HIGH-SPEED FIR FILTER DESIGN AND OPTIMIZATION USING ARTIFICIAL INTELLIGENCE TECHNIQUES

CEN LING

(B. Eng., USTC; M. Eng., IMPCAS)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2005

Acknowledgements

I would like to express my most sincere appreciation to my supervisor Dr. Lian Yong for his guidance, patience and encouragement throughout the period of my research. His stimulating advice benefits me in overcoming obstacle on my research path.

My earnest gratitude also goes to Professor Yong Ching Lim and Dr Sadasivan Puthusserypady for their valuable advices.

Most importantly, I wish to thank my parents, Cen Shi and Fan Huilan, for their love and support and for all things they have done for me in my life and thank my bothers, Cen Lei and Cen Weidong, my sister Cen Wei for their care, encouragement and love. I also would like to thank my husband, Liu Ming for his encouragement and accompanying during the period. Special thanks to my son Liu Cenru for the happiness he has given to me.

I am also grateful to all friends in Signal Processing and VLSI Design Lab in the Department of Electrical and Computer Engineering for making my years in NUS a happy time. They are Mr. Francis Boey, Dr. Yu Yajun, Mr. Liu Xiaoyun, Dr. Yang Chunzhu, Mr. Yu Jianghong, Ms. Cui Jiqing, Mr. Luo Zhenyin, Mr. Zhou, Xiangdong, Mr. Liang Yunfeng, Ms. Zheng Huanqun, Ms. Sun Pinping, Mr. Wang Xiaofeng, Mr.

Lee Jun Wei,, Mr. Yu Rui, Ms. Hu Yingping, Mr. Cao Rui, Mr. Gu Jun, Mr. Wu Honglei, Mr. Tong Yan, Mr. Chen Jianzhong and Mr. Pu Yu.

Finally, I wish to acknowledge National University of Singapore (NUS) for the financial support provided throughout my research work.

Contents

Summary

Finite impulse response (FIR) digital filters are preferred in most of the wireless communication systems and biomedical applications due to its linear phase properties. A major drawback of the FIR filters is the large number of arithmetic operations needed for its implementation, which limits the speed of the filter and requires high power. It is well known that the coefficients of an FIR filter can be quantized into sum or difference of signed powers-of-two (SPoT) values leading to a multiplication-free implementation. In the application-specific integrated circuit (ASIC) implementation, a long FIR filter can operate at high speed without pipelining if it is factorized into several short filters whose coefficients are in the form of SPoT terms. Such implementation reduces the hardware cost and lowers the power consumption significantly as it converts multiplication to a small number of shift and add operations. However, the design of FIR filter with SPoT coefficient values is a complex process requiring excessive computer resources, especially in situations where several filters have to be jointly designed.

In this thesis, several optimization schemes based on the artificial intelligence techniques are presented for the design of high-speed FIR filters with SPoT coefficient values. Firstly, genetic algorithm (GA) based optimization methods are proposed for the design of low power high-speed FIR digital filters. The high-speed and low power features are achieved by factorizing a long filter into several cascaded subfilters each with SPoT

coefficients. Significant savings on hardware cost are achieved due to the fact that the information which is related to hardware requirement is affiliated to the fitness function as an optimization criterion. An adaptive genetic algorithm (AGA) with varying population size and probabilities of genetic operations is proposed to improve the optimization performance of conventional GA. Secondly, two hybrid algorithms are presented for the synthesis of very sharp linear phase FIR digital filters with SPoT coefficients based on frequency response masking technique (FRM). They are generated by combining the GA with an oscillation search (OS) algorithm and with the simulated annealing (SA) algorithm, respectively. The OS and SA algorithms are used to improve the convergence speed of the GA and prevent premature convergence. Thirdly, an efficient algorithm is proposed for the design of general FIR filters with SPoT coefficient values, where AGA, SA and tabu search (TS) techniques cooperate during the optimization process. The proposed algorithm achieves not only the improvement of solution quality but also the considerable reduction on computational efforts. Fourthly, a modified micro-genetic algorithm (MGA) is applied to overcome the drawbacks of the conventional GA of long computation time by utilizing a small population. To avoid entrapment in local optimum, the MGA is modified to adjust the probabilities of crossover and mutation during the evolutionary process. The proposed method can design digital FIR filters with SPoT coefficient values in much higher speed than conventional GA.

List of Figures

List of Tables

Glossary of Abbreviations

AGA: Adaptive genetic algorithm

AGSTA: A hybrid algorithm formed by integrating adaptive genetic algorithm, simulated annealing and tabu search techniques

AI: Artificial intelligence

ASIC: Application-specific integrated circuit

CSA: Carry save adders

CSD: canonic signed-digit

DSP: Digital signal processing

FIR: Finite impulse response

FRM: Frequency-response masking

FPGA: Field Programmable Gate Array

GA: Genetic algorithm

GSA: A hybrid algorithm formed by integrating genetic algorithm and simulated annealing algorithm

IIR: Infinite impulse response

ILP: Integer linear programming

MILP: Mixed-integer linear programming

MGA: Micro-genetic algorithm

NPR: Normalized peak ripples

OS: Oscillation search

OSGA: Oscillation search genetic algorithm

PRP: Proportional relation-preserve method

SA: Simulated annealing

SPoT: Signed powers-of-two

SSS: Simple symmetric-sharpening method

TS: Tabu search

VLSI: Very Large Scale Integration

List of Symbols

a_1, a_2 and a_3: Positive weighting coefficients in fitness function

b: Search index in OS algorithm

B: Coefficient word length

B_j: Coefficient word length of j^{th} subfilter

df_T: Temperature decreasing factor in SA algorithm

Diff: Difference between the absolute value of the maximal and minimal coefficients

E: Energy

ΔE: Change of energy

f: Fitness of a chromosome

f_T: Sum of the fitness values in the population

f_{avgo}, f_{avgn}: Mean of the best fitness values during a specific number of generations

f_{besti}: Best fitness in i^{th} generation.

f_{max}: Best fitness of current population

f_{avg}: Average fitness of current population

f': Larger fitness value of the two chromosomes to be crossed

G: Number of generations

G_S: Start point for calculating the adjustment of population size in AGA

G_{OS}: Number of generations in the application condition of OS

G_{sA}: Number of generations in the application condition of SA

g_{qn}: Controllable quantization noise gain

$h(i)$: i^{th} coefficient of an FIR filter

$h_{org}(i)$: Initial value of i^{th} coefficient

h_i: Coefficients of i^{th} subfilter

$H(z)$: z-transform transfer function of a FIR filter

$H(e^{j\omega})$: Frequency response of $H(z)$

$H_i(z)$: z-transform transfer function of the i^{th} subfilter

$H_d(e^{j\omega})$: Frequency responses of the desired filter

$H_{ip}(z)$: z-transform transfer function from the input of the i^{th} subfilter to the output of the p^{th} subfilter

$H_{in}(f)$: The frequency responses of the prototype filter

$H_{out}(f)$: The frequency responses of the transformed filter

$H_a(z)$: Bandedge shaping filter in FRM structure

$H_{ai}(z)$: i^{th} cascaded subfilter of $H_a(z)$

$H_c(z)$: Complementary filter of $H_a(z)$

$H_{Ma}(z)$, $H_{Mc}(z)$: A pair of masking filters in FRM structure

k_1, k_2, k_3 and k_4: Positive weighting coefficients to control the adjustment of mutation probabilities in AGA

