting addition for $ys = 0$ and subtraction for $ys = 1$. The sig-

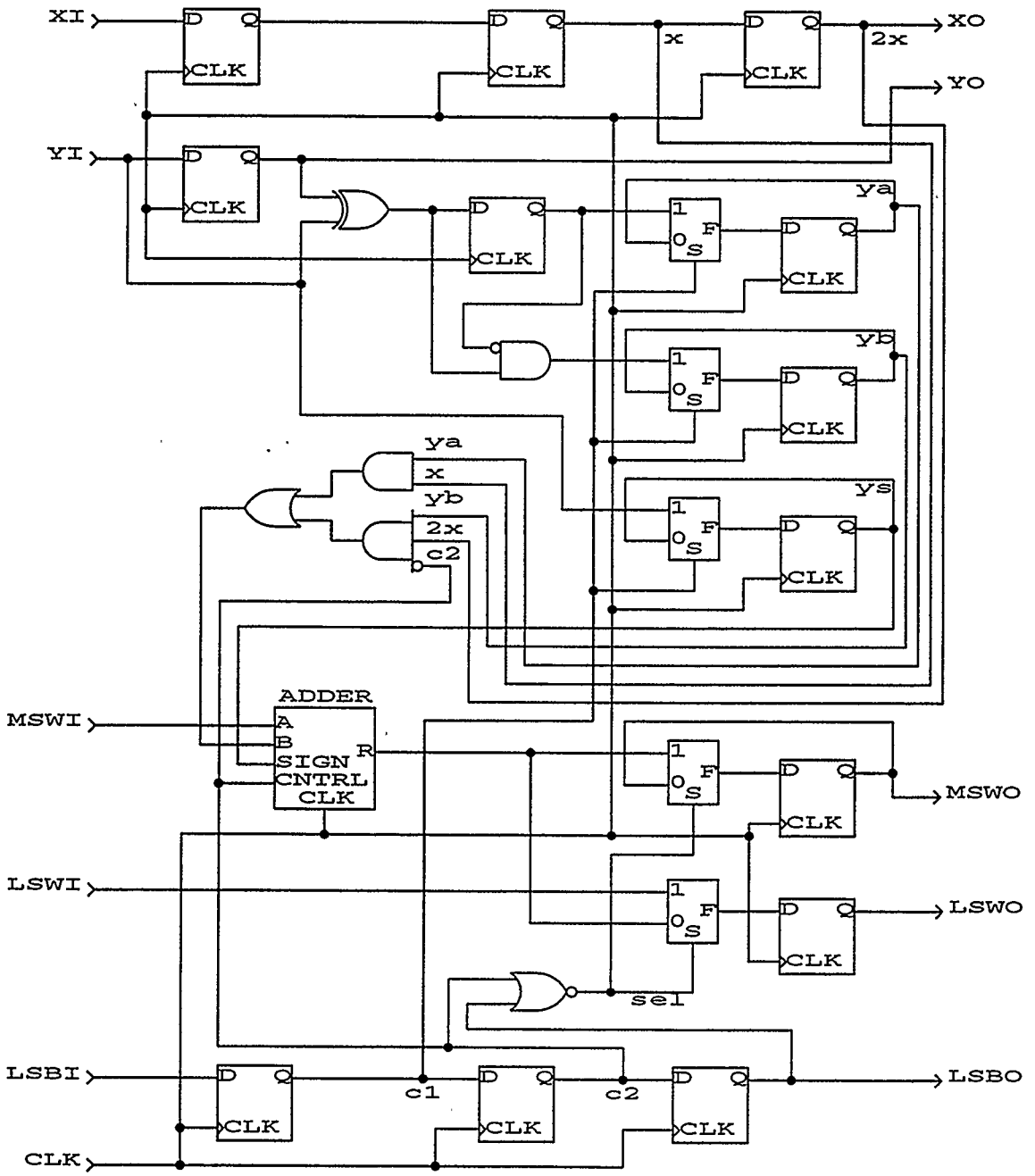


Fig. 4.13 *i*-th Modified Booth Multiplier Cell (*BOOTH*)

nals *ys*, *yb*, and *ya* will remain unchanged until a future assertion of the control signal *CNTRL*, associated with the arrival of new multiplier coefficient bits. The output of the *ADDER* will be the partial product sum PPS_i. Because the modified Booth algorithm is computing only *m*/2 partial products, the first two LSBs of this result are not necessary for the calculation of any future partial product sum, and will be appended to the LSW of the product being formed. This is facilitated by the signal *sel* being low for two bit periods, causing the *LSWI/LSWO* multiplexor to select the LSBs of PPS_i. The logic 0 value of the *sel* signal also facilitates the two-bit sign extension of the previous partial product sum which was formed in this cell and which is currently leaving the cell on the *MSWO* line. When the *sel* signal is logic 1, the partial product sum will leave the cell on the *MSWO* line and enter the next cell on its *MSWI* line. In this way, since each partial product is shifted left two bit positions relative to the previous partial product sum, the sign extension required by the TC number system is satisfied. The modified Booth multiplier cell in Fig. 4.13, when realized without the *LSWI/LSWO* data path, will be referred to as a *BOOTH* cell.

In the hardware realization of the modified Booth multiplier, the requirement that $y_{-1} = 0$ (c.f. Eqn. (4.5)) may be satisfied by ensuring that during the bit-period prior to the LSB of the coefficient *Y* arriving at the first cell, the corresponding *YI* input is logic 0. Furthermore, the data signal *X* requires a two-bit sign extension, i.e. the three MSBs of *X* must have the same sign. One of these sign extension bits is required to avoid overflow, and the other is to ensure that no sign information is lost when *X* is shifted left by one bit.

The modified Booth algorithm offers significant improvements over the corresponding direct TC and Booth multiplication algorithms in an actual hardware implementation.

The primary improvement stems from a reduction in the number of partial products (from m to $m/2$). This reduction, in turn, gives rise to two further improvements. First, since there are only half as many multiplier cells, and since each cell is of less than twice the complexity as compared to the direct TC or Booth cells, the total amount of hardware required to implement the multiplier is reduced. Second, the latency of the multiplication operation is reduced from $2 \cdot m + 1$ bits to $3 \cdot m/2 + 2$ bits.

4.2.7 The Returned Multiplication Product

The modified Booth multiplier receives as input the n-bit multiplicand X, which can take on a maximum value of $X_{max} = 2^{n-1} - 1$ and a minimum value of $X_{min} = -2^{n-1}$, as well as the m-bit multiplier Y, which can take on a maximum value of $Y_{max} = 2^{m-1} - 1$ and a minimum value of $Y_{min} = -2^{m-1}$. All possible products $P = Y \cdot X$ will possess two sign bits, with the one exception of the maximum product $P_{max} = Y_{min}X_{min}$ which will have only one sign bit. However, if Y_{min} is restricted to a minimum value of $Y_{min} = -2^{m-1} + 1$, then all the products P will have two sign bits. Therefore, no sign information is lost if the MSB of the (n+m)-bit product is discarded, reducing the product to n+m-1 bits. This (n+m-1)-bit product is in turn reduced to an n-bit result by removal of the m-1 LSBs. The returned n-bit result is shown by the shaded portion in Fig. 4.14. In this way, the multiplication operation has effectively been performed in accordance with

$$P_{eff} = \frac{(Y \cdot X)}{2^{m-1}} = Y_{eff} \cdot X, \tag{4.6}$$

where $Y_{eff} = Y/2^{m-1}$. Furthermore, with $Y_{max} = 2^{m-1} - 1$ and $Y_{min} = -2^{m-1} + 1$, $Y/Y_{max} < 1$ and $Y/Y_{min} > -1$. Thus, the effective multiplier coef-

ficient Y_{eff} is a number in the range $-1 < Y_{eff} < 1$. Therefore, the restriction on Y_{min} is of no practical significance, because the multiplication by $Y_{eff} = -1$ can be achieved either through a phase inversion in the digital filter SFG, or through the usual operation of complementing X and adding 1 to its LSB. In addition, the returned n-bit product will have its LSB available after a latency of $L = 3 \cdot m/2 + 1$ bit-clock periods.

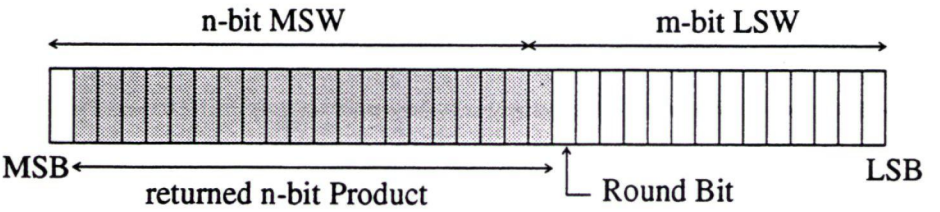


Fig. 4.14 Returned n-bit Product

In actual digital filter design, it may so happen that the required multiplier coefficient Y_{req} lies outside the range ± 1. In such situations, the required multiplication can be achieved by first scaling the signal X by 2^i (facilitated by an i-bit left shift), and then multiplying the result by an effective multiplier value of

$$Y_{eff} = \frac{Y_{req}}{2^i}, \tag{4.7}$$

where i is chosen such that 2^i is the smallest possible number that is greater than Y_{req}. Of course, the scaling operation must be performed such that after the i-bit left shift, the new LSBs appended to the signal X are logic 0, and such that the sign information is retained.

4.2.8 Multiplier Product Rounding

When returning an $(n+m)$-bit binary product as an n-bit number, it is desirable to obtain a result which is as close to the original binary product as possible. The process of obtaining this result is called rounding. The IEEE standard 754 default rounding mode is *round to nearest/even*, which states that "*in this mode the representable value nearest to*

the infinitely precise result shall be delivered; if the two nearest representable values are equally near, the one with its least significant bit zero shall be delivered" [31]. The implementation of round to nearest/even in a bit-serial multiplier requires additional hardware associated with the decision involving the outcome of the half-way cases. This decision making process is facilitated through the use of a sticky bit. The sticky bit is defined as 0 if all of the bits to the right of the round bit (c.f. Fig. 4.14) are 0, and is defined as 1 if any of these bits is 1. In order to compute the sticky bit in a bit-serial multiplier, it is required to retain the *LSWI/LSWO* circuitry in the multiplier cells. In addition, a chain of D flip-flops is required to store the LSW to facilitate the computation of the sticky bit. The sticky bit then becomes the carry-in signal (at the round bit position) to the half-adder performing the rounding operation in the final multiplier cell. Although round to nearest/even does not involve an excessive hardware cost, a more conventional binary rounding system, *round to nearest/up*, provides identical results with the exception of half-way cases, and entails neither the decision making process nor the corresponding hardware requirement. Round to nearest/up is performed by adding 1 to the round bit, and then truncating the result to n bits. As such, the LSW forming circuit is not required, nor is the storage of the LSW. In round to nearest/up, a number with both an integer and fractional part would round towards ∞ in the case of a tie, i.e. the case of rounding to an integer number would result in $-3.5 \rightarrow -3$ and $4.5 \rightarrow 5$.

To incorporate the round to nearest/up process, slight changes are made to the last modified Booth cell in the multiplier. These changes are reflected in Fig. 4.15, where the LSW forming circuitry has been removed. Changing the *NOR* gate in the original cell to an inverter results in the removal of the MSB of the product (the redundant sign-bit) from

the *MSWO* signal. This inverter is then removed by reversing the *MSWO* multiplexor inputs. The second change is to incorporate a half-adder circuit with carry-recirculation to facilitate the addition of 1 to the round bit (c.f. Fig. 4.14). When the round bit of the partial product sum leaves the *ADDER* cell, the control signal *c2* will be high. This will cause the

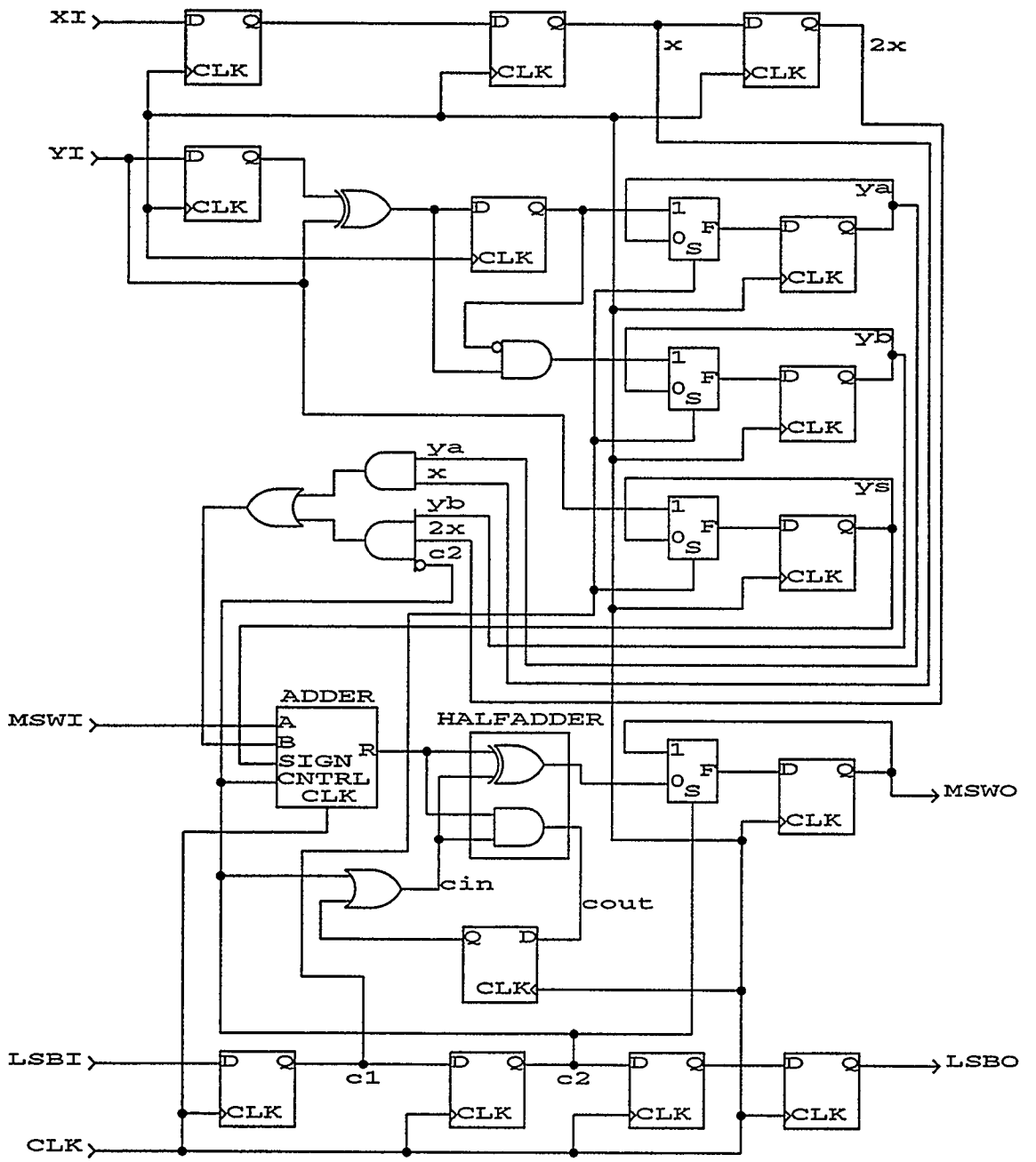


Fig. 4.15 Final Multiplier Cell Incorporating Rounding Hardware (*BOOTHR*)

carry-in signal *cin* to the half-adder circuit to be logic 1. The remaining carry-in signals for this product will be the half-adder carry-out signals *cout* from the previous bit additions. The final change is to add a D flip-flop to the *LSBI/LSBO* path. This results in the output signal *CNTRL* having an asserted value synchronized with the LSB of the *MSWO* signal.