$k(\omega)$: Positive weight in each band

L_{end}: Convergence condition of SA

L_{it}: Number of reordering iterations in pre-SA process

M: Interpolation factor

N: Overall filter length

N_i: Length of i^{th} subfilter

N_a: Length of $H_a(z)$

N_{ai}: Length of $H_{ai}(z)$

N_T: Total length of all subfilters

N_{sub}: Sum of the lengths of partial subfilters

O: Objective function

O_{OS}: Objective function in OS

P: Population size

P_{org}: Original population size

ΔP : Adjustment of population size

p: Number of cascaded subfilters

p_c: Crossover probability

p_m: Mutation probability

p_a: Adaptive acceptance probability in TS

p_{os}: Application probability of OS

Q, R: Integers denoting the power of two

$s(i)$: Ternary digit from {-1,0,1}

S: SPoT number represented to a precision 2^{Q} by L - Q ternary digits $s(i)$

S_T: Total number of the SPoT terms

S_S: Pre-specified maximal number of SPoT terms

T: Temperature

T_{end}: Ending temperature

$Trig(n,\omega)$: Trigonometric function

TM_j: Truncation margin of j^{th} cascaded subfilter

$X(e^{j\omega})$: Frequency response of input signal

$Y(e^{j\omega})$: Frequency response of output signal

Z_T: Total number of zero-coefficients

ω_p: Passband edge

ω_s: Stopband edge

θ: Passband edge of $H_a(z)$

ϕ: Stopband edge of $H_a(z)$

Chapter 1

Introduction

Digital signal processing (DSP) techniques have been increasingly applied in most engineering and science fields due to the explosive development in digital computer technology and software development. Digital filters are basic building blocks for DSP systems. There are two types of filters: finite impulse response (FIR) filters and infinite impulse response (IIR) filters. Since FIR filters possess many desirable features such as exact linear phase property, guaranteed stability, free of limit cycle oscillations, and low coefficient sensitivity [64-66], they are preferred in most of the wireless communication systems and biomedical applications. However, the order of an FIR filter is generally higher than that of a corresponding IIR filter meeting the same magnitude response specifications. Thus, FIR filters require considerably more arithmetic operations and hardware components - delay, adder and multiplier. This makes the implementation of FIR filters, especially in applications demanding narrow transition bands, very costly. When implemented in VLSI (Very Large Scale Integration) technology, the coefficient multiplier is the most complex and the slowest component. The large number of arithmetic operations in the implementation also increases the power consumption. In the modern applications, such as military devices, wearable devices and portable mobile

communication devices, the portability and low power dissipation play a very important role.

To address the problem, considerable attention and efforts have been made on reducing the complexities and power consumptions for the DSP systems. The cost of implementation of an FIR filter can be reduced by decreasing the complexity of the coefficients [1, 4-5, and 67]. Coefficient complexity reduction includes reducing the coefficient word length and representing coefficients in effective form. One of the most efficient ways is to design filters with coefficients restricted to the sum or difference of signed powers-of-two values [1]. This leads to a so-called multiplication-free implementation, i.e. the filter's coefficient multipliers can be replaced by simple shift-and-add circuits. Thus, the implementation complexity can be reduced, resulting in significant increase in the speed and reduction in power dissipation.

1.1 Signed Powers-of-Two Based Filter Design

To design FIR digital filters over the signed powers-of-two (SPoT) discrete space was firstly proposed by Lim and Constantinides [68]. Extensive research has shown that the complexity of an FIR digital filter can be reduced by quantizing its coefficients into SPoT values. This converts multiplication to simple operations of shift and add. Relatively small chip area is required in VLSI realization, resulting in low cost, high speed, and high yield. This section briefly describes the SPoT number characteristics and existing optimization techniques for the design of digital filters subject to SPoT coefficients.

A number, S, is called an SPoT number in this thesis, if it is represented to a precision 2^Q by R - Q ternary digits $s(i)$ according to

$$S = \sum_{i=Q}^{R-1} s(i)2^i,\ s(i) \in \{-1, 0, 1\},\ Q \le i \le R-1, \tag{1.1}$$

where R and Q are integers. Each nonzero digit term, $s(i) \neq 0$, is counted as a SPoT term. The word length of S is (R-Q) bits. S is discrete values in increments of 2^Q in the range

$$-2^R + 2^Q \le S \le 2^R - 2^Q, \tag{1.2}$$

in which there are $2^{R-Q+1} - 1$ distinct values.

Preliminary studies have showed that only a limited number of SPoT terms are required to meet a respectable set of specifications if a good optimization technique exists. Hence, to represent the coefficients of a filter in this way, the coefficient multipliers can be replaced by a small number of add/subtract-shift operations. The hardware complexity as well as power consumption is thus largely reduced.

During the last three decades, there has been significant research interest in the design of digital filters with discrete coefficients [1, 3-14, 69-74]. In [3] Munson has proposed a method to obtain discrete coefficients by simply rounding the real valued coefficients of the desired filter, which provides an optimal solution in the time domain error norm or in the output minimal mean-square error norm sense. Considering the optimal design solution in the frequency domain, Kodek [4] has introduced integer linear programming (ILP) to solve the filter design problem. However, relatively long coefficient word length and exponentially increased computer time with respect to the filter length make ILP only suitable for the design of low-order FIR filters. To improve the performance of ILP for designing high-order FIR filters with discrete coefficients, a mixed-integer linear

programming (MILP) technique has been proposed by Kodek. Although optimal design in the minimax sense can be found using MILP, the application of this technique is limited by high computing cost. Considerable efforts for the improvement of MILP have been made to reduce its computational complexity for high-order FIR filter design in [4]. In [1], Lim and Parker proposed an improved MILP method for the design of FIR filters with SPoT coefficient values. It is reported that their method can be efficiently used in the design of filters with lengths up to 70 [5]. However, this is not long enough for some designs, e.g. a filter with very sharp transition band. It is shown in [69] that MILP can minimize the total number of SPoT terms if the problem is appropriately formulated, thus leading to a filter with minimal implementation cost. However, MILP requires excessive computing resources if the filter length is long. The computational cost required increases exponentially with the number of variables to be optimized.

In [6], Zhao and Tadokoro proposed a suboptimal design for powers-of-two coefficient based FIR filters. This algorithm is composed of two methods. The first is a suboptimal design which preserves a proportional relation between the conventional FIR filters and the powers-of-two coefficient based FIR filters, referred to as the proportional relation-preserve method (PRP). The second is the application of the simple symmetric-sharpening method (SSS) which is applied when the PRP method cannot realize the given filter specifications. It is shown in [6] that with the help of the PRP and SSS methods, FIR filters with lengths greater than 200 can be efficiently designed with powers-of-two coefficient values, which addresses the very high computational cost of the MILP.

An improved algorithm for the optimization of FIR filter with SPoT coefficient value is proposed by Samueli in [7], which allocates an additional nonzero digit in the canonic signed-digit (CSD) code to the larger coefficients to compensate for the non-uniform nature of CSD coefficient distribution. The two-stage algorithm consists of search for an optimum scale factor and a bivariate local search in the neighborhood of the scaled and rounded CSD coefficients. It is illustrated that a significant improvement in the frequency response can be obtained at the price of minimal increase in filter complexity resulting from the additional CSD digits.

Tree search with weighted least-squares criteria [70-71] is proposed in the design of certain types of filters, which replaces the linear programming algorithm in tree search method by a suitable weighted least-squares algorithm. In such algorithms, the filter's coefficient values are quantized one at a time. The remaining un-quantized coefficients are optimized in the weighted least-squares sense. The computing time required is approximately proportional to the cube of the number of filter coefficients to be optimized but the optimal solution is not guaranteed.