The multiplier chosen for use within the LDI Jaumann digital filter implementation will consist of a tandem connection of *m*/2 - 1 *BOOTH* cells together with a *BOOTHR* cell. As an example, an $m = 6$ bit modified Booth multiplier is shown in Fig. 4.16, where the LSW forming circuitry is not included. In this multiplier, an input control signal *C0* will depart as an output control signal *C10* synchronized with the LSB of the *n*-bit product.

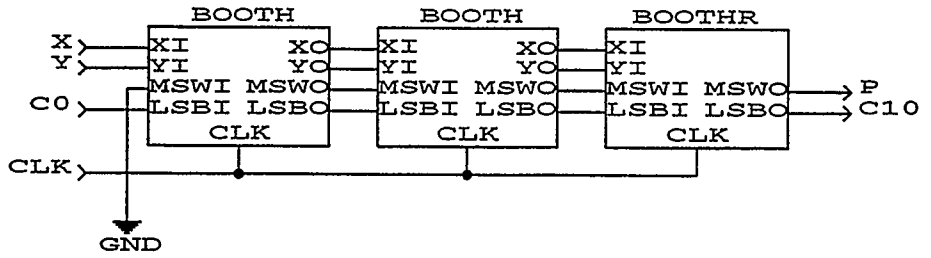


Fig. 4.16 Modified Booth Multiplier with $m = 6$ (*BOOTH6*)

The waveforms shown in Fig. 4.17 illustrate the operation of the modified Booth multiplier for the case where the *n*-bit multiplicand *X* is a 12-bit signal (which includes the 2 sign extension bits), and the *m*-bit multiplier *Y* is a 6-bit signal. In Frame 1, the multiplicand $X = 13$, and the multiplier $Y = 16$, yielding $Y_{eff} = 0.5$, and $P_{eff} = Y_{eff} \cdot X = 6.5$. This result will be rounded up to a 12-bit integer 7 = 000000000111\b, and output as the signal P_{eff}.In Frame 2, the multiplicand $X = 13$, and the multiplier $Y = -16$, yielding $Y_{eff} = -0.5$, and $P_{eff} = Y_{eff} \cdot X = -6.5$. This result will be rounded up to a 12-bit inte-

ger -6 = 000000000111\b, and output as the signal P_{eff}.

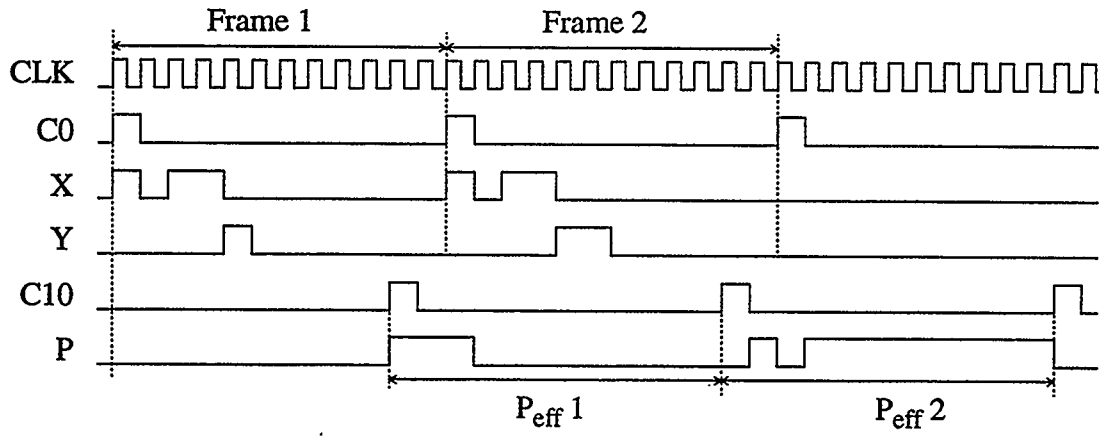


Fig. 4.17 Modified Booth Multiplier Waveforms

4.3 Data Registers

This section is concerned with the development of the storage and shift register cells required for the implementation of a bit-serial multirate LDI Jaumann digital filter.

4.3.1 Parallel-to-Serial Shift Register

The multirate bit-serial LDI Jaumann digital filter receives its digital input signal from an analog-to-digital (A/D) converter, and receives its multiplier coefficient values as digital signals from an EPROM. Both of these signals arrive in a parallel format and are subsequently converted to a serial format. These conversions are performed through the use of a parallel to serial shift register such as that shown in Fig. 4.18a, where the individual modules represent a 2:1 multiplexor in combination with a D flip-flop as shown in Fig. 4.18b.

The parallel-to-serial shift register in Fig. 4.18 receives the parallel data at the *D3*, *D2*, *D1*, and *D0* inputs, and provides the serial data at the output *SER*. By asserting the

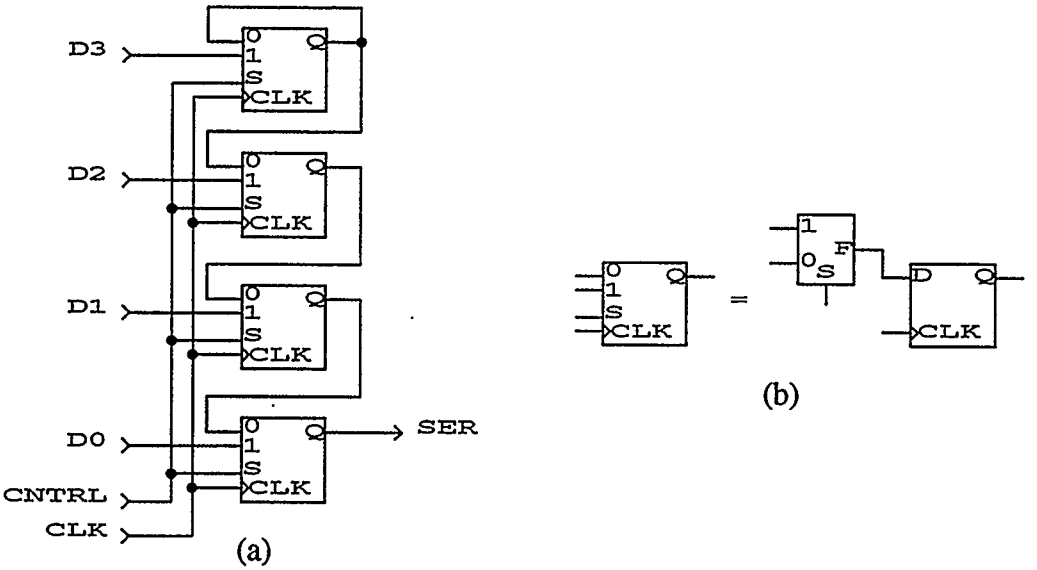


Fig. 4.18 4-Bit Parallel to Serial Shift Register (*PTOS4*)

CNTRL input, the D flip-flop multiplexors will select the respective *D* inputs. When the *CNTRL* signal becomes logic low, the data will shift out of the register in a serial format. The feedback loop on the top D flip-flop multiplexor will sign-extend the output serial signal until the next time data is latched in. These shift registers will be denoted as *PTOSx*, with *x* representing the number of parallel input bits.

4.3.2 Serial-In/Serial-Out Storage Register

In a bit-serial digital filter implementation incorporating multiplexed hardware components, it is usually required to store certain data signals for processing in future sample periods. This operation is performed using the serial-in/serial-out shift register shown in Fig. 4.19.

In the *SHREG4* cell the signal applied at the *SHFT* input is asserted to allow the data to enter the shift register at the input *D*, while the data already stored in the shift register will exit at the output *Q*. When the signal applied to the *SHFT* input is low, the data will

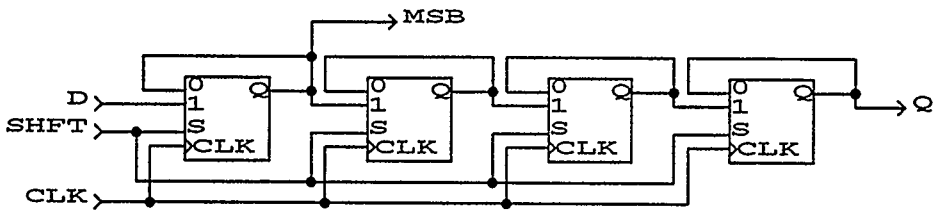


Fig. 4.19 4-Bit Serial-In / Serial-Out Shift Register (*SHREG4*)

recirculate in the individual D flip-flops. These shift registers will be denoted as *SHREGx*, with x representing the number of bits stored within the register.

4.3.3 Downsampling Serial Shift Register

The multirate bit-serial LDI Jaumann digital filter to be implemented in this thesis requires the serial output from a LP Jaumann digital filter to be downsampled by a factor of 4, and supplied as the serial input to a HP Jaumann digital filter. A 4-bit downsampling shift register satisfying these interface requirements is shown in Fig. 4.20, where the lower shift register chain is operating at a bit-clock rate of f_{clk}, and the upper shift register chain is operating at a bit-clock rate of $f_{clk}/4$. The operation of the downsampling serial shift register is as follows.

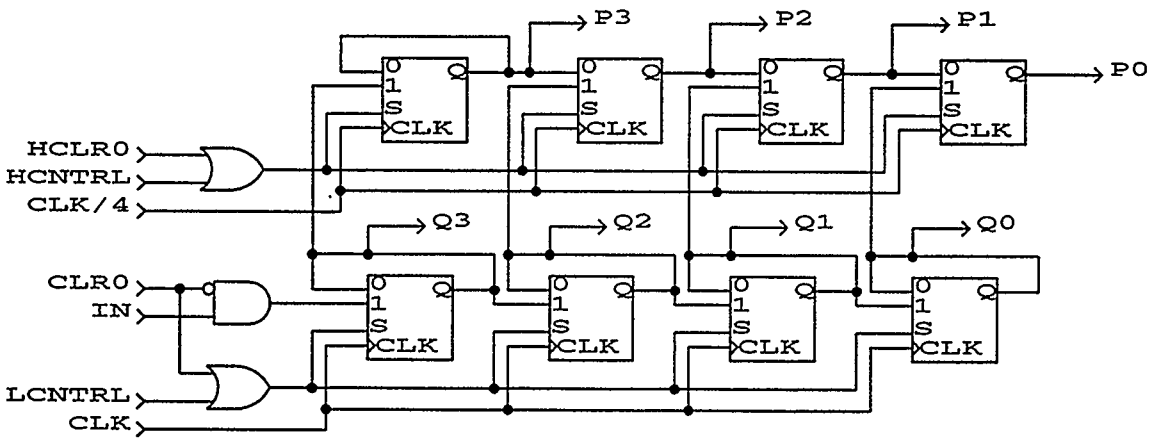


Fig. 4.20 4-Bit Downsampling Serial to Serial Shift Register

When the serial output from the LP digital filter arrives at the shift register input *IN* the *LCNTRL* signal is asserted for 4 bit-clock periods to allow the input data to shift into the lower register. Once the *LCNTRL* signal becomes logic low, the input data bits are latched and will recirculate around their respective D flip-flops. By asserting the *HCNTRL* signal the data currently latched in the lower shift register will be loaded into the top shift register, leaving the *P0* output when the *HCNTRL* signal becomes logic low. The *HCNTRL* signal is only asserted once for every four sets of data entering the lower register, thus satisfying the downsampling requirement.

4.3.4 Delay Register

In a fully pipelined bit-serial digital filter, it is required to incorporate an effective signal delay of W bit-clock periods in order to realize the z^{-1} operation. Moreover, it is required that the internal data signals arrive at arithmetic cells with their LSBs synchronized. These requirements are accomplished through the use of a delay register, which is a chain of D flip-flops as illustrated in Fig. 4.21. In this register, the routing of a clear signal to every D flip-flop is avoided by providing a method for a synchronous clear. In this way, all delay registers are cleared by applying a logic high signal to the *CLR* input during the system reset which causes logic 0 to be shifted into the chain. These delay registers will be denoted as *DELx*, with x representing the number of bit-period delays within the chain.

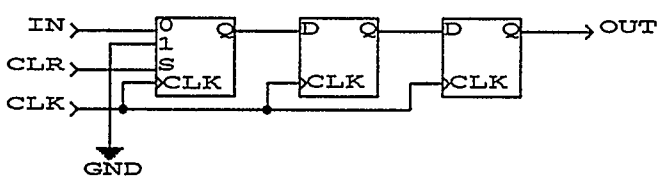


Fig. 4.21 3-Bit Delay Register (*DEL3*)

4.3.5 State Storage Shift Register

In a bit-serial LDI Jaumann digital filter incorporating multiplexed hardware components, the continuous streams of bits representing the individual data signals are required to be stored and/or delayed in such a manner that their LSBs are available at integral multiplies of the SWL. This storage can take on two distinct forms, one of which is associated with the update of a filter state, while the other is associated with the storage of an intermediate signal value.

The filter state update operation is performed by the *STATE* cell shown in Fig. 4.22, where the *ADD* cell and *DELx* cells are as introduced previously. This *STATE* cell corresponds to an LDI Jaumann digital filter employing a SWL $W = 22$ and a multiplier latency of $L_m = 10$, where the *DEL10* and *DEL11* cells were selected to satisfy the architectural requirements (see Sec. 3.5).

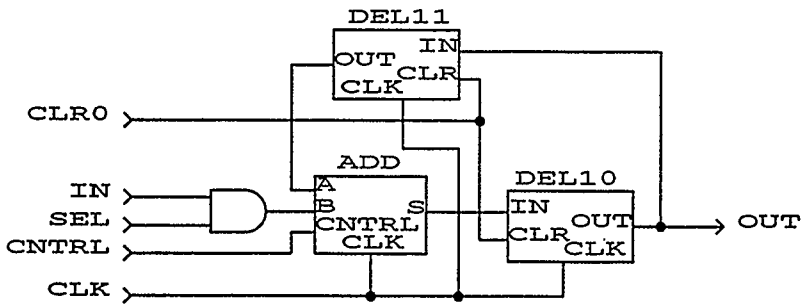


Fig. 4.22 State Storage Shift Register (*STATE*)

The operation of the *STATE* cell in Fig. 4.22 is as follows. The input signal arriving at *IN* will be a stream of data arriving from the output of a multiplier. The control signal at the *SEL* input will be asserted to allow the multiplier product corresponding to this state update operation to enter the *ADD* cell, where it will be added to the state value currently stored in the register. The *CNTRL* input signal is a LSB pulse waveform to cause the initial

carry-in to be logic 0. When the state update operation is complete, the input signal will be logic 0 and the new state value can recirculate around the delay chain. Since the state update operation could result in a carry-out following the final bit-addition of a state update operation, the *CNTRL* signal must be reapplied to prevent this carry from being added to the currently stored value.

The operation of storing the intermediate signals X_{10} and X_{20} is performed by the *STATE0* cell shown in Fig. 4.23. The *STATE0* cell is the same as the *STATE* cell in Fig. 4.22, except for the fact that the *ADD* cell has been replaced by a 2:1 multiplexor in combination with a D flip-flop. In the *STATE0* cell, the assertion of the select signal *SEL* will allow new data to enter the register, and when unasserted will allow the data to recirculate through the delay chain.

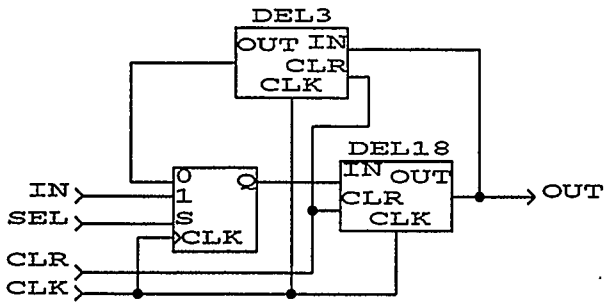


Fig. 4.23 Intermediate Signal Storage Register (*STATE0*)

4.3.6 Serial-to-Parallel Shift Register

The multirate bit-serial LDI Jaumann digital filter to be implemented in this thesis generates a serial digital output signal which is subsequently converted to a parallel format for input to a digital-to-analog (D/A) converter. This conversion is performed through the use of a serial-to-parallel shift register such as that shown in Fig. 4.24.

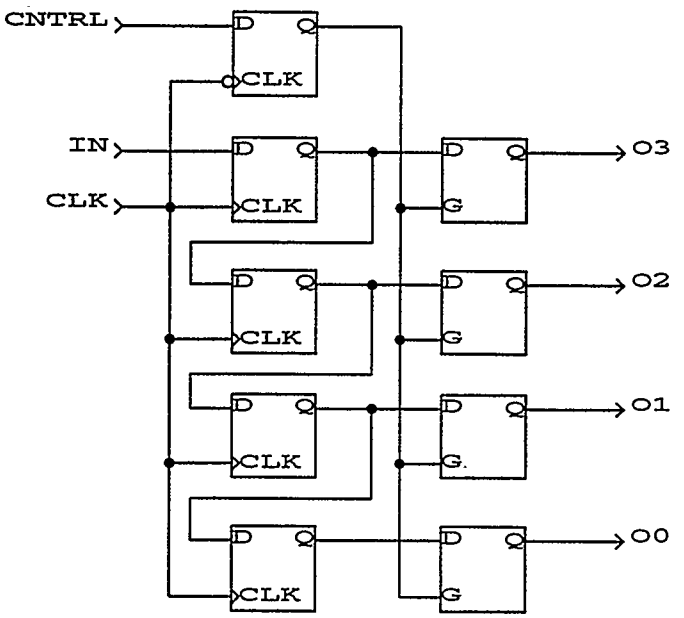


Fig. 4.24 4-Bit Serial to Parallel Converter (*STOP4*)

The shift register in Fig. 4.24 receives its serial data at the input *IN* and produces its latched parallel data at the outputs *O3*, *O2*, *O1*, and *O0*. The serial data will continuously enter the register, and when it is aligned such that the LSB of the desired result is at the input to the *O0* latch, a logic high signal is applied at the control input *CNTRL* to facilitate the latching of the 4-bit output. Once the data settles to a constant value, i.e. during the next bit-period after latching, the D/A converter is supplied with a convert signal. These shift registers will be denoted as *STOPx*, with x representing the number of latched output bits.

4.4 Bit-Serial Control Signal Generator

The multirate bit-serial LDI Jaumann digital filter requires a control signal generator which produces a set of periodic waveforms, as shown in Fig. 4.25, where the SWL is chosen as $W = 4$ bits, and the sample processing time is chosen as $3 \cdot W$.

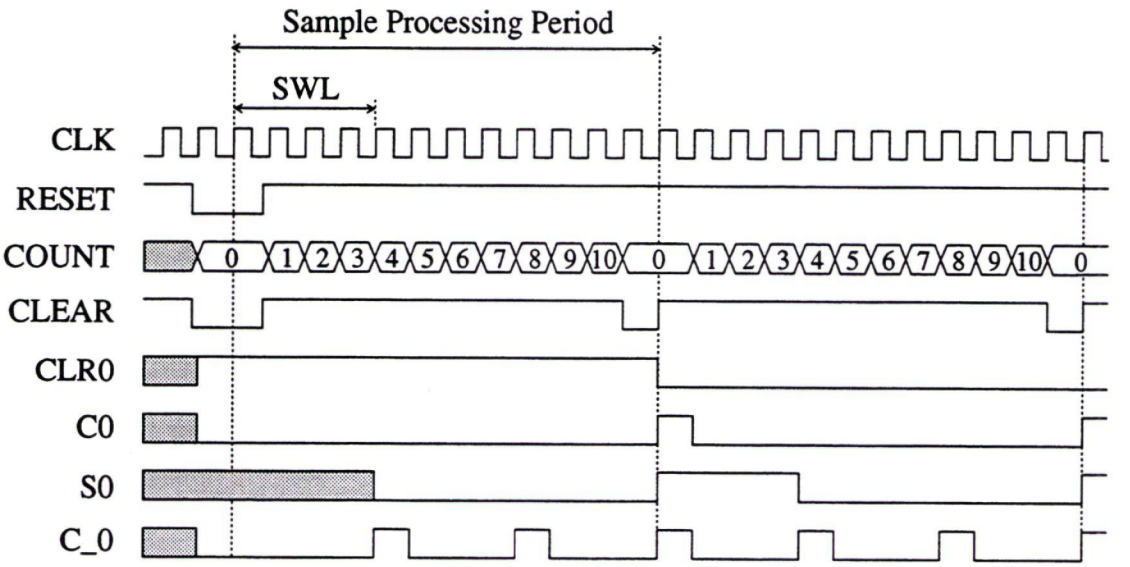


Fig. 4.25 Bit-Serial Control Signal Examples

The central part of the control generator shown in Fig. 4.26 is a counter [34] chosen to accommodate the high-speed clock rate normally associated with bit-serial systems. The operation of the control generator is as follows.

The control generator is reset by a logic 0 being applied to the *RESET* input. This causes the *CLEAR* signal entering the counter to be logic 0, clearing the counter to a count of 0. In addition, the logic 0 applied at the *RESET* input will cause the *CLR0* signal to become logic 1. The count will increment by one on each rising edge of the input signal *CLK*. When the count reaches 10, the *CNT10* signal becomes logic 1. The *CNT10* signal will leave the associated D flip-flop at a count of 11, and the counter will then be cleared. At the first time when a count of 11 is reached, the multiplexor and D flip-flop combination generating the *CLR0* signal will be cleared, causing the *CLR0* signal to become logic 0 when the count reaches zero. The *CLR0* signal will then remain at logic 0 until a future application of logic 0 to the *RESET* input. When the *CNT10* signal exits the D flip-flop, the count is 11, and the output signal *C11* becomes logic 1. This signal is then further delayed

by one bit-period to facilitate the generation of the *C0* signal. In general, the control signals *Cx* for given integers *x* may be generated by either detecting a count of *x-1* and then latching this count with a D flip-flop (thus delaying this count by one-bit period), or by

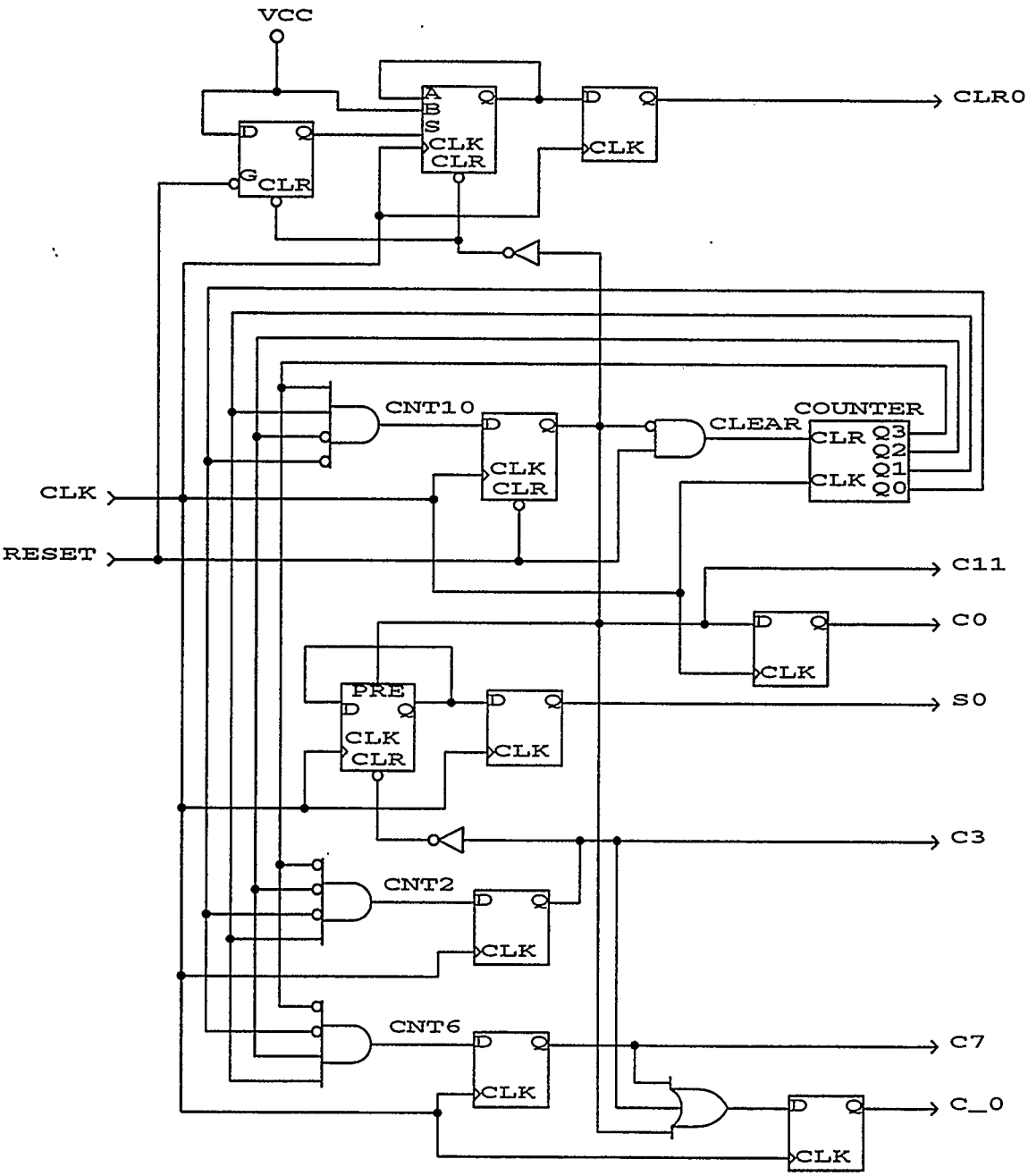


Fig. 4.26 Control Generator Cell Example

using D flip-flops to delay an existing control signal by an appropriate amount of time (as done here for *C0*). The *C_0* pulse, which is logic 1 during the LSB time of each SWL during the sample processing cycle, is generated by the logical *OR* operation being applied to the signals *C11*, *C3*, and *C7*, with the output being latched with a D flip-flop. This results in a pulse that is high at t = 0, 4, 8 (corresponding to *C0*, *C4*, and *C8*) which by definition is the *C_0* pulse.

Finally, the selection signal *S0* may be obtained by means of a D flip-flop with clear and preset inputs. By applying the *C11* pulse to the preset input, the signal *S0* will become logic 1 at t = 0, and similarly, by applying the inverted *C3* signal to the clear input the signal *S0* will become logic 0 at t = 3. A similar approach is used to obtain any selection control signal required.

4.5 Chapter Summary

This chapter has presented the TC bit-serial arithmetic hardware cells required for an FPGA implementation of a bit-serial multirate LDI Jaumann BP digital filter. The arithmetic cells required to perform the constituent operations of addition, subtraction, and multiplication have been discussed. The salient features of the modified Booth multiplier have been discussed in connection with its application within bit-serial DSP systems requiring reduced area and reduced latency. The shift register cells required to perform the operations of parallel-to-serial conversion, serial-in/serial-out data storage, downsampling, delay, state signal storage and update, and serial-to-parallel conversion have also been discussed. A representative example of a control signal generator has been presented.