In the quantization guided by coefficient sensitivity analysis technique [72-73], each coefficient is first set to its nearest single-SPoT-term number. The second-SPoT-term is then allocated to the filters' coefficients one at a time in decreasing order of the coefficient sensitivity, until the frequency response meets the given specification. Coefficient sensitivity is defined as the sum of the increase in the peak passband ripple value and the increase in the peak stopband ripple value when the coefficient is set to its

nearest single-SPoT-term number. A modified sensitivity criterion considers the average ripple magnitude changes over all the frequency grid points in the passband and stopband. In [35], a SPoT term allocation scheme is proposed, where each coefficient of the filter is allocated a certain number of SPoT terms according to the coefficient's statistical quantization step-size and sensitivity subject to a given total number of SPoT terms. After the assignment of the SPoT terms, MILP is used to optimize the coefficient values. In [8], Li et al. proposed a polynomial-time algorithm for designing FIR filter with SPoT coefficient values. In [8], SPoT terms are dynamically allocated to the currently most deserving coefficient, one at a time, to minimize the L^{∞} distance between the SPoT coefficients and their corresponding infinite word length values. Since the complexity of the algorithm is polynomial-time, the computation time to design a FIR filter is rather short. The computational complexity increases linearly with the increase of the word length of filter coefficients. Hence, the algorithm is suitable to design filters with high order and long word length of coefficients. In [74], each coefficient is firstly assigned SPoT terms using the technique of [8]. Subsequently, a pool of SPoT terms is created for each coefficient according to the coefficient's infinite precision value. A dynamic programming technique is used to allocate SPoT terms taken from the coefficient's pool of SPoT terms to each coefficient.

Artificial intelligence (AI) techniques have been widely used to solve optimization problems which are not easy to handle by conventional optimization approaches. Recently, techniques such as genetic algorithm (GA), and simulated annealing (SA) received increasing attention in science and engineering fields. Many techniques based

on AI [9-14] have been proposed for the design of FIR filters with discrete coefficients. In [9-10], the SA based method is proposed. However, due to the numerous local optimal, many runs of the design for the same filter were needed to find a satisfying solution if the filter length was longer than 39. The GA proposed by J.H. Holland [16] is an artificial system based on the principle of natural selection where stronger individuals are the likely winners in a competitive environment. As a stochastic algorithm, the GA is a robust and powerful optimization method for solving problems with a large search space which are not easily solved by exhaustive methods. Many publications [11-14] have reported that the GA is effective for the design and optimization of FIR digital filters with SPoT coefficients due to its properties such as multi-objective, coded variables and natural selection. In [11], Lee and Ahmadi demonstrated the design of 1-D FIR filters using the GA. They examined the usefulness of various error norms and coding schemes applied to the filter coefficients and their impact on convergence rate and optimal results. To improve the optimization performance and increase the calculation efficiency, some modifications have been made on the conventional GA, such as improved genetic operators [12-13], efficient coding schemes [14-15] and new natural selection process [16]. In these publications, they all presented useful development on the GA and demonstrated that their algorithms can outperform the conventional GA in FIR filter design.

However, the optimal design of FIR filters in the SPoT space is very complicated. It is shown in [10] that there are many local optima existing in the design of filters with length greater than 39. Although these AI based algorithms are global optimization techniques

in theory, they risk finding a suboptimal solution and has low convergence speed in complex applications.

1.2 Research Objectives and Major Contribution of this Thesis

The major drawback for the SPoT based FIR filters is the complexity associated with the quantization of each coefficient into SPoT space. The process requires huge amount of computer resources and takes very long time to find the optimal SPoT coefficients, especially for high-order filters. Searching the optimal coefficients in a discrete space can be formulated as a nonlinear optimization problem. If the desired objective of minimization is the normalized peak ripple magnitude, the quotient of the peak ripple magnitude and passband gain should be used as the objective function to be optimized [5]. Since this quotient is nonlinear, the objective function is a nonlinear function. This makes linear programming or simple iterative methods usually lead to sub-optimal designs, except exhausted search. Unacceptable computational cost makes it impractical to utilize exhausted search even for middle-length filter design. Although the GA and SA have potential to find global optima solutions, they risk finding suboptimal solutions due to the following reasons.

1) Global optimal solutions can be possibly achieved when all parameters are jointly optimized. This further puts a high demand on optimization methods with the increase of problem dimension and solution space. The large search space associated with high problem dimension leads to numerous local optima, where a

general purpose optimization algorithm may be unable to jump out of a local optimum without external help.

2) The GA and SA belong to a class of stochastic optimization techniques. They find solutions without incorporating any rules of the problem to be optimized. This is a great advantage and also a biggest disadvantage of these algorithms. Therefore, they do not always evolve towards a good solution; they only evolve away from bad circumstances. That is why they risk finding a suboptimal solution and have low convergence speed in complex applications. Each optimization problem has its own properties. It is possible to boost the optimization performance if these properties are incorporated into optimization process.

3) Many control parameters, e.g. population pool size and the probabilities of genetic operations in the GA, should be set up before applying these algorithms. The suitable settings of these parameters are critical to the performance of optimization. How to set these parameters for a special optimization problem is still an open issue. Also, parameter settings optimal in the earlier stages of the search typically become inefficient during the later stages.

Furthermore, the performance of a filter with discrete valued coefficients is also limited by the number of bits used in the quantization process. It was shown in [2] that the peak ripple of the amplitude response decreases with increasing filter length up to a certain length when an FIR filter with discrete valued coefficients is implemented in the direct form. Significant reduction in peak ripple beyond that limit cannot be attained easily without increasing the coefficient precision. In order to reduce the peak ripple while

keeping the precision of the coefficient values unchanged, other forms of realization must be considered. It has been reported in [2] that the reduction on peak ripple can be possibly achieved by implementing the filter in cascade form. Factorizing a long FIR filter into several short direct form filters can also shorten the critical path, which leads to higher throughput. The factorization coupled with SPoT based coefficients yields a cost effective high-speed FIR filter. However, there lacks of systematic methods to deal with the design complexity rising from the joint optimization of several subfilters each with SPoT coefficients.

The frequency-response masking (FRM) technique [36-48] is one of the most computationally efficient ways for the synthesis of arbitrary bandwidth sharp linear phase FIR digital filters. A great benefit of the FRM approach is significant reduction in the number of multiplications which can be as high as 98% as reported in [37]. Combining FRM and SPoT techniques, it is possible to implement a high speed FIR filter using either Field Programmable Gate Array (FPGA) devices or application-specific integrated circuit (ASIC) as in [56]. In one stage FRM structure, there are three subfilters, i.e. the bandedge shaping filter and a pair of masking filters[1]. The optimization process is extremely complicated if at least 3 subfilters have to be jointly designed.

In this thesis, the research objectives are to develop efficient approaches based on AI techniques to address the above problems in the design of FIR filters with SPoT coefficients. Several tailor-made optimization algorithms are developed to improve the design performance, which suit the need of different design requirements.

[1] The bandedge shaping filter in FRM structure is used to synthesize the sharp transition band, which can be much longer than two masking filters in a single stage FRM design; and a pair of masking filters together with the complement of interpolated bandedge shaping filter construct the arbitrary bandwidth of the overall filter.

The following is claimed to be the contributions of the thesis.

1. An adaptive genetic algorithm (AGA) with adjustable population size and genetic operation probabilities is proposed to improve the convergence performance of the traditional GA. A systematic optimization method based on the AGA is proposed for the design of cascade form FIR filters. By factorizing a long filter into several short filters each with SPoT coefficients, a cost effective high-speed FIR filter can be yielded. To reduce the word lengths of the output signals, several of the least significant bits are truncated from the output of cascaded subfilters. The instructions for the design of cascade form filters with truncation are derived from simulation. An empirical equation to estimate optimal truncation margin is proposed.
2. Two novel hybrid algorithms are proposed for the design of FRM FIR filters. The oscillation search genetic algorithm (OSGA) is generated by integrating the GA with an oscillation search (OS) algorithm that is proposed according to the properties of filter coefficients, which can be efficiently utilized to jointly design the bandedge shaping filter and masking filters in FRM structure. By combining the GA and SA, another hybrid algorithm (GSA) is proposed for the design of high-speed FRM FIR filters, where the long bandedge shaping filter is replaced by several cascaded short filters.
3. Compared with FRM filters that have very sharp transition bands, general FIR filters with relative broad transition bands have simple objective function. Without the need of conducting complicated computation of objective function,

the research for general FIR filter design is focused on the solution quality and algorithm stability. To this end, a novel hybrid genetic algorithm (AGSTA) is developed for the optimal design of FIR filters.

4. To find acceptable solutions with the least computational cost, a modified micro-genetic algorithm (MGA) is proposed to reduce the computational cost by utilizing a very small population. To avoid entrapment in local optima, the proposed MGA includes a strategy that adjusts the probabilities of crossover and mutation during the evolutionary process. The modified MGA can be efficiently applied in the design of discrete valued filters.

1.3 Organization of the Thesis

This thesis is organized as follows.

1. Chapter one gives an introduction to the problems considered in this thesis. The signed powers-of-two coefficient property and the existing SPoT coefficient design techniques are also reviewed. The research objective and major contributions made in this thesis are presented at the end of this chapter.
2. In Chapter two, GA based approaches are presented for the design of low power high-speed FIR digital filters. The high-speed and low power features are achieved by factorizing a long filter into several cascaded subfilters each with SPoT coefficient values. With the help of genetic encoding scheme, the coefficients of all subfilters are quantized into SPoT values simultaneously. The

proposed methods reduce the hardware cost significantly as the information which is related to hardware requirement is affiliated to the fitness function as an optimization criterion. To further reduce the hardware cost, the signal is truncated at the output of subfilters. Some useful guidelines for truncation are presented. An empirical equation to estimate the optimal truncation margin is derived. To improve the convergence performance of the conventional GA, an AGA is proposed in this chapter, which adaptively adjusts the population size and the probabilities of genetic operations during optimization process. It is shown by means of examples that significant savings in terms of hardware cost are achieved by using the proposed methods.

3. In Chapters three and four, two hybrid algorithms are proposed for the design and optimization of very sharp linear phase FIR digital filters with discrete valued coefficients based on the FRM technique. The first algorithm, OSGA, integrates the OS algorithm into the optimization process of the GA. The OS algorithm is developed according to the properties of filter coefficients, which is used to reduce the computational cost required by the conventional GA. Furthermore, it can also help prevent GA from premature convergence by escaping from local optima. The second algorithm, GSA, combines the GA with the SA for the design of FRM filters, where the FRM filter structure has been modified to improve the throughput by replacing the long bandedge shaping filter with several cascaded short filters. The coefficients of all subfilters are designed with SPoT values resulting significant reduction in both hardware cost and power consumption. The

hardware cost is reduced due to the fact that the coefficient word length is included as one of the terms in the fitness function during the optimization. The performance of these two algorithms is illustrated through design examples.

4. For general FIR filter design with broad transition bands, a hybrid algorithm, AGSTA, is proposed in Chapter five. Generating the AGA and the features of SA and TS technology leads to a hybrid genetic scheme, where the AGA is used as the basis of the hybrid algorithm. The SA algorithm is applied to optimize a certain number of chromosomes with better fitness values when the further improvement of fitness cannot be achieved for a pre-specified number of generations in optimization process. The use of the SA algorithm is to help escape from the local optima and to prevent premature convergence. In the AGSTA, the concept of tabu is used to improve the convergence speed by reducing search space according to the properties of filter coefficients. Compared with other algorithms, the proposed AGSTA achieves not only an improved solution quality but also the considerable reduction of computational effort.
5. Although designing FIR filters is more than a job that can be done off-line, the computational complexity does become an issue. To find acceptable solutions with the least computational cost, a modified micro-GA is presented in Chapter six. The MGA overcomes the drawbacks of the conventional GA of long computation time by utilizing a small population. To avoid trapping into local optima, the proposed MGA is modified to adjust the probabilities of crossover and mutation during the evolution. It is suitable in the design of FIR filters in both

direct form and cascade form. With the help of the MGA, considerable savings on computational cost can be achieved in comparison with the conventional GA.

6. The seventh chapter consists of the conclusions of the thesis and some recommendations for future research.

1.4 List of Publications

Part of the research work reported in this thesis is published or accepted for publication as follows.

[1] Yong Lian and Ling Cen, "A genetic algorithm for the design of low power high-speed FIR filters," *Proc. of the IEEE Seventh International Symposium on Signal Processing and its Applications*, vol. 1, pp. 181 -184, Paris, France**,** Jul. 1-4, 2003.

[2] Ling Cen and Yong Lian, "High speed frequency response masking filter design using genetic algorithm," *Proc. of the IEEE International Conference on Neural Networks & Signal Processing*, vol. 1, 14-17, pp. 735 – 739, Nanjing, China, Dec. 14-17, 2003.

[3] Ling Cen and Yong Lian, "Low-Power implementation of frequency response masking based FIR filters," *Proc. of the IEEE Fourth International Conference on Information, Communications & Signal Processing and Fourth Pacific-Rim Conference on Multimedia*, vol. 3 , 15-18, pp. 1898 – 1902, Singapore, Dec. 15-18, 2003.

[4] Ling Cen and Yong Lian, "Complexity reduction of high-speed FIR filters using micro-genetic algorithm," *Proc. of the IEEE First International Symposium on Control*,

Communications and Signal Processing, pp. 419 – 422, Hammamet, Tunisia, March 21-24, 2004.

[5] Ling Cen and Yong Lian, "A modified micro-genetic algorithm for the design of multiplierless digital FIR filters," *Proc. of the IEEE International Conference on Digital Techniques in Electrical Engineering*, pp. 52-55, Chiang Mai,. Thailand, Nov. 21-24, 2004.

[6] Ling Cen and Yong Lian, "A hybrid GA for the design of multiplication-free frequency response masking filters," *Proc. of the IEEE International Symposium on Circuits and Systems*, pp. 520-523, Japan, May 23-26, 2005.

[7] Ling Cen and Yong Lian, "Hybrid Genetic algorithm for the design of modified frequency- response masking filters in a discrete space," *Circuit Syst. Signal Process.*, vol. 25, no. 2, pp. 153-174, Apr. 2006.

[8] Ling Cen, "A Hybrid Genetic Algorithm for the Design of FIR filters with SPoT Coefficients," *Journal of Signal processing (Elsevier)*, vol. 87, no. 3, pp. 528-540, March 2007.

Chapter 2

Design of Cascade Form FIR Filters

2.1 Introduction

To design an FIR digital filter over the signed powers-of-two (SPoT) discrete space leads to a so-called multiplication-free implementation, i.e., the filter's coefficient multipliers can be replaced by simple shift-and-add circuits. Thus, the computational complexity of the filter is reduced. Significant increase in the speed and reduction in power consumption can be achieved. Many methods have been developed for optimizing the frequency response of a digital filter subject to SPoT constrains imposed on its coefficient values, which have been introduced in Chapter 1.

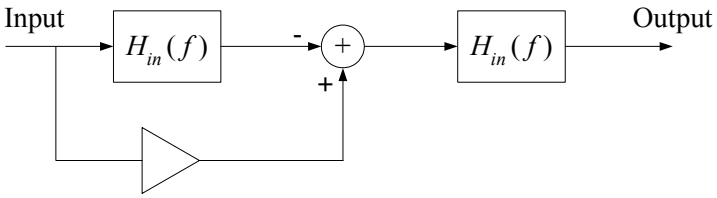
In SPoT based filter design, global optimal solutions can possibly be achieved if all parameters are jointly optimized. This puts a high demand on optimization methods. When existing AI based optimization techniques such as the GA and SA are utilized, the high problem dimension and large search space may cause high probability of premature convergence and low convergence speed. Suboptimal solutions are usually achieved due to the tradeoff between the searching speed and solution quality. It has been shown in [10] that it starts providing many local optimal solutions if the number of filter coefficients is higher than 39. If the SA was applied in such design, many more runs were needed to

find a good solution [10], which further increased computational cost. Furthermore, there are several control parameters in AI based algorithms, which critically control the optimization performance. The setting of these parameters itself is a complex optimization problem. It is most important to develop suitable ways to find optimal values for these parameters.