CHAPTER 5

BIT-SERIAL IMPLEMENTATION OF A PRACTICAL MULTIRATE BANDPASS LDI JAUMANN DIGITAL FILTER

This chapter presents the design and bit-serial implementation of a practical multirate BP digital filter, where the design requirements include the simultaneous satisfaction of prescribed magnitude/frequency and group-delay/frequency response specifications for applications within the commercial digital CODECs. The multirate BP digital filter is designed as a combination of two LDI Jaumann digital filters, namely a 5th order LP digital filter and a 3rd order HP digital filter. The min-max type optimization satisfaction routine discussed in Chapter 2 is applied to these Jaumann digital filters in order to obtain the respective multiplier coefficient values. The $N_m = 1$ bit-serial architectures presented in Chapter 3 are adopted for the BP digital filter implementation. The bit-serial hardware cells discussed and developed in Chapter 4 are used together with the Actel 1.2μ FPGA technology for the corresponding bit-serial implementation of the resulting multirate BP digital filter. Measured magnitude/frequency and group-delay/frequency results are compared to the theoretical response characteristics.

5.1 Multirate Digital Filter Design by Optimization

The central part of a commercial digital CODEC consists of a multirate BP digital filter as shown in Fig. 5.1, where the BP digital filter is composed of a LP digital filter operating at a sample frequency of f_{SLP} and characterized by a transfer function $H_{LP}(z_{LP})$, a HP digital filter operating at a sample frequency of f_{SHP} and characterized by a transfer

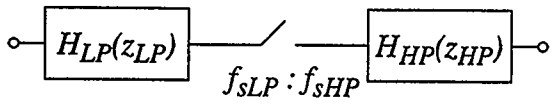


Fig. 5.1 Multirate Digital Filter Schematic Diagram

function $H_{HP}(z_{HP})$, and an f_{sLP}:f_{sHP} downsampling switch inserted between the LP and HP digital filters. Here, z_{LP}^{-1} represents the unit-delay operator associated with the sample frequency f_{sLP}, and z_{HP}^{-1} represents the unit-delay operator associated with the sample frequency f_{sHP}.

The frequency response of the LP digital filter in Fig. 5.1 can be represented as

$$H_{LP}(e^{j\Omega_{LP}}) = M_{LP}(\underline{x}_{LP}, \Omega_{LP})\, e^{j\phi(\underline{x}_{LP}, \Omega_{LP})} \tag{5.1}$$

where

$$M_{LP}(\underline{x}_{LP}, \Omega_{LP}) = \left| H(e^{j\Omega_{LP}}) \right| \tag{5.2}$$

represents the magnitude/frequency response,

$$\phi_{LP}(\underline{x}_{LP}, \Omega_{LP}) = Arg\{H(e^{j\Omega_{LP}})\} \tag{5.3}$$

represents the phase/frequency response, and

$$\tau_{LP}(\underline{x}_{LP}, \Omega_{LP}) = -\frac{1}{f_{sLP}} \frac{d\phi_{LP}(\underline{x}_{LP}, \Omega_{LP})}{d\Omega_{LP}} \tag{5.4}$$

represents the absolute group-delay/frequency response of the LP filter. Moreover, $\Omega_{LP} = 2\pi f_{LP}/f_{sLP}$ (for $0 \le \Omega_{LP} \le f_{sLP}/2$) represents the normalized real frequency variable, and $\underline{x}_{LP}$ represents the vector of the constituent LP filter multiplier coefficients.

Similarly, the frequency response of the HP digital filter can be represented as

$$H_{HP}(e^{j\Omega_{HP}}) = M_{HP}(\underline{x}_{HP}, \Omega_{LP})\, e^{j\phi(\underline{x}_{HP}, \Omega_{HP})} \tag{5.5}$$

where

$$M_{HP}(\underline{x}_{HP}, \Omega_{HP}) = \left| H(e^{j\Omega_{HP}}) \right| \tag{5.6}$$

represents the magnitude/frequency response,

$$\phi_{HP}(\underline{x}_{HP}, \Omega_{HP}) = Arg\{H(e^{j\Omega_{HP}})\} \tag{5.7}$$

represents the phase/frequency response, and

$$\tau_{HP}(\underline{x}_{HP}, \Omega_{HP}) = -\frac{1}{f_{sHP}} \frac{d\phi_{HP}(\underline{x}_{HP}, \Omega_{HP})}{d\Omega_{HP}} \tag{5.8}$$

represents the group-delay/frequency response of the HP filter. Moreover, $\Omega_{HP} = 2\pi f_{HP}/f_{sHP}$ (for $0 \le \Omega_{HP} \le f_{sHP}/2$) represents the normalized real frequency variable, and $\underline{x}_{HP}$ represents the vector of the constituent HP filter multiplier coefficients.

A 5th order LDI Jaumann digital filter operating at a sample frequency of $f_{sLP} = 32$ kHz is used to realize the LP transfer function $H_{LP}(z_{LP})$, and a 3rd order LDI Jaumann digital filter operating at a sample frequency of $f_{sHP} = 8$ kHz is used to realize the HP transfer function $H_{HP}(z_{HP})$. The SFG of the corresponding LP Jaumann digital filter is as shown in Fig. 5.2a, and the SFG of the corresponding HP Jaumann digital filter is as shown in Fig. 5.2b, where $\underline{x}_{LP} = [m_{11}, m_{12}, m_{13}, m_{21}, m_{22}]^T$ for the LP digital filter and $\underline{x}_{HP} = [h_{scale}, h_{11}, h_{21}, h_{22}]^T$ for the HP digital filter, with T denoting transpose.

The multirate BP digital filter in Fig. 5.1 is designed to satisfy prescribed magnitude/frequency and group-delay/frequency specifications similar to those required for applications within the commercial digital CODECs. Specifically, the magnitude/frequency response is desired to fall within a certain tolerance region characterized by lower and upper bounds, and to satisfy a pair of equality constraints, namely 0 dB magnitude at

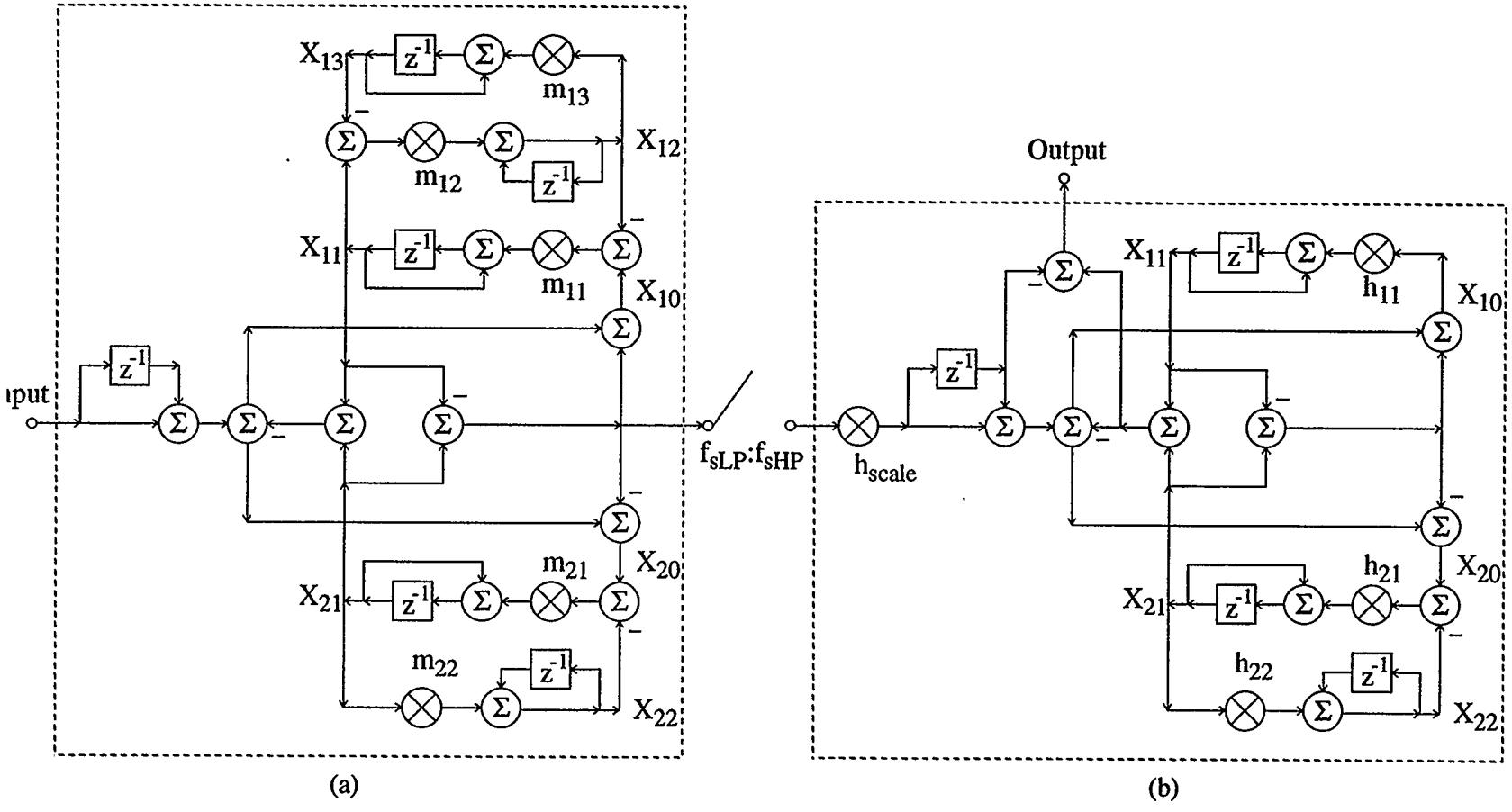


Fig. 5.2 Multirate LDI Jaumann Digital Filter Signal Flow-Graph

(a) 5th Order LP (b) 3rd Order HP

1000 Hz and ∞ dB attenuation at 60 Hz. In addition, the group-delay/frequency response is desired to satisfy a pair of inequality constraints, namely relative group-delays of less than 280 μs at the frequencies 400 Hz and 3200 Hz. In order to achieve these design requirements, the following minimization is applied to the BP digital filter.

$$\begin{aligned} &\underset{\underline{x}}{\text{minimize}}\ e(\underline{x}) \\ &\text{subject to:} \quad M(\underline{x}, 2\pi \cdot 1000/32000) = 1 \\ &\qquad\qquad\ M(\underline{x}, 2\pi \cdot 60/32000) = 0 \\ &\qquad\qquad\ \tau(\underline{x}, 2\pi \cdot 400/32000) - \tau_0 \le 280\ \mu s \\ &\qquad\qquad\ \tau(\underline{x}, 2\pi \cdot 3200/32000) - \tau_0 \le 280\ \mu s \end{aligned} \tag{5.9}$$

where $\underline{x} = [\underline{x}_{LP}^T | \underline{x}_{HP}^T]^T$, and where

$$e(\underline{x}) = \underset{0 \le \Omega \le \pi}{\text{Max}} \begin{cases} \dfrac{M_l(\Omega) - M(\underline{x},\Omega)}{M_l(\Omega)} & \text{if } M(\underline{x}, \Omega) \le \dfrac{M_l(\Omega) + M_u(\Omega)}{2} \\ \dfrac{M(\underline{x}, \Omega) - M_u(\Omega)}{M_u(\Omega)} & \text{if } M(\underline{x}, \Omega) \ge \dfrac{M_l(\Omega) + M_u(\Omega)}{2} \end{cases} \tag{5.10}$$

with $M_l(\Omega)$ and $M_u(\Omega)$ representing the lower and upper bounds of the prescribed magnitude/frequency response tolerance region, respectively. Moreover,

$$M(\underline{x}, \Omega) = M_{LP}(\underline{x}_{LP}, \Omega_{LP})\, M_{HP}(\underline{x}_{HP}, \Omega_{HP}) \tag{5.11}$$

and

$$\tau(\underline{x}, \Omega) = \tau_{LP}(\underline{x}_{LP}, \Omega_{LP}) + \tau_{HP}(\underline{x}_{HP}, \Omega_{HP}) \tag{5.12}$$

where

$$\tau_0 = \underset{2\pi \cdot 400/32000 \le \Omega \le 2\pi \cdot 3200/32000}{\text{Min}} \{\tau(\underline{x}, \Omega)\} \tag{5.13}$$

The minimization problem in Eqn. (5.9) contains both equality and inequality constraints, making a corresponding numerical optimization complicated. In order to simplify matters, this minimization can be replaced by the following equivalent form which con-

sists of inequality constraints only.

$$\begin{aligned} &\underset{\underline{x}}{\text{minimize}}\ E(\underline{x}) \\ &\text{subject to:}\quad 1 - M(\underline{x}, 2\pi \cdot 1000/32000) \le 0 \\ &\qquad\qquad \tau(\underline{x}, 2\pi \cdot 400/32000) - \tau_0 \le 280\ \mu s \\ &\qquad\qquad \tau(\underline{x}, 2\pi \cdot 3200/32000) - \tau_0 \le 280\ \mu s \end{aligned} \tag{5.14}$$

where

$$E(\underline{x}) = max\{e(\underline{x}), M(\underline{x}, 2\pi \cdot 60/32000), M(\underline{x}, 2\pi \cdot 1000/32000) - 1\} \tag{5.15}$$

and where $e(\underline{x})$ has been defined in Eqn. (5.10).

The min-max type optimization satisfaction method in [26] is used to obtain the values of the multiplier coefficients (infinite precision) in the vector $\underline{x}$ by solving the minimization problem in Eqn. (5.14). The quantization of the optimized multiplier coefficients to finite precision values is discussed in the following section.

5.2 Non-Ideal Finite Precision Effects

The successful design of a digital filter relies on satisfying the given magnitude/frequency and group-delay frequency response specifications in the presence of the non-ideal finite-precision arithmetic effects. These finite-precision arithmetic effects include the transfer function errors introduced by multiplier coefficient quantization, the large amplitude errors due to internal signal overflow, and the small amplitude errors (limit cycles) due to internal signal quantization. This section will present a discussion of the non-ideal effects due to finite precision arithmetic along with the measures taken to offset them.

5.2.1 Multiplier Coefficient Quantization

In a fixed-point digital filter implementation, the values of the constituent multiplier coefficients must be quantized to a finite wordlength. The quantized multiplier coefficients

give rise to an inevitable departure from the original infinite precision magnitude/frequency and group-delay response characteristics, as the poles and zeroes of the original transfer function move to new locations in the z-plane. If the digital filter is sensitive to these quantization errors, then the required response specifications may no longer be satisfied after quantization.

The infinite-precision multiplier coefficient values obtained from the optimization process were subsequently quantized to a minimum finite precision length while ensuring that the desired magnitude/frequency and group-delay/frequency response specifications remain satisfied by the BP digital filter. Due to the exceptionally low passband sensitivity of the constituent LDI Jaumann digital filters, it is possible to quantize the multiplier coefficient values to 6 bits for the LP Jaumann filter, and to 8 bits for the HP Jaumann filter, as given in Table 5.1.

Lowpass Filter (6 bits)		Highpass Filter (8 bits)	
m_{11}	0.25000	h_{11}	0.1015625
m_{12}	0.46875	h_{21}	0.0312500
m_{13}	0.78125	h_{22}	0.5078125
m_{21}	0.21875	h_{scale}	1.0312500
m_{22}	1.37500		

Table 5.1 Quantized Multiplier Coefficient Values

It should be noted that the multiplier coefficient m_{22} will be realized in the hardware implementation as 2×0.6875, and the multiplier coefficient h_{scale} as 2×0.506625.

5.2.2 System Wordlength Determination

In the design of fixed-point IIR digital filters, it is extremely important to ensure that during normal operation the digital filter does not suffer from the harmful effects of internal signal overflow or excessive round-off errors. In order to prevent the input signal to a fixed-point digital filter from exceeding the dynamic range of the filter, some form of signal scaling must be performed. This scaling may take the form of a multiplier at the filter input, which is used to reduce the input signal size sufficiently so that the internal signal amplitudes may all be correctly represented in the available number of bits. However, in order to prevent the requirement of an extra multiplication operation, it is possible to increase the SWL by padding the input signal with overflow guard bits at the MSB end to allow the maximum internal signal size to be correctly represented.

Similarly, the effects of multiplier round-off error need to be considered. When the $(n+m)$-bit product is rounded to n-bits, a signal quantization error occurs. Since the signal output of the multiplier is fed back to other components in a IIR filter, these errors can accumulate and become significant. In order to prevent this from happening, the input signal can be padded with round-off error guard bits at the LSB end.

5.2.3 Determination of Overflow and Round-Off Error Guard Bit Requirements

By using TC arithmetic, the intermediate signal overflows do not introduce an error in the overall output signal provided that the signals at the multiplier inputs as well as the signal at the filter output can be represented correctly in the available SWL.

An absolute bound on the growth of an amplitude-limited signal from the digital filter input to the respective multiplier inputs can be determined in terms of the L1-norm of the

impulse response from the digital filter input to the respective multiplier inputs. The maximum value of the signal bound at the multiplier inputs is used to determine the necessary number of signal overflow guard bits. If a maximum bound of x is obtained (associated with the signal at a specific multiplier input), then to guarantee that an amplitude-limited input signal does not cause harmful signal overflow effects, the SWL must include $\lceil log_2(x) \rceil$ upper guard bits.

For the 5th order LP Jaumann digital filter, the L1-norm of 5.530 from the LP filter input to the input of the multiplier m_{11} (c.f. Table 5.2) represents the maximum internal signal gain. Thus, to guarantee that an amplitude-limited input signal does not cause harmful signal overflow effects, the LP filter SWL must include $\lceil log_2(5.530) \rceil = 3$ overflow guard bits. For the 3rd order HP Jaumann digital filter, the L1-norm of 7.804 from the HP filter input to the input of the multiplier h_{21} (c.f. Table 5.3) represents the maximum internal signal gain. Thus, to guarantee that an amplitude-limited input signal does not cause harmful signal overflow effects, the HP filter SWL must include $\lceil log_2(7.804) \rceil = 3$ upper guard bits.

Gain from filter input to multiplier input		Gain from multiplier output to filter output	
m_{11}	5.530	m_{11}	3.802
m_{12}	5.263	m_{12}	3.960
m_{13}	3.486	m_{13}	2.850
m_{21}	4.684	m_{21}	3.155
m_{22}	2.544	m_{22}	1.120
output	2.026		
Max Gain	5.530	Σ Gains	14.887

Table 5.2 L1-norm Results For LP Filter

The L1-norm may also be used to determine an upper bound on the error in the output signal due to the errors caused by the inherent quantization incurred at the multiplier outputs. By calcu-

Gain from filter input to multiplier input		Gain from multiplier output to filter output	
h_{scale}	2.000	h_{scale}	2.878
h_{11}	3.706	h_{11}	4.923
h_{21}	7.804	h_{21}	20.530
h_{22}	1.321	h_{22}	5.087
output	2.968		
Max Gain	7.804	$\sum$ Gains	33.418

Table 5.3 L1-norm Results for HP Filter

lating the gain from each multiplier output to the filter output, and assuming worst case additive gains, one can find an upper bound on the quantization error. If quantization is achieved by rounding, the bound is then multiplied by 0.5 to obtain the maximum error introduced. If quantization is achieved by truncation, then the bound is left unchanged to obtain the maximum error introduced. If the sum of the gains from multiplier output nodes to the filter output is x, then $\lceil log_2(x) \rceil$ lower guard bits are required to limit the quantization noise in the output signal to an equivalent value of one-half bit.

For the LP Jaumann digital filter, the sum of the gains from the multiplier outputs to the filter output is 14.870 (c.f. Table 5.2), and the gain from filter input to filter output is 2.026. Since the multiplier outputs are obtained through a rounding process, and since the filter output is obtained through a truncation process, $\lceil log_2(14.870 \times 0.5 + 2.026) \rceil = 4$ round-off guard bits are required to limit the quantization noise in the output signal to an equivalent value of one-half bit. For the HP Jaumann digital filter, the sum of the gains from the multiplier outputs to the filter output is 30.540 (c.f. Table 5.3), while the gain

from filter input to filter output is 2.968, leading to a requirement of $\lceil log_2 (33.418 \times 0.5 + 2.968) \rceil = 5$ round-off guard bits to limit the quantization noise in the output signal to an equivalent value of one-half bit.

In accordance with the above considerations, the minimum SWL for the LP Jaumann digital filter due to scaling considerations is obtained as $W_s = 2+3+12+4 = 21$ bits, accommodating a 12-bit input signal, and 2 sign extension bits (as required by the modified Booth multiplier). Similarly, the minimum SWL for the HP Jaumann digital filter is obtained as $W_s = 2+3+12+5 = 22$ bits.

5.3 Selecting a Bit-Serial Architecture

The Actel ACT^{TM}2 1.2μ FPGA technology was chosen to demonstrate the viability of a hardware realization of the multirate BP digital filter. This choice was made after the considerations of cost, available design tools, and chip programming facilities. The A1280 is the largest Actel FPGA currently available, and allows implementations including up to 8000 gate-array gates. These gate-array gates are available in the form of 1232 programmable logic modules, where two types of modules are available. They are combinatorial logic C-modules and sequential logic S-modules. The A1280 FPGA also offers two high-drive low-skew dedicated clock networks.

After a preliminary analysis regarding the sample rate and the gate-count requirements for the BP LDI Jaumann digital filter, a bit-serial architecture employing a single multiplexed multiplier $(N_m = 1)$ for each of the constituent LP and HP digital filters was chosen. This architecture imposes a minimum SWL due to hardware latency of

$$W_l = L_m + 2 = (\frac{3 \cdot m}{2} + 1) + 2 = \frac{3 \cdot m}{2} + 3$$

which leads to $W_l = 12$ for the LP filter, and $W_l = 15$ for the HP filter, and where L_m has been taken as the latency of the modified Booth multiplier. Since $W_s = 21$ for the LP filter and $W_s = 22$ for the HP filter, the minimum SWL for the LP filter is $W_{min} = 21$, and the minimum SWL for the HP filter is $W_{min} = 22$ (c.f. Eqn. (3.2)). Furthermore, the LP filter is required to have a sample rate of 32 kHz, and the HP filter a sample rate of 8 kHz. In order to avoid the need for two separate clock signal generators, a SWL of $W = 22$ bits is selected for the LP filter. This will allow the HP filter to operate at a clock rate of $f_{clkHP} = f_{clkLP}/4$, which may be achieved simply through the use of a divide-by-4 circuit.

The above considerations lead to the LP digital filter operating at a bit-clock rate of $f_{clkLP} = 22 \times 5 \times 32$ kHz $= 3.52$ MHz, and the HP digital filter operating at a bit-clock rate of $f_{clkHP} = 0.84$ MHz.

For the LP LDI Jaumann digital filter $n = 5$, $n_1 = 3$, and $n_2 = 2$; and for the HP LDI Jaumann digital filter $n = 3$, $n_1 = 1$, and $n_2 = 2$. Using the state update pseudocode from Fig. 3.5, the set of state update equations:

$$\begin{aligned} X_{12} &= [X_{11} - X_{13}] * m_{12} + X_{12} \\ X_{22} &= [X_{21} - \mathrm{GND}] * m_{22} + X_{22} \\ X_{21} &= [X_{10} - X_{12}] * m_{11} + X_{11} \\ X_{21} &= [X_{20} - X_{22}] * m_{21} + X_{21} \\ X_{13} &= [X_{12} - \mathrm{GND}] * m_{13} + X_{13} \end{aligned} \qquad 5.16$$

are obtained for the 5th order LP Jaumann digital filter, while the set of equations

$$
\begin{aligned}
Input &= Input * h_{scale} \\
X_{22} &= [X_{21} - \mathrm{GND}] * h_{22} + X_{22} \\
X_{11} &= [X_{10} - \mathrm{GND}] * h_{11} + X_{11} \\
X_{21} &= [X_{20} - X_{22}] * h_{21} + X_{21}
\end{aligned}
\qquad 5.17
$$

are obtained for the 3-rd order HP Jaumann digital filter. These equations are subsequently assigned to hardware in the corresponding LP and HP bit-serial architectures using the methods described in Ch. 3. The schematic diagrams for the resulting bit-serial LP and HP LDI Jaumann digital filters are shown in Fig. 5.3 and Fig. 5.4, respectively.

For the LP LDI Jaumann digital filter schematic diagram in Fig. 5.3, the 12-bit serial TC digital input signal will enter the circuit at *INPUT*, the 6-bit multiplier coefficient will enter at *COEFF*, and the 22-bit output signal will exit at *OUT*. The remaining input signals are for control purposes. The operation of the LP digital filter circuit is as follows.

Following an initialization period during which all data paths are cleared, the LSB of an input sample will arrive at *INPUT* at $t_{LP} = 4$. This input signal is then routed through a 4:1 multiplexor and a D flip-flop combination to facilitate the padding of the 12-bit input data signal to include 5 lower guard-bits and 7 upper guard-bits. Furthermore, the input signal also arrives at the input to *SHREG12* at $t_{LP} = 4$, where the assertion of the control signal *S4_15* will cause the previous input sample to shift out while the present input sample shifts in. The previous sample is also routed through a 4:1 multiplexor and a D flip-flop combination to facilitate the padding of the signal to the 22-bit format, with the 2:1 multiplexor and the D flip-flop combination above *SHREG12* latching the sign bit of the previous sample to allow the required sign extension.

The above operations cause the LSBs of the 22-bit data signals representing the present input sample and the past input sample to arrive at the top left *ADD* cell at $t_{LP} = 0$.