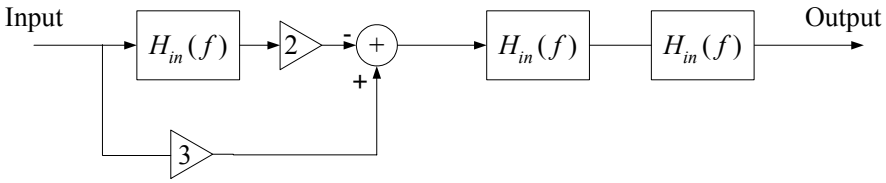
The quantization performance is limited by the coefficient precision. In a direct form filter, after the filter length increases to a certain bound, it is difficult to attain significant reduction in peak ripple of the amplitude response without increasing the coefficient word length [2]. However, increasing the filter length and coefficient word length will increase the hardware cost in filter implementation and reduce the operation speed of the filter. It is reported in [2] that by implementing the filter in cascade form, smaller peak ripple can be possibly achieved without increasing the coefficient precision. Furthermore, the throughput of a long filter can be improved by factorizing it into several short direct form filters. Quantizing the coefficients of an FIR filter into SPoT terms and factorizing a long filter into several short filters will lead to a VLSI implementation with less power dissipation, lower cost, and higher speed.

When two identical filters are cascaded together, the peak passband ripple magnitude of the cascaded filter will be twice as large as that of the individual filters. Twicing [75] and sharpening [76] are two methods to improve the performance of a filter by both increasing stopband rejection and decreasing passband error. Fig. 2.1 shows the block diagrams of twicing and sharpening schemes, where $H_{in}(f)$ denotes the frequency responses of the prototype filter. The transformed filter, $H_{out}(f)$, is given by

$H_{out}(f) = H_{in}(f)[2 - H_{in}(f)]$ in twicing scheme and $H_{out}(f) = H_{in}^2(f)[3 - 2H_{in}(f)]$ in sharpening scheme. The major idea of the twicing and sharpening schemes is to do a better job of filtering by suitably combining the results of several passes through the same filter.



(a) Twicing



(b) Sharpening

Fig. 2. 1 Block diagrams of twicing and sharpening schemes.

It is demonstrated in [17] that, if the filters to be cascaded are not identical and they can be jointly designed, the peak passband ripple magnitude of the cascaded filter might be smaller than those of the individual filters. Thus, the saving in the filter length can be produced in comparison with the filter constructed using identical filters.

However, the optimal design of cascaded discrete coefficient filters is a nonlinear process requiring excessive computer resources, even for small designs [1]. An iterative optimization approach has been introduced in [2] where each subfilter is optimized separately by using the linear programming techniques. In this method, one of the subfilters is fixed and the other subfilter is designed as an equalizer to compensate for the ripples of the fixed one. The roles of the fixed filter and the equalizer are interchanged

and the respective filter redesigned. The process is continued until convergence. The method works very well for breaking a long filter into two, but it becomes very complicated if the number of subfilters goes beyond two. There is no guarantee that the filter synthesized by the technique reaches the global optimum. In [18], the author presented a method based on the SA technique to factorize a long filter into several cascaded subfilters of 2- or 4-order in continuous space. However, there are two limitations in their methods. First, they cannot design a filter with any desired number of cascaded subfilters; second, they cannot be directly utilized to design filters with discrete valued coefficients.

In this chapter, we will address the design of cascade form FIR filters with SPoT coefficients based on the evolutionary algorithms. In our approaches, all subfilters are jointly designed to meet the given specifications, where the peak ripple in each band is simultaneously minimized by global optimization techniques. To overcome the drawbacks of the conventional GA, an adaptive genetic algorithm (AGA) is proposed, which adaptively adjusts the control parameters during the evolutionary process. Several of the least significant bits are truncated from the outputs of intermediate stages to decrease the word length of output signal.

This chapter is organized as follows. A high-speed filter structure is presented in Section 2.2. Following that, a method to factorize a long FIR Filter into cascaded short filters with optimal quantization noise is discussed. The GAs are introduced in Section 2.4 as the major optimization technique utilized in this chapter, as well as in this thesis. The implementation details for the design of high-speed low power FIR filters using the GA

is discussed in the following section. The AGA with varying population size and probabilities of crossover and mutation is proposed in Section 2.6. A systematic method based on the AGA is proposed in Section 2.7, where the method presented in Section 2.5 is further improved. Finally, the conclusions are exposed.

2.2 A High-Speed Filter Structure

Most of the FIR filters can be implemented using either Direct Form I or II structures. The Direct Form II is preferred for the high-speed filter implementation due to its relatively short critical path from the input to the output. The simple carry save adders (CSA) can be used in the accumulation path to eliminate the propagation of carry signals in a full adder that speed up the arithmetic operations. There are two drawbacks for such a structure. First, the input is broadcasted to each filter tap simultaneously that requires very large fan-out from the input source. In most implementations, buffers are inserted in the input path to reduce the loading of input signal and to increase the speed. Second, the word length of each delay element is increased significantly as it is the sum of the word length of coefficient and input signal. As a result, large amounts of D flip-flops and full adder cells are needed.

In a Direct Form I implementation, the critical path depends on the length of the accumulation path that adds all delay-weighted input samples. For an odd length filter of length N, the number of adders in the accumulation path is equal to $(N+1)/2$ taking into account of the symmetric property of a linear phase FIR filter. A traditional way to

improve the speed for such a structure is to employ pipeline or parallel process techniques at the cost of increasing hardware complexity. It is obvious that the critical path in a Direct Form I can be shortened if the filter length is reduced. This prompts to consider the factorization of a long filter into several short filters as shown in Fig. 2.2. The shorter each subfilter is, the higher the throughput is. Such a cascaded structure is well suited for both Direct Form I and II structures. The improvement of the speed depends on the number of factorized subfilters. Furthermore, to design a filter in a cascaded structure, it is possible to reduce the peak ripple without increasing the coefficient precision. The factorization coupled with SPoT based coefficients yields a cost effective high-speed low power FIR filter. The design of such a filter has posed a challenge to the filter designers as the global optimization can be achieved only if all subfilters are optimized simultaneously in a discrete space.

A digital FIR filter is characterized by the following z-transform transfer function

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

where $h(n)$ represents the filter coefficients. The zero phase frequency response of the filter is given by

$$H(e^{j\omega}) = \sum_{n=0}^{N-1} h(n)\,\mathrm{Trig}(n,\omega)\ , \tag{2.2}$$

where $Trig(n,\omega)$ is a trigonometric function depended on the type of filter used.

If the filter is constructed by p cascaded subfilters shown in Fig. 2.2, its z-transform transfer function is given by

$$H(z) = \prod_{i=1}^{p} H_i(z), \tag{2.3}$$

where $H_i(z)$ is the z-transform transfer function of the i^{th} subfilter.

Let $h_i(n)$ be the coefficients of i^{th} subfilter and N_i be its length., the frequency response of the overall filter, $H(z)$, can be expressed as

$$H(e^{j\omega}) = \prod_{i=1}^{p} \left(\sum_{n=1}^{N_i - 1} h_i(n) Trig_i(n, \omega) \right), \tag{2.4}$$

where $Trig_i(n, \omega)$ is trigonometric function.

The design of cascade form discrete valued coefficient filter is a discrete optimization problem of finding a set of discrete valued $\{h_i(n)\}$, so that $H(e^{j\omega})$ is a best approximation to a desired function with respect to a given criterion; the criterion used here is the minimax one. Equation (2.4) is nonlinear in the coefficients and is difficult to optimize in the discrete space [1]. This render the optimal design of cascaded discrete coefficient filter a nonlinear process. To address the problem, the methods based on AI techniques will be presented in the later sections.

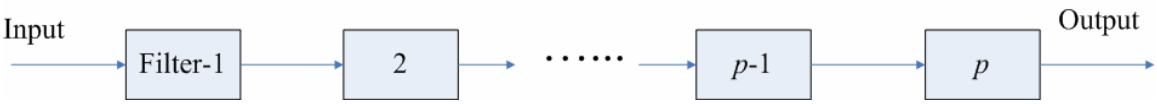


Fig. 2. 2 A cascade form filter consisting of p subfilters.