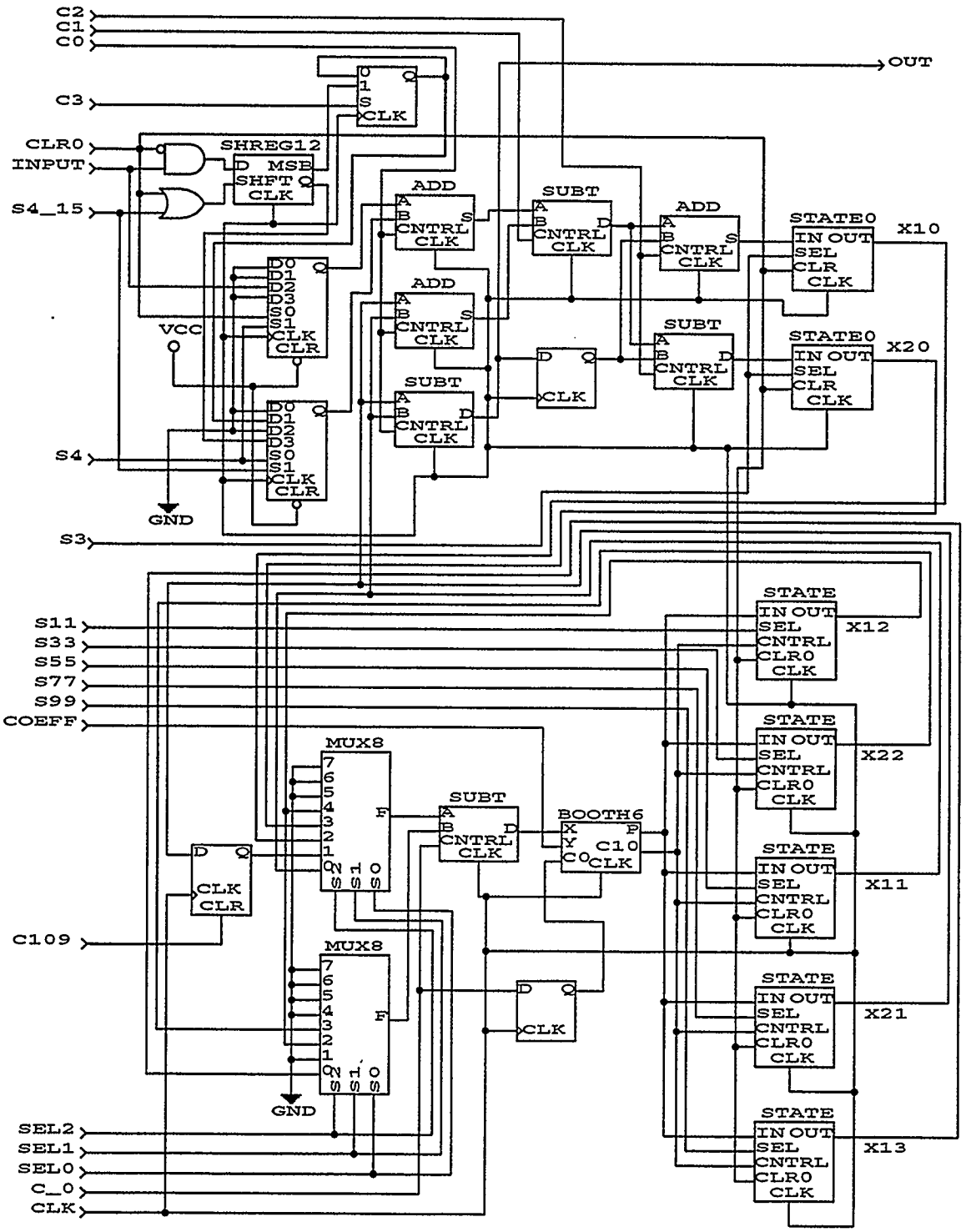


Fig. 5.3 Schematic Diagram for Bit-Serial LP LDI Jaumann Digital Filter

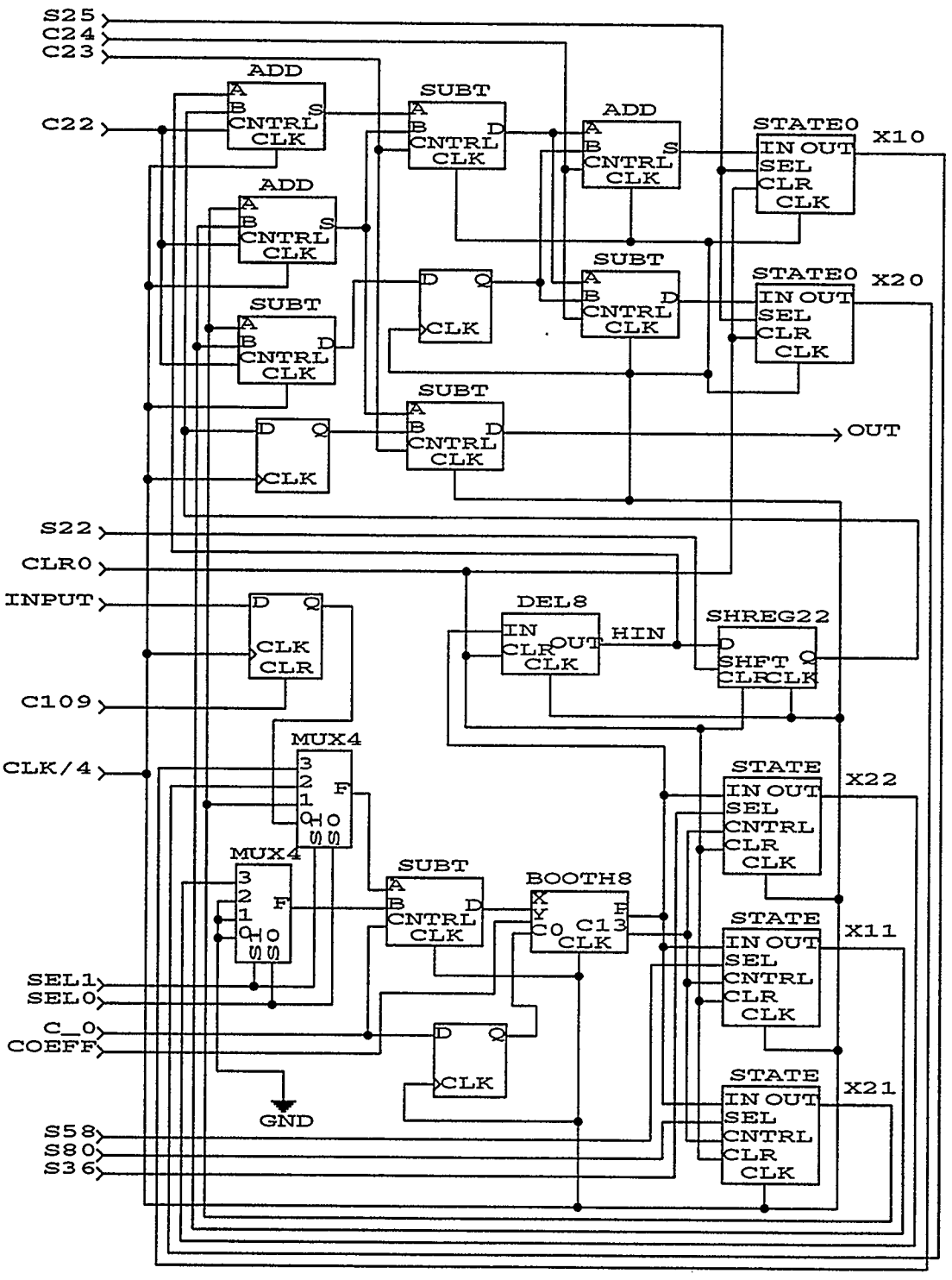


Fig. 5.4 Schematic Diagram for Bit-Serial HP LDI Jaumann Digital Filter

Moreover, at $t_{LP} = 0$ the LSBs of the signals *X11* and *X21* will arrive at the *ADD* and *SUBT* cells directly below the top left *ADD* cell. The present input sample is processed together with the previous input sample as well as the signals *X11* and *X21* to produce the LSB of the output signal at $t_{LP} = 1$, and the LSBs of the signals *X10* and *X20* at $t_{LP} = 22$. The *STATE0* cells will recirculate the signals *X10* and *X20* in such a manner that the LSBs of these signals become available at the future times $t_{LP} = 44, 66, 88, 0$.

The remaining bottom half of the LDI Jaumann digital filter in Fig. 5.3 performs the state update operations in the order given by Eqn. 5.16. Starting at $t_{LP} = 0$, the upper *MUX8* cell will select the signal *X11* and the lower *MUX8* cell will select the signal *X13*. These signals will enter the *SUBT* cell, the output of which arrives at the *BOOTH6* cell together with the LSB of the multiplier coefficient signal *m12* at $t_{LP} = 1$. The LSB of the product will enter the top *STATE* cell at $t_{LP} = 11$, where it will be added to the previous *X12* signal. At $t_{LP} = 22$ two times the signal *X21* will be selected by the top *MUX8* cell, while the bottom *MUX8* cell will select GND. These two signals are then subtracted with the result arriving at the multiplier together with the LSB of the coefficient signal *m22* at $t_{LP} = 23$. At $t_{LP} = 33$, the product will enter the *STATE* cell associated with *X22*, where the state update operation will be performed. This process will continue in a similar manner with the state update operations for the signals *X11*, *X21*, and *X13*. At $t_{LP} = 0$ of the next sample processing period the entire process will start all over.

For the HP LDI Jaumann digital filter in the schematic diagram in Fig. 5.4, the serial 22-bit TC digital input signal will enter the circuit at *INPUT*, the 8-bit multiplier coefficient will enter the circuit at *COEFF*, and the 22-bit output will exit at *OUT*. The remainder of the input signals are for control purposes.

Following an initialization period during which all data paths are cleared, the LSB of the downsampled LP filter output will arrive at *INPUT* at $t_{HP} = 0$. This input is then passed through a D flip-flop to facilitate a multiply-by-2 operation, prior to being selected by the top *MUX4* cell as an input to the *SUBT* cell. The bottom *MUX4* cell selects GND as the other input to the *SUBT* cell, resulting in the shifted input signal arriving at the *BOOTH8* multiplier cell together with the multiplier coefficient signal *hscale* at $t_{HP} = 1$. At $t_{HP} = 14$, the product will arrive at the *DEL8* cell, which will cause the signal *HIN* to have its LSB available at $t_{HP} = 22$. The present *HIN* signal will then enter the top left *ADD* cell together with the previous *HIN* signal (currently stored in *SHREG22*). Also at $t_{HP} = 22$, the LSBs of the signals *X11* and *X21* will enter the *ADD* and *SUBT* cells below the top left *ADD* cell. The signal *HIN* and its previous sample counterpart, together with the signals *X11* and *X21*, will be processed to give the LSB of the output signal *OUT* at $t_{HP} = 23$, and the LSBs of the signals *X10* and *X20* at $t = 44$, where the *STATE0* cells will recirculate the *X10* and *X20* signals in such a manner as to provide their respective LSBs at the future times of $t_{HP} = 66, 88, 0, 22$ (where $t_{HP} = 0, 22$ belong to the next sample period).

The remaining bottom half of the HP Jaumann digital filter in Fig. 5.4 performs the state update operations in the order given by Eqn. 5.17. Starting at $t_{HP} = 22$, the upper *MUX4* cell will select the signal *X21* and the lower *MUX4* cell will select GND. These signals will enter the *SUBT* cell, the result of which together with the LSB of the multiplier coefficient signal *h22* will arrive at the *BOOTH6* cell at $t_{HP} = 23$. The LSB of the product will enter the top *STATE* cell at $t_{HP} = 36$, where it will be added to the previous *X22* signal to complete the state update operation. At $t_{HP} = 44$ the signal *X10* will be selected by the

top multiplexor, while the bottom multiplexor will select GND. These two signals are then subtracted, with the result and the LSB of the coefficient signal *m22* arriving at the multiplier at $t_{HP} = 45$. At $t_{HP} = 58$, the product will enter the *STATE* cell associated with *X11* and the state update operation will be completed. This process will continue in a similar manner with the state update operations for *X21*. Finally, during the $t_{HP} = 88$ to $t_{HP} = 0$ period the multiplier is in an idle mode with its output being ignored.

5.4 Digital Filter System

The digital filter system will require an analog-to-digital (A/D) converter to provide a digital input signal to the BP digital filter, an EPROM to provide the multiplier coefficients, and a digital-to-analog (D/A) converter followed by a reconstruction filter to provide an analog output signal, as shown in Fig. 5.5.

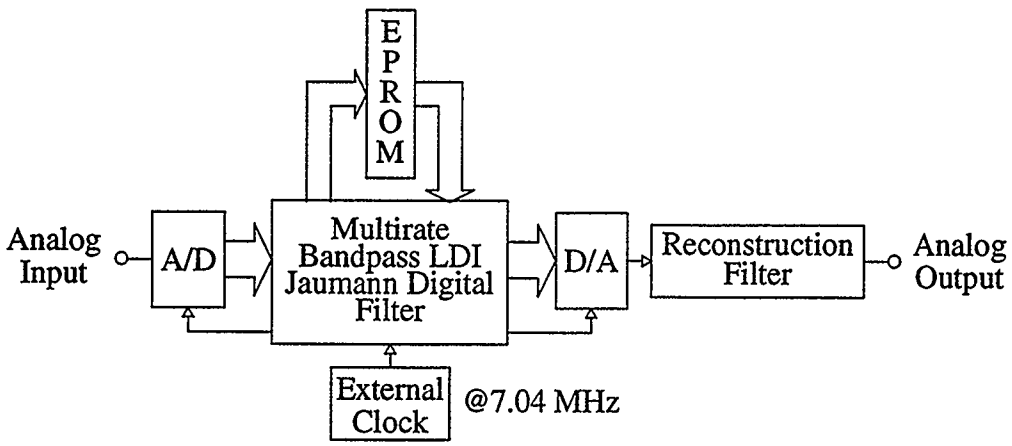


Fig. 5.5 Multirate Bandpass Digital Filter System

The schematic diagram for the constituent multirate BP LDI Jaumann digital filter is as shown in Fig. 5.6, where the input and output nodes correspond to input and output pads in the constituent FPGA chip. The operation of the multirate BP LDI Jaumann digital filter is discussed in the following.

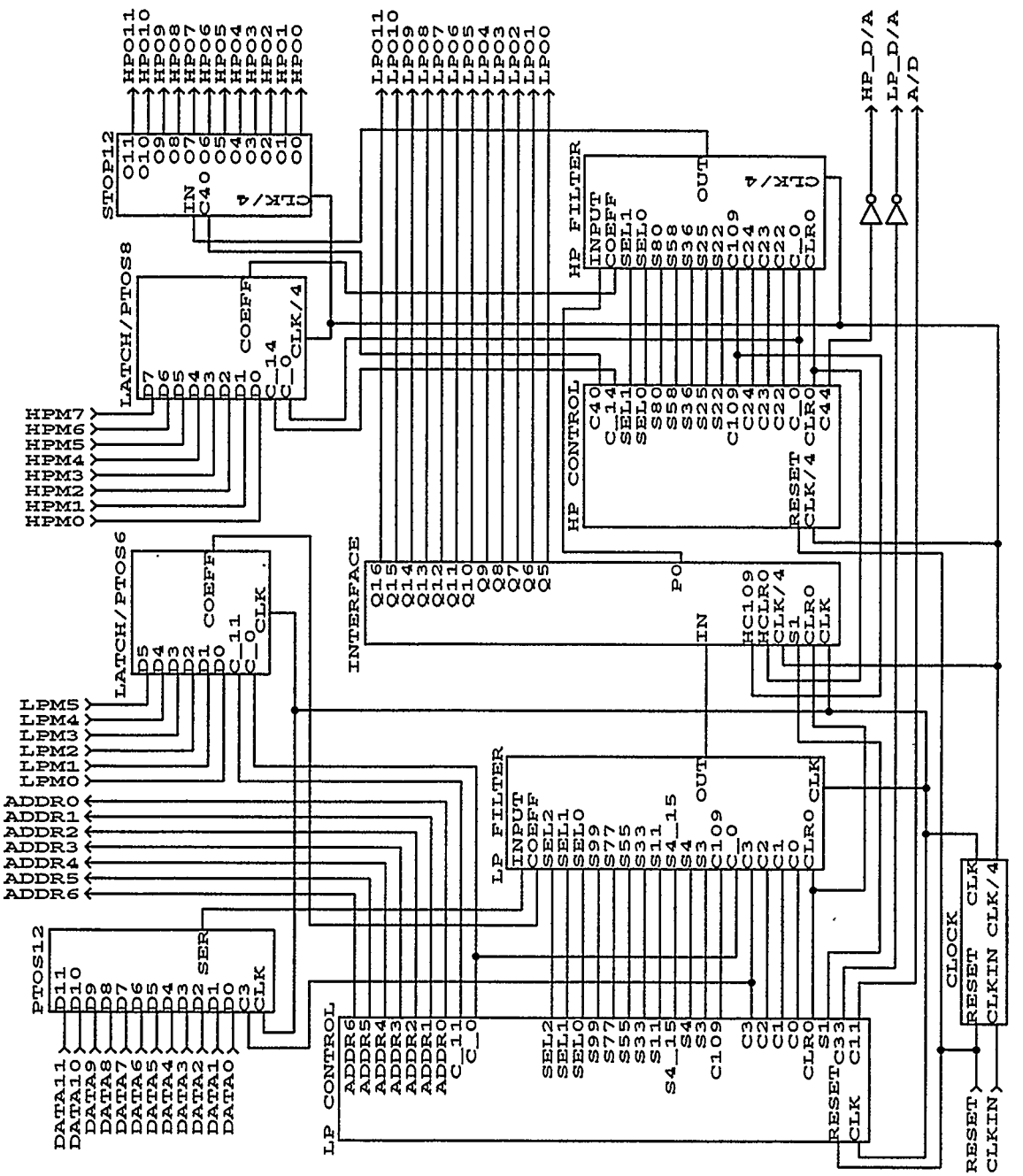


Fig. 5.6 Multirate Bandpass Digital Filter

A logic low signal applied to the *RESET* input will clear the D flip-flops in the *CLOCK* cell (see Fig. 5.7), where a divide-by-two circuit will provide the LP bit-clock, and a divide-by-eight circuit will provide the HP bit-clock. This reset signal is also used to clear the LP control generator (*LP CONTROL*) and the HP control generator (*HP CONTROL*) counters, and all D flip-flop chains in the circuit (via the respective LP and HP *CLR0* signals). When the signal applied to the *RESET* input returns to logic high, the generation of all required control signals will start. It should be noted that because the LP control generator operates at 4 times the bit-clock rate as the HP control generator, a pair of signals having the same name in the LP and HP circuits will signify distinct signals (e.g. the control signal *SEL0* in *LP CONTROL* is distinct from the signal *SEL0* in *HP CONTROL* despite the fact that both signals have the same name).

At $t_{LP} = 11$, the analog input signal will be sampled by the A/D converter. This is achieved by the LP control generator supplying a convert signal to the A/D. The resulting 12-bit TC digital input signal from the A/D converter will arrive at *DATA11* through *DATA0* of the parallel-to-serial shift register *PTOS12*. The EPROM data will be selected by the address signals *ADDR6* through *ADDR0*, which directly correspond to the count of

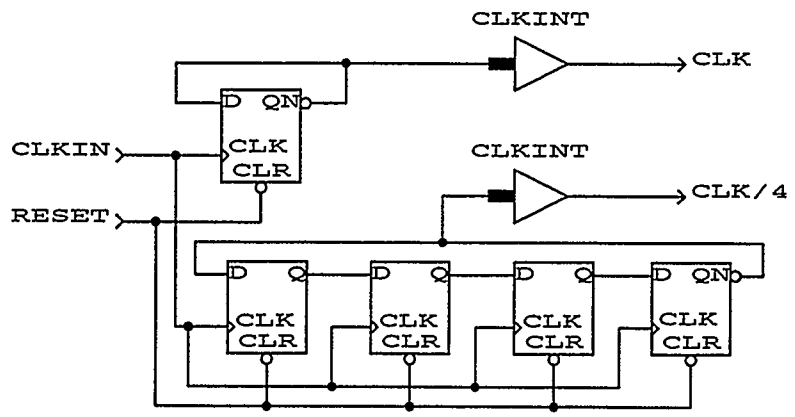


Fig. 5.7 CLOCK Cell

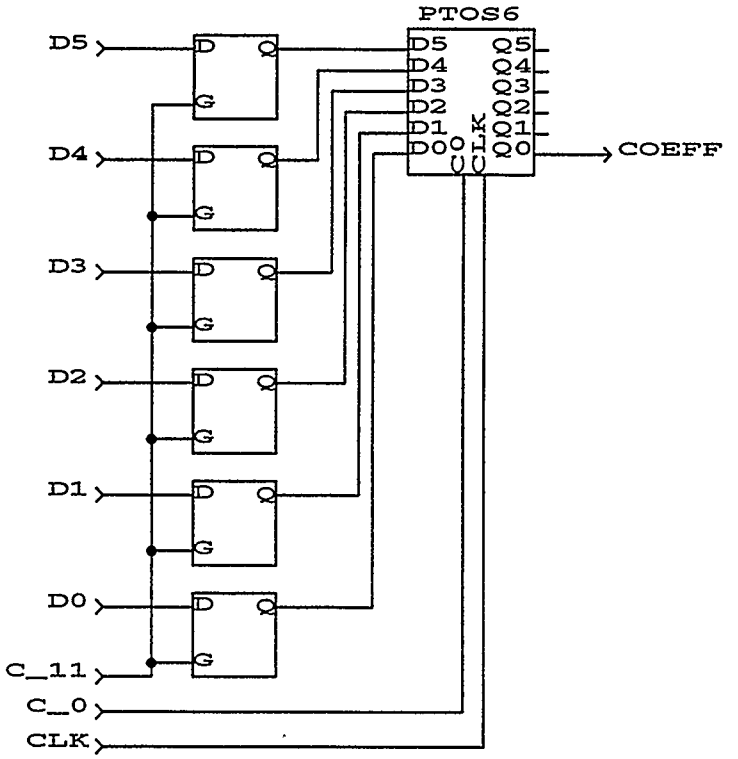


Fig. 5.8 LATCH / PTOS6 Cell

the LP control generator counter. This EPROM data will represent the 6-bit LP filter multiplier coefficient signals which will arrive at *LPM5* through *LPM0*, and the 8-bit HP filter multiplier coefficient signals, which will arrive at the inputs *HPM7* through *HPM0*. In both cases, the multiplier coefficient data will be latched as the input to a parallel-to-serial shift register by the signals *C_11* (LP) and *C_14* (HP), where the *LATCH/PTOS6* cell is shown in Fig. 5.8. The application of the *C_0* (LP) signal to the *LATCH/PTOS6* cell will result in the LP coefficient data arriving at the LP filter multiplier at times corresponding to the control signals *C1*, *C23*, *C45*, *C67*, and *C89*. Similarly, the application of the *C_0* (HP) signal to the *LATCH/PTOS8* cell will result in the HP coefficient data arriving at the HP filter multiplier at times corresponding to the control signals *C1*, *C23*, *C45*, and *C67*.

The assertion of the *C3* signal to the *PTOS12* circuit will result in the LP filter receiv-

ing the LSB of the 12-bit input signal at $t_{LP} = 4$. The output from the LP filter will enter the *INTERFACE* cell, with the control signal *S1* selecting the starting time of entry. Every fourth time an output from the LP filter enters this cell, the HP control signal *C109* will latch the output into a serial output shift register, to be subsequently processed by the HP filter. In addition, *LPO11* through *LPO0* represent the output digital signal from the LP filter., and the corresponding D/A convert signal *LP_D/A* is provided. The HP filter will provide an output to the serial-to-parallel output shift register *STOP12*, where the upper and lower guard bits are ignored and the 12-bit digital output signal is latched. At $t_{HP} = 44$, a convert signal *HP_D/A* is applied to the D/A converter to produce an analog output signal. This output signal is then processed by the reconstruction filter. The process described above will repeat continuously until the next application of the reset signal.

5.5 Actel FPGA Implementation

The constituent arithmetic and shift register cells were simulated using ViewLogic (PowerView and WorkView) schematic capture tools. The operation of the multirate BP digital filter was verified at the gate-level through an impulse response simulation.

The steps in realizing the design in Actel FPGA technology are summarized below.

1. Specify and fix Input/Output pads to pin locations.
2. Design verification (i.e. fanout, module-count, etc.)
3. Automatic place and route.
4. Back-annotated delay information.
5. Repeat impulse response simulation using technology representative delays.
6. Program the FPGA.

The FPGA is then tested, with the results being discussed in the following section.

5.6 Measured Magnitude/Frequency and Group-Delay/Frequency Characteristics

This section presents the methodology for measuring the magnitude/frequency and group-delay/frequency response characteristics of the multirate BP LDI Jaumann digital filter together with the corresponding results.

The multirate BP LDI Jaumann digital filter system shown in Fig. 5.5 is associated with a magnitude/frequency response $M_S(f)$, where

$$M_S(f) = M'_S(f)\,M_{\mathrm{BP}}(f), \tag{5.18}$$

and where $M'_S(f)$ represents the magnitude/frequency response of the overall system excluding the BP digital filter, and where $M_{BP}(f)$ represents the magnitude/frequency response associated with the BP digital filter itself. Therefore, the magnitude/frequency response of the BP digital filter may be obtained in accordance with

$$M_{BP}(f) = \frac{M_S(f)}{M'_S(f)}. \tag{5.19}$$

The magnitude/frequency response of the BP digital filter is shown in Fig. 5.9, where an enlarged passband magnitude/frequency response is shown in Fig. 5.10, and where an enlarged lower-stopband magnitude/frequency response is shown in Fig. 5.11. In these figures, the simulated response associated with quantized multiplier coefficients is represented by solid curves, the tolerance region is represented by dashed curves, and the measured response characteristic is represented by diamonds. In all cases, the response has been preshaped by the required factor of $\mathrm{sinc}^3(\omega/32000)$. The measured results are virtually identical to the simulation results, and yield a gain of 1.00284 {{0.0246 dB}} at 1000 Hz, corresponding to an error of 0.284 %. The measured results also demonstrate a

loss of 44.3 dB at 60 Hz.

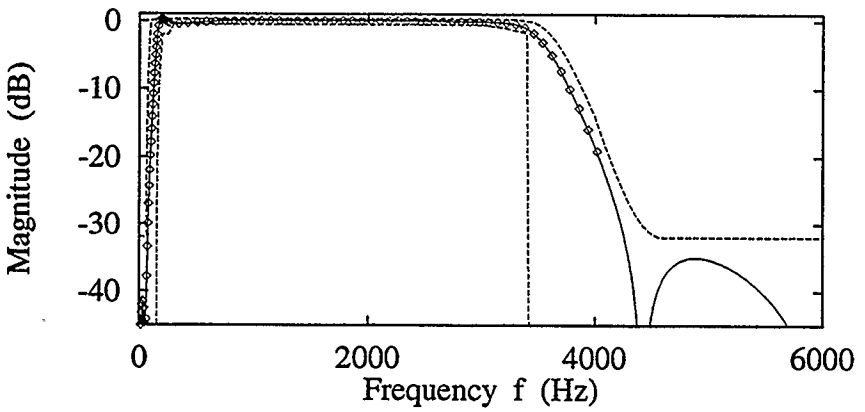


Fig. 5.9 Magnitude/Frequency Response

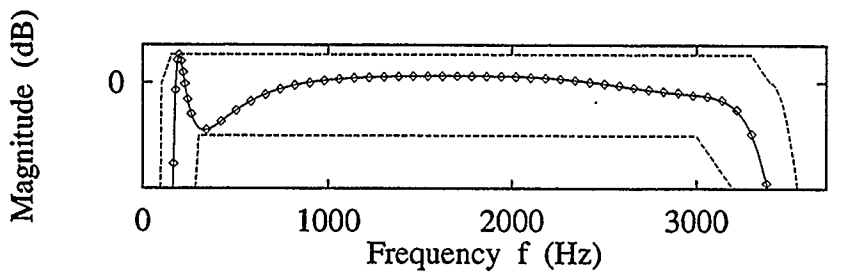


Fig. 5.10 Passband Magnitude/Frequency Response

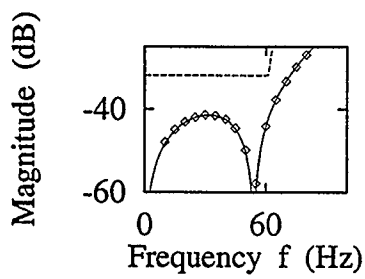


Fig. 5.11 Lower Stopband Magnitude/Frequency Response Characteristic

The multirate BP LDI Jaumann digital filter system shown in Fig. 5.5 is associated with a phase/frequency response $\phi_s(\Omega)$ where

$$\phi_s(\Omega) = \phi'(\Omega) + \phi_{BP}(\Omega), \tag{5.20}$$

and where $\phi'_s(\Omega)$ represents the phase/frequency response of the overall system excluding the BP digital filter, and where $\phi_{BP}(\Omega)$ represents the magnitude/frequency response associated with the BP digital filter itself. Therefore, the phase/frequency response of the BP digital filter may be obtained in accordance with

$$\phi_{BP}(\Omega) = \phi_s(\Omega) - \phi'_s(\Omega). \tag{5.21}$$

Then, the absolute group-delay/frequency response can be approximated as

$$\tau_{BP}(\Omega) \approx -\frac{1}{f_s}\frac{\Delta\phi_{BP}(\Omega)}{\Delta\Omega} \tag{5.22}$$

where $\Delta\phi_{BP}(\Omega)/\Delta\Omega$ is found by measuring the change in phase at two frequency points spaced by small $\Delta\Omega$. Furthermore, the relative group-delay/frequency response may be obtained in accordance with

$$\tau_{rel}(\Omega) = \tau_{BP}(\Omega) - \tau_{0BP}, \tag{5.23}$$

where τ_{0BP} is the minimum value attained by $\tau_{BP}(\Omega)$ in the passband.

The relative group-delay/frequency response is shown in Fig. 5.12, where the simulated response incorporating quantized multiplier coefficients is indicated by the solid curve, and the maximum relative group-delay specification of 280 μs in the 400 Hz through 3200 Hz band is indicated by the dashed curve. The simulated response demonstrates τ_{rel} = 265 μs at 400 Hz, and τ_{rel} = 252 μs at 3200 Hz. The phase/frequency response was measured only in the vicinity of these two critical frequencies. Furthermore, the phase meter used for these measurements was not able to accurately measure the phase, i.e. the errors in phase were possibly in the order of a few degrees. The phase measurements were taken in the vicinity of 400, 3200 and 1500 Hz, where the frequency

Ω_{min} (in Hz) was known to lie in the vicinity of 1500 Hz. Using graphical analysis to calculate the group-delay (linear best fit line), resulted in the measured values of τ_{rel} = 306 μs at 400 Hz, and τ_{rel} = 259 μs at 3200 Hz. These values are certainly within a reasonable range of the expected values when the measurement equipment, and human error factors are taken into consideration. As a further item of interest, using plots of the simulated phase/frequency response, the same method was used for the graphical determination of the group-delay values, and led to the corresponding results of τ_{rel} = 293 μs at 400 Hz, and τ_{rel} = 290 μs at 3200 Hz.