2.3 Quantization Noise Reduction for Cascaded FIR Filters Using Simulated Annealing

In practical applications, finite word length effects in terms of quantization errors degrade the performance of cascade connection structures. Since arithmetic quantization

occurs at the output of each subfilter, the corruption of the signal at an early subfilter can be enhanced by subsequent subfilters resulting in overall quantization noise that is unacceptably large. In this section, a method to factorize a long FIR filter into several cascaded subfilters with minimum quantization noise is presented.

To reduce the quantization noise, all subfilters should be jointly designed to make the ripples of each subfilter compensate each other. One of the possible ways to solve the problem is to find optimal solutions by using exhaustive search, but it is extremely time-consuming and impractical even for moderate filters [18]. In [18], a method based on simulated annealing is proposed to factorize a long filter into combination of second and fourth-order subfilters. It is reported in [18] that their method can find comparable solutions with much shorter computational time in comparison with exhaustive search. However, the method proposed in [18] can only factorize a long filter into second and fourth order subfilters, which largely limits the application of the method. Simply cascading adjacent second and fourth order subfilters to construct a longer subfilter will largely change the quantization noise. This prompts the modification of the method in [18] to factorize a long filter into any desired number of subfilters. More freedom to select the factorization structure is given to the designers. In the modified method, after the order of zero-sets each containing 2 or 4 zeros of the long FIR filter are achieved, instead of constructing a cascaded subfilter using one zero-set like the method in [18], several successive zero-sets are grouped to form a subfilter. The quantization noise is calculated based on the resultant cascade structure. By this way, a filter can be factorized into any desired and reasonable number of subfilters.

2.3.1 Simulated Annealing Algorithm

Before we present the way to factorize a long filter, let us review the simulated annealing (SA) technique [19]. The SA derives its name from analogy with an annealing process in physics/chemistry where we start the process at high temperature and then lower the temperature slowly while maintaining thermal equilibrium. This technique makes use of stochastic methods via computer-generated pseudorandom numbers to find the optimum values of the cost functions that characterize large-scale and complex systems. It is especially suited for solving combinatorial optimization problems whose objective is to minimize the cost function of a finite discrete system that has a large number of possible solutions.

In the SA algorithm, the system starts from a high temperature and the temperature lowers slowly while maintaining thermal equilibrium. A cooling scheme is utilized in SA to avoid entrapment in local minima, which can also simplify and speed up the complex computation by eliminating the need of large computer memory. The search process terminates when the system becomes stable. There are two important parameters in SA algorithm, i.e. energy E and temperature T. The energy E is interpreted as a numerical cost and the temperature T as a control parameter. The numerical cost assigns to each configuration in the combinatorial optimization problem, a scalar value that describes how desirable that particular configuration is to the solution. The simulated annealing process will converge to a configuration of minimal energy provided that the temperature is decreased no faster than logarithmically. However, such an annealing schedule is too slow to be of practical use. In practice, we must resort to a finite-time approximation of

the asymptotic convergence of the algorithm. The price paid for the approximation is that the algorithm is no longer guaranteed to find a global minimum, but it can always produce near optimum for most practical application.

2.3.2 Minimization of Quantization Noise

Assuming a long FIR filter denoted as $H(z)$ is factorized into p numbers of short filter, its z-transform transfer function is shown in (2.3). The controllable quantization noise gain [18] is defined as

$$g_{qn} = \sum_{i=2}^{p} \frac{1}{2\pi} \int_{-\pi}^{\pi} | H_{ip}(e^{j\omega}) |^2 d\omega, \tag{2.5}$$

where p is the number of cascaded subfilters, $H_{ip}(z)$ represents the z-transform transfer function from the input of the i^{th} subfilter to the output of the p^{th} subfilter, i.e. $H_{ip}(z) = \prod_{m=i}^{p} H_m(z)$, and $H_{ip}(e^{j\omega})$ is the frequency response of $H_{ip}(z)$. The energy (E) is set to be the controllable quantization noise gain (g_{qn}), i.e. $E=g_{qn}$.

To avoid overflow at the output of each subfilter, it must be scaled to meet the following condition according to L_∞ -norm scaling [18]:

$$||H_{1i}||_\infty = \max | H_{1i}(e^{j\omega}) | = 1, \ 0 \le \omega \le \pi, \ 1 \le i \le p. \tag{2.6}$$

A program is developed based on this method, which accepts the continuous-time domain based coefficient values of a filter as input and outputs the optimal or sub-optimal factorization solution. There are three steps in the procedure.

1. Calculate the zeros of the long FIR filter *H*(z)

First, the real valued coefficients of the long filter $H(z)$ is designed by using REMEZ exchange algorithm. In this thesis, direct form filters with real valued coefficients are all

designed by using REMEZ algorithm. The zeros of $H(z)$ are found and grouped into different sets according to their types by using the method in [20]. Each set consists of 2 or 4 zeros according to their location in the complex z-plane and all sets are arranged in an arbitrary ordering.

2. Calculate the initial data for simulated annealing algorithm

This step is used to reduce the quantization noise gain to a certain level which is acceptable to the SA algorithm, where the sequence of the zero-sets are reordered in an iterative manner. The best noise gain calculated in this part is recorded as initial value of noise gain and temperature for the simulated annealing process. The reordering process is to choose a random length subsequence from a random position in the current zero sequence, either to replace it with the same subsequence in reversed order or to shift it to another random position between two sets of zeros in the current zero sequence with equal probability [21-22]. After each reordering process, the zeros are grouped according to their positions, i.e., the earliest several sets of zeros are put together into the first group, and construct a subfilter using all zeros in the same group. The number of the zero groups is equal to the number of the subfilters specified by the designers. The quantization noise gain based on the subfilter sequence is calculated using the cost function (2.5). If the noise gain is lower than the current lowest value in the early iterations, the noise gain and its associative zeros ordering and subfilter sequence are recorded as the current optimum results that will be used in the next reordering iteration process.

In this stage, the convergence condition is defined as the maximal number (L_{it}) of reordering iterations. The choice of L_{it} depends on the length of the overall filter, i.e. the

number of decision variables. More iteration loops are required when the filter order is higher, where the solution space is larger. However, since it is only a stage before the SA and also the computational cost is relative low, the set of L_{it} is not very much important. According to the experience, through about 100 reordering iterations (i.e. L_{it} = 100), the quantization noise gain can be reduced to an acceptable level, which can be utilized as initial solution in the SA algorithm.

3. Simulated annealing

Simulated annealing is applied in this step. The best noise gain and associated ordering of zero sets calculated in the second part are used as the initial solution. The value of temperature T is set to be the minima energy value calculated from the second part. Begin with these initial values, the reordering iterations are repeated, where the same method as used in the second part are applied to reorder the sequence of zero sets. In each reordering loop, the energy and its associated ordering of zeros and form of subfilter will be recorded as new best results to replace the current ones (such loop is called as successful one) only when one of the following two conditions are met. First, the energy value is smaller than the current best value; second, a generated random number which is uniformly distributed on (0, 1) is less than the value of exp (-ΔE/T), where ΔE denotes the change of the energy value. The aim of the second condition is to prevent SA from entrapping in local optima. In a successful loop, the temperature T would decreases by a specified factor (df_T). It can be chosen between 0.7 and 0.99, which gradually decreases the temperature to avoid trapping in a local optimum. Smaller df_T speeds up convergence while larger df_T may find better solutions at the cost of long computation time.

In [18], the authors used consecutive "misses" (the number of rejection of new ordering) as an indication of convergence. The SA process terminates if the number of consecutive misses reaches a given maximal value. However, such convergence criterion may not be sufficient without the control of T.

In our program, the reordering process terminates if one of the following two conditions is satisfied:

1) The temperature is smaller than a specified ending temperature (T_{end}).

2) There is no more improvement in the best energy value for the given number (L_{end}) of the continuous iteration process.