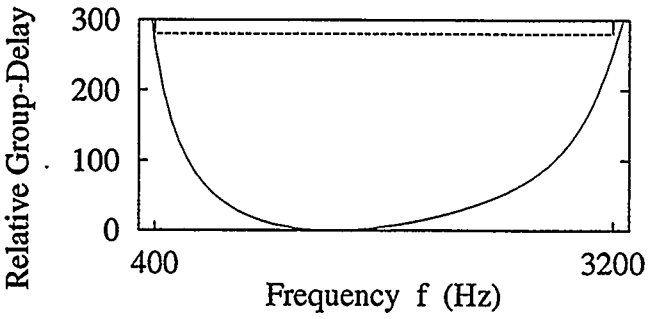


Fig. 5.12 Relative Group-Delay/Frequency Response Characteristic

5.7 Chapter Summary

This chapter has presented the design and bit-serial implementation of a practical multirate LDI Jaumann BP digital filter satisfying desired magnitude/frequency and group-delay/frequency response specifications. The multirate BP digital filter has been designed as a combination of two LDI Jaumann digital filters, namely a 5th order LP digital filter and a 3rd order HP digital filter. The respective multiplier coefficient values were obtained by applying the min-max type optimization satisfaction routine discussed in Chapter 2 to these Jaumann digital filters. The N_m = 1 bit-serial architectures presented in

Chapter 3 have been used for the corresponding LP and HP LDI Jaumann digital filter implementations. The bit-serial arithmetic hardware cells, including the modified Booth multiplier and the shift register cells discussed and developed in Chapter 4, have been realized within the Actel 1.2μ FPGA technology for the bit-serial implementation of the resulting multirate Jaumann BP digital filter. Measured magnitude/frequency and group-delay/frequency responses have been compared to the corresponding theoretical values, verifying the response characteristics of the bit-serial BP digital filter implementation.

CHAPTER 6

CONCLUSIONS

6.1 Summary of Thesis

This thesis has presented a comprehensive approach to the design and bit-serial implementation of a practical multirate LDI Jaumann BP digital filter. This BP digital filter finds applications within the existing commercial digital CODECs, where it is required to satisfy certain magnitude/frequency response characteristics for processing speech signals, while at the same time ensuring that the resulting group-delay/frequency response characteristic does not cause distortion in processing digital data.

In Chapter 2, the LDI Jaumann digital filters having Cauer configurations were discussed, and subsequently chosen for the realization of the above multirate BP digital filter because of their important practical features. In particular, they have the salient feature of exhibiting very low passband sensitivity to multiplier coefficient quantization errors. Moreover, they require the theoretical minimum number of multiply operations for the realization of LP, BP, HP, and BS transfer functions of a given order, making a corresponding area-efficient implementation feasible. A gradient based min-max optimization routine was discussed for the design of Jaumann digital filters capable of satisfying magnitude/frequency and group-delay/frequency specifications simultaneously. The required gradient calculations are based on explicit expressions which were presented for the magnitude/frequency and group-delay/frequency response characteristics of IIR digital filters with respect to the constituent multiplier coefficients, together with explicit expressions which were presented for the corresponding derivatives.

In Chapter 3, the various concepts associated with the development of bit-serial DSP architectures were reviewed. The general form of a bit-serial hardware cell was discussed together with the various forms of the required control signals. The main considerations to take into account in the development of these DSP architectures was discussed in connection with digital filter SFGs. These included the trade-offs between the required sample rate and the number of physical hardware multipliers employed, as well as the SWL and the level of hardware cell multiplexing. Finally, an approach for realizing a general-order LDI Jaumann digital filter as a bit-serial architecture was presented.

In Chapter 4, the arithmetic operations of TC bit-serial addition, subtraction, and multiplication were discussed together with their corresponding hardware cell implementations. The modified Booth multiplier cell, together with a suitable method for the return of a rounded product was presented. Moreover, the shift register cells required to facilitate the operations of parallel-to-serial conversion, serial-in/serial-out data storage, downsampling, delay, state signal storage and update operations, and serial-to-parallel conversion were given. Finally, the development of a bit-serial digital filter control signal generator concluded the chapter.

In Chapter 5, the design and bit-serial FPGA implementation of a multirate LDI Jaumann BP digital filter satisfying specifications similar to those required within the commercial digital CODECs was presented. The design employed a combination of a 5th order LP Jaumann digital filter operating at a sample frequency of 32 kHz, and a 3rd order HP Jaumann digital filter operating at a sample frequency of 8 kHz. A software package implementing the min-max procedure for the optimization of the transfer function coefficients was modified and extended to the corresponding optimization of the multirate LDI

Jaumann BP digital filter structure. A discussion of the SWL with respect to the scaling considerations was presented. The bit-serial implementation of the resulting multirate digital filter uses the Actel 1.2μ FPGA technology. A single multiplexed modified Booth multiplier was employed within each of the constituent LP and HP Jaumann digital filters to ensure an area efficient implementation while achieving the required sample rate. Measured magnitude/frequency and group-delay/frequency results were compared to the theoretical response characteristics, verifying the response characteristics of the bit-serial multirate LDI Jaumann BP digital filter implementation.

6.2 Contributions of Thesis

In Chapter 2, a new structure was proposed for HP and BS LDI Jaumann digital filters. The resulting LDI Jaumann digital filters have the same high-quality characteristics as their LP or BP counterparts and can realize highly stable transfer functions.

An approach for realizing a general-order LDI Jaumann digital filter as a bit-serial architecture was presented in Chapter 3. This approach allows practical Jaumann digital filters to be realized in a corresponding bit-serial architecture in a straightforward manner.

In Chapter 4, a comprehensive discussion of the operation of the modified Booth multiplier was given, together with the development of a representative control signal generator.

The explicit expressions derived in Chapter 2 for evaluating the magnitude/frequency and group-delay/frequency response of IIR digital filters, together with the explicit expressions derived for the derivatives of magnitude/frequency and group-delay/frequency response with respect to the constituent multiplier coefficients were used within a constrained min-max gradient based optimization method to facilitate the design of a multi-

rate LDI Jaumann BP digital filter satisfying simultaneous magnitude/frequency and group-delay/frequency specifications. This involved the modification and extension of an existing software package implementing the min-max procedure for the optimization of the transfer function coefficients to one in which the multirate filter structure was optimized. The BP digital filter was realized as a tandem connection of a 5th order LP and a 3rd order HP digital filter, where the constituent digital filters themselves were realized as LDI Jaumann digital filters. A bit-serial hardware implementation of the BP digital filter using an Actel 1.2μ FPGA was presented.

6.3 Suggestions for Further Work

The typical application of the min-max optimization satisfaction routine would be the transfer function of a general order digital filter. However, in this thesis, the min-max optimization routine was applied to the optimization of the LDI Jaumann digital filter structure rather than its transfer function. This was necessary to guarantee that the optimized multiplier coefficients would still correspond to a realizable LDI Jaumann digital filter structure. Furthermore, the optimization was applied to a specific digital filter structure, i.e. the combination of a 5th order LP and 3rd order HP LDI Jaumann digital filter operating at different sample rates. It should be possible to extend the min-max optimization to the case of general-order LDI Jaumann digital filters.

With the increased loads on transmission lines it is desirable to be able to multiplex components (i.e. the multirate digital filter) across many channels. This may not be achievable with a bit-serial based system, and the bit-parallel system may exhibit too poor an area-speed relationship to be feasible. In such a case, a digit serial [35] DSP system may be the best choice. Bit-serial systems process individual bits of data each bit-clock

period, and bit-parallel systems process individual words of data each clock period (where the bit-serial clock is typically much faster than the bit-parallel clock due to much shorter critical path latencies). Digit serial systems, on the other hand, process some multiple of bits during each clock cycle. Furthermore, depending on the unfolding factor chosen [35], digit-serial systems require approximately the same amount of shift register hardware as bit-serial systems, while requiring a slightly increased amount of hardware to realize the arithmetic cells for addition, subtraction, and multiplication. The realization of digital filters as digit-serial architectures is only beginning to be explored, and as such offers an interesting area for future research.

REFERENCES

[1] P. B. Denyer and D. Renshaw, *VLSI Signal Processing: A Bit-Serial Approach*, Addison-Wesley, 1985.

[2] R. F. Lyon, "A bit-serial VLSI architectural methodology for signal processing," in *VLSI81*, pp. 131-140, ed. J. P. Gray, Academic Press, 1981.

[3] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice-Hall, 1989.

[4] L. B. Jackson, *Digital Filters and Signal Processing*, Kluwer Academic Publishers, 1989.

[5] T. W. Parks and C. S. Burrus, *Digital Filter Design*, John Wiley & Sons, 1987.

[6] A. Antonio, *Digital Filters: Analysis and Design*, McGraw-Hill, 1979.

[7] J. Bellamy, *Digital Telephony*, John Wiley & Sons, 1982.

[8] L. T. Bruton, "Low sensitivity digital ladder filters," *IEEE Trans. Circuits and Syst.*, vol. CAS-22, pp. 168-176, Mar. 1975.

[9] B. Nowrouzian, L. T. Bruton, and N. R. Bartley, "A novel approach to the exact design of LDI Jaumann digital filters," in *Proc. 1990 International Symposium Circuits and Syst.*, New Orleans, LA, U.S.A., pp. 2144-2148, May 1990.

[10] L. M. Smith and B. Nowrouzian, "Fixed-point bit-serial implementations of LDI Jaumann digital filters," in *Proc. IEEE Pacific Rim Conf. Comm. Comp. Signal Proc.*, Victoria, B.C. vol. 1, pp. 112-115, May 1993.

[11] L. M. Smith and B. Nowrouzian, "A bit-serial multirate LDI Jaumann digital filter for digital CODEC applications," submitted to *Proc. CCVLSI 93*.

[12] V. Friedman, "Oversampled data conversion techniques," in *IEEE Circuits and Devices*, pp. 39-45, Nov. 1990.

[13] L. B. Jackson, "On the interaction of round-off noise and dynamic range in digital filters," *Bell System Tech. Journal*, vol. 49, pp. 159-184, Feb. 1970.

[14] L. E. Turner, D. A. Graham, and P. B. Denyer, "The analysis and implementation of digital filters using a special purpose CAD tool," *IEEE Trans. Education*, vol. 32, pp. 287-297, Aug. 1989.

[15] C. J. Kulach, N. R. Bartley, and L. T. Bruton, "An integrated environment for the design, simulation, and testing of multidimensional digital filters," in *Proc. IEEE Pacific Rim Conf. Comm. Comp. Signal Proc.*, Victoria, B.C. vol. 1, pp. 224-227, May 1993.

[16] A. Fettweis, "Digital filter structures related to classical filter networks," *Arch. Elektron. Uebertrag.*, vol. 25, pp. 79-89, Feb. 1971.

[17] A. Sedlmeyer and A. Fettweis, "Digital filters with true ladder configuration," *International Journal of Cir. Theory and Appl.*, vol. 1, pp. 5-10, Mar. 1973.

[18] A. Fettweis, H. Levin, and A. Sedlmeyer, "Wave digital lattice filters," *International Journal of Circuit Theory and Appl.*, vol. 2, pp. 203-211, June 1974.

[19] A. H. Gray, Jr., and J. D. Markel, "Digital lattice and ladder filter synthesis," *IEEE Trans. Audio Electroacoustics*, vol. AU-21, pp. 491-500, Dec. 1973.

[20] L. T. Bruton and D. A. Vaughan-Pope, "Synthesis of digital ladder filters from LC filters," *IEEE Trans. Circuits and Syst.*, vol. CAS-23, pp. 395-402, June 1976.

[21] B. Nowrouzian, L.T. Bruton, and D.G. Agnew, "A novel approach to the exact design of LDI digital filters," in *Proc. 35th Midwest Symposium Circuits and Syst.*, Washington, DC. U.S.A., pp. 467-470, August 1992.

[22] B. Nowrouzian, "Theory and design of LDI lattice digital and switched-capacitor filters," *IEE Proceedings-G*, vol. 139, Aug. 1992.

[23] L. E. Turner, E. S. K. Liu, and L. T. Bruton, "Digital LDI ladder filter design using the bilinear transformation," in *Proc. 1984 International Symposium Circuits and Syst.*, Montreal, P.Q., Canada, pp. 1017-1020, May 1984.

[24] B. Nowrouzian, N. R. Bartley, and L. T. Bruton, "Design and DSP-Chip implementation of a novel bilinear-LDI digital Jaumann filter," *IEEE Trans. Circuits and Syst.*, vol. CAS-37, pp. 695-706, June 1990.

[25] B. Nowrouzian and M. J. Svihura, "High speed real-time design and implementation of Cauer-type Jaumann digital filters," in *Proc. 34th Midwest Symposium Circuits and Syst.*, Monterey, CA, pp. 692-695, May 1991.

[26] K. Shimizu and T. Hirata, "Optimal design using min-max criteria for two-dimensional recursive digital filters," *IEEE Trans. Circuits and Syst.*, vol. CAS-33, pp. 491-501, May 1986.

[27] K. K. Primlani and J. L. Meador, "A non-redundant-radix-4 serial multiplier," *IEEE Journal of Solid State Circuits*, vol. 24, pp. 1729-1736, Dec. 1989.

[28] A. D. Booth, "A signed binary multiplication technique," *Quart. Journal Mech. Appl. Math.*, vol. 4, pp. 236-240, 1951.

[29] O. L. MacSorley, "High-speed arithmetic in binary computers," *Proc. IRE*, vol. 49, pp. 67-91, Jan. 1961.

[30] R. F. Lyon, "Two's complement pipeline multipliers", *IEEE Trans. Communications*, vol. COM-24, pp. 418-425, Apr. 1976.

[31] M. R. Santoro, G. Bewick, and M. A. Horowitz, "Rounding algorithms for IEEE multipliers," in *Proc. 9-th Symposium on Computer Arithmetic*, pp. 176-183, 1989.

[32] L. E. Turner and B. K. Ramesh, "Low sensitivity LDI ladder filters with elliptic magnitude response," *IEEE Trans. Circuits and Syst.*, vol. CAS-33, pp. 697-706, July 1986.

[33] "PAPILLON I: A Constrained Min-Max Optimization Routine for Mutlirate Digital Filters," Internal Report, Department of Electrical and Computer Engineering, The University of Calgary, August 1993.

[34] *The Programmable Gate Array Data Book*, Xilinx, Inc., pp. 6-33, 1989.

[35] K.K. Parhi, "A systematic approach for design of digit-serial signal processing architectures," *IEEE Trans. Circuits and Syst.*, vol. CAS-38, pp. 358-375, Apr. 1991.