Smaller T_{end} provides SA more chances to find better solutions at the cost of computational effort, while larger T_{end} may make SA "miss" the optimal solutions by conducting less reordering loops. T_{end} should be chosen by considering the trade-off between computational cost and solution quality. Based on the simulation study, suitable value of T_{end} can be chosen from 10E-1 to 10E-6. It is found that the value of T_{end} is dependent on the number of the subfilters, i.e. smaller T_{end} is more competent for the design of high-order filters with more number of subfilters.

The choice of L_{end} is also based on the consideration of the trade-off between computational cost and solution quality. It can be seen from simulation study that if the best energy cannot be improved in 100 continuous iterations the improvement can be achieved only with a very small probability. Therefore, L_{end} can be set to be 100. Larger values are preferred if computational complexity is less important. However, the simulation study shows that larger value than 500 is unnecessary.

By grouping the successive zero-sets and minimizing the quantization noise based on the set of longer subfilter, an FIR filter can be factorized into the desired number of subfilters with near optimal arithmetic quantization noise gain, which largely reduces the frequency response deterioration caused by quantization. After achieving the optimization solution in continuous space, linear programming can be applied to quantize each coefficient into SPoT space for all subfilters. However, linear programming can only individually quantize the coefficients for each subfilter, which hinders the quantization from reaching optimal. In the following sections, systematic methods to jointly optimize all subfilters will be proposed.

In the next section, we will briefly review GAs before a new optimization scheme that employs a GA to optimize all subfilters in a discrete space is presented.

2.4 Genetic Algorithms (GAs)

Genetic algorithms (GAs) are proposed by J.H. Holland [16] in the early 1970's, whose basic idea and mechanism are borrowed from genetic evolution and natural selection, where the potential solutions to an optimization problem are evolved through selection, breeding and genetic variation. According to the principle of natural selection, in GAs, the stronger individuals are likely the winners in a competing environment. With the development of low cost and high speed computers, GAs, as a tool for search and optimization have reached a mature stage. In the last decade, GAs have found its

applications in a broad range of fields such as music generation, genetic synthesis, strategy planning, machine learning and VLSI technology.

In GAs, the potential solution for the optimization problem is represented by a set of encoded variables which are equivalent to the genetic material of individuals in nature. All variables are concatenated as the genes of a chromosome. Grouping a specified number of chromosomes generates a population which is manipulated by GAs to solve an optimization problem. GAs are based on the heuristic assumptions that the best solutions can be found in regions of the search space containing a relatively high proportion of good solutions and that these regions can be explored by the genetic operations of selection crossover and mutation. For each problem, there is a unique fitness function to provide the mechanism for evaluating the performance of each chromosome. Each solution is associated with a fitness value that reflects how good it is. The fitter chromosome with higher fitness value means a better solution to the optimization problem, which has a higher chance of survival and a tendency to produce good quality offspring. Since the new generation is possible to have better characteristics than the old one, an optimum solution for the given problem can be obtained. The following shows the summary of the basic genetic algorithm structure.

Genetic algorithm () [23]

{

Initialize population;

Evaluate population;

While termination criterion not reached

```
        {
        select solutions for next population;
        perform crossover and mutation;
        evaluate population;
        }
    }
```

There are three operators: reproduction, crossover, and mutation in the basic GAs structure.

1) Reproduction

In reproduction operation, fitter chromosomes are selected from the current population and copied into a mating pool for further genetic operation according to a given parent selection scheme. A chromosome with a higher fitness value has a higher probability to contribute one or more offspring in the next generation. It means highly fit chromosomes have better characteristic and have more chance to be chosen and allowed to mate. Roulette Wheel Selection is one of the most commonly used techniques for selection mechanism.

The Roulette Wheel Selection procedure is as follows:

A. Sum the fitness of all individuals in the population, named as total fitness (f_T).

B. Generate a random number, n, between 0 and f_T.

C. Return the first individual whose fitness, added to the fitness of the preceding individuals, is greater than or equal to n.

2) Crossover

Crossover is the basic operation for yielding new chromosomes. The gene information contained in the two chosen parents is recombined to generate two children which expected to have useful information from their parents. In GAs implementation, a certain binary strings of two parent's chromosomes are exchanged to yield two children chromosomes with specific crossover probability ranging from 0 to 1.

Basically, there are three methods to carry out crossover, i.e. single-point crossover, multipoint crossover and uniform crossover. Single-point crossover is the simplest implementation method, where crossover is carried out at a single point. Multipoint crossover is similar to single point crossover except that several crossover points are randomly selected. Uniform crossover is performed over the entire string length of bits. Firstly, a mask with the same length as the chromosome is generated randomly which consists of a bit string of "0" or "1". The children are generated according to the information in the mask, i.e. if the i^{th} bit of mask is 0, the i^{th} bit of Child 1 is the same as Parents 1 and the i^{th} bit of Child 2 is the same as Parents 2; if the i^{th} bit of mask is 1, the i^{th} bit of Child 1 is the same as Parents 2 and the i^{th} bit of Child 2 is the same as Parents 1.

3) Mutation

The offspring from crossover operation is sent to mutation operation that introduces new genetic material into the population to maintain the diversity of population. In GAs implementation, the mutation operation examines every bit contained in the offspring and randomly alters it with a very small probability. The mutation probability is typically in the range 0.001-0.05, which is dependent on the size of population and the length of individual chromosome.

In the GAs implementation, there are several control parameters which critically affect the optimization performance of the GAs, i.e. population size, the probabilities of crossover and mutation. With small population, the GAs easily converge into local optima; while with large population the optimization process requires long computation time and huge amount of computer resource to find optimal solutions. As for crossover and mutation operations, increasing the crossover probability increases the recombination of parents' information but it also increases the disruption of good strings; increasing the mutation probability tends to transform the genetic search into a random search but it also helps introduce new genetic material. The trade-off between solution quality and computational cost should be carefully considered in the implementation of GAs. A number of guidelines have been recommended for choosing these parameters, e.g. in [23-29]. However, these guidelines are inadequate as the choice of the optimal parameters becomes specific to the problems to be optimized. Many researchers have put considerable efforts into the variation of the simple GA by relieving the users of the burden of specifying the values of these parameters [27]. Greferstellt [27] has taken the choice of probabilities of crossover and mutation as an optimization problem and recommended the use of a second-level GA to determine these parameters. However, since two levels of GA are needed to run simultaneously, Greferstellt's approach is proven to be computationally expensive. The idea of adapting crossover and mutation operators to improve the performance of GAS has been employed in [30-33], which adaptively adjust these parameters according to the optimization performance or the scattering characteristic of the population. Many researchers have proposed useful

variations on genetic operations by hybridizing GA with other algorithms, e.g. in [59-62]. Although many researchers have proposed valuable ideas in the implementation of adaptive GAs or hybrid GAs, with the increasing complexity of optimization problems in real world, better technologies for GA implementation are still in need. In the Section 2.6, an effective adaptive GA is introduced for the optimization of FIR filters in a discrete space.

2.5 GA for the Design of Low Power High-Speed FIR Filters

2.5.1 GA Implementation

The optimal design of cascade form filters can be possibly achieved if all subfilters are jointly optimized. High problem dimension and large search space lead to numerous local optima, which increases the computational complexity. This is especially serious for high-order filters. The GA may provide a solution due to three reasons. First, all subfilters can be simultaneously designed. Second, the coefficients can be quantized in SPoT space without rounding process as the GA directly deals with coded variables in a discrete space. Third, the information directly related to hardware cost can be affiliated to the fitness function as an optimization criterion.

When the GA is applied to design a digital filter in cascaded realization, the coefficients of all subfilters are encoded and concatenated to represent the chromosomes of filters as a string of ternary encoding digits [11]. The word length of filter coefficient is equal to the ternary string length. Cascading the encoded coefficients of all subfilters forms a vector

of ternary bit stream, named as chromosome. Such chromosomes proceed to the genetic operation and the new generation of chromosomes is produced as a potential solution to the optimization problem.

Three important issues on the design technique, such as initialization, objective function and fitness function, and replacement strategies are covered here.

A. Initialization

Before coming to the optimization process, several design parameters should be decided, such as the number of subfilters, the lengths of subfilters and the initial population.

The z-transform transfer function of the overall filter, $H(z)$, is given in EQ. (2.3), which is constructed by p cascaded subfilters shown in Fig. 2.2. The value of p can largely affect the hardware cost involved in filter implementation. We shall show this through a design example. Let us consider the design of a linear phase FIR low-pass filter that meets the following specifications: the normalized passband and stopband edges of the filters are 0.15 and 0.22, respectively, the maximum passband ripple and minimum stopband attenuation are 0.01 and 40 dB, respectively. The overall length of the filter with SPoT coefficient values is 53 [2]. To check how the number of subfilters affects the hardware cost, the filters are designed with 2, 3, 4 and 5 cascaded subfilters, respectively. The hardware cost is calculated based on the following assumptions. First, all subfilters are implemented by Direct-Form I; Second, the input signal word length is 8-bit; Third, each delay element is implemented by D-flip-flop with 8 transistors; Fourth, all adders are carry ripple adders with 28 transistors for each 1-bit full adder cell. In this thesis, the calculation of hardware cost is all based on these assumptions. Fig. 2.3 shows the

relationship between hardware cost in terms of the number of transistors and the number of subfilters in the cascaded filters. It is clear from the figure that the number of subfilters largely affects the hardware cost. It can be seen that the relationship between the number of subfilters and the number of transistors is not a linear one. In a cascade from filter, the output signal of the earlier stage is taken as the input signal of the later stage, which causes an increase in the word length of the overall output signal, consequently, an increase in hardware requirement. Such situation is increasingly serious with the increase of the number of subfilters. Hence, there is an increase trend for the hardware cost when a filter is factorized into more subfilters. However, to realize a filter in cascade form, it is possible to use shorter word length to represent the coefficients while still keeping the peak ripple [2]. Therefore, more subfilters possibly leads to shorter word lengths, which can in turn reduce the hardware requirement. Hence, with the increase of the number of subfilters, the hardware requirement in terms of the number of transistors may be increased and also may be decreased. This can explain why it is not a linear relationship in Fig. 2.3. It is interesting and reasonable to assume that there is an optimal factorization mode that minimizes the hardware cost. In this example, the filter consisting of 3 cascaded subfilters requires fewest transistors. The simulation study shows that p can be set to be 3 for the designs with middle lengths ranging from 20-60, which is a safe choice. The lengths of subfilters depend on three factors. The first is the overall filter length denoted as N; the second is the value of p; and the third is the zeros of the overall filter. If the lengths of the subfilters in Fig. 2.2 are denoted as N_i, i=1, 2,…p, the relationship between N and N_i can be given as

$$N = \sum_{i=1}^{p} N_i - p + 1. \tag{2.7}$$

Here, we assume that the overall filter length keeps unchanged when it is implemented in both cascade form and direct form. For example, we can factorize a 26-order filter into two 13-order subfilters, such that the length of the overall filter is still 27, i.e. 27=14×2-2+1.

When selecting the lengths of subfilters, we try to avoid large differences among the subfilters. Furthermore, the lengths of subfilters are also dependent on the distribution of zeros of the filter with real valued coefficients. In other words, we need to keep the integrity of the zero-sets grouped according to their location in the complex *z*-plane. The length of each subfilter is chosen according to the above considerations.

In the GA, the initial population as seeds to start the optimization process can affect the optimization performance to a certain degree. In our implementation, a long filter is firstly designed in cascaded realization with real valued coefficients using the method proposed in Section 2.3 and then these coefficients are simply rounded to the pre-specified coefficient precision. By this way, an initial filter with discrete coefficients can be obtained. The coefficient values of this filter are perturbed to achieve the initial population.

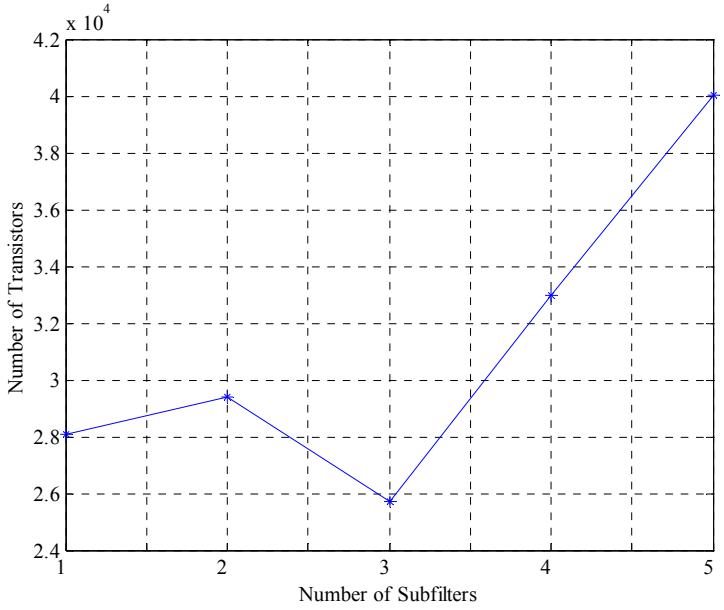


Fig. 2. 3 The relationship between hardware cost and the number of subfilters.

B. Objective function and fitness function

The objective function is a main source to provide the mechanism for evaluating the status of each chromosome. This is an important link between the GA and the problem to be optimized. It takes chromosomes as input and produces objective values as measurement to the chromosomes' performance.

According to the minimax error criterion, the objective function of an FIR filter can be written as

$$O = \max_{\omega \in [0,\omega_p] \cup [\omega_s,\pi]} [\, k(\omega) \,|\, H(e^{j\omega}) - H_d(e^{j\omega}) \,|\,], \tag{2.8}$$

where ω_p and ω_s are the passband and stopband edges, respectively, $k(\omega)$ is the required positive weight in each band, and $H(e^{j\omega})$ and $H_d(e^{j\omega})$ are the frequency responses of the filter under design and desired filter, respectively.

The fitness value calculated from the fitness function is used to reflect the degree of excellence of a chromosome. The information about the number of bits and the number of SPoT terms used in the coefficients is added to the fitness function as an optimization criterion. The fitness function is defined as

$$f = \frac{1}{O} + \frac{a_1}{S_T} + a_2 \times \sqrt[p]{\frac{1}{Diff}}\ , \tag{2.9}$$

where S_T represents the total number of the SPoT terms used in all subfilters, p is the number of subfilters, and *Diff* can be expressed as

$$Diff = \prod_{i=1}^{p} \left[\max |h_i| - \min |h_i| \right], \tag{2.10}$$

where h_i is the coefficients of the i^{th} subfilter. In the fitness function, a_1 and a_2 are positive weighting coefficients which affect the optimization results by controlling the contribution of an individual term in the fitness function. The first term in fitness function plays a major role, which controls the minimization of the peak ripples. The second term is used to minimize the number of SPoT terms and the third term is to minimize the number of bits used to represent the coefficients of each subfilter. It is important to assign proper values to the weighting coefficients, a_1 and a_2, such that the second and third terms in (2.9) play a minor role. Usually at the beginning of the evolutionary process, the fitness value of first term is very small, i.e. less than 1, especially in the case where the initial population is randomly obtained. Hence, a_1 and a_2 should be chosen in a way that the sum of the second and third terms in (2.9) is less than 1 for all acceptable solutions. Since the second term is related to the number of SPoT terms, the value of a_1 can be a function of the pre-specified maximal number of SPoT

terms used in all subfilters, i.e. $a_1 = \alpha S_S$, where S_S is the pre-specified maximal number of SPoT terms. To find the two coefficients a_1 and a_2, a large number of filters with lengths ranging from 20 to 70, ripple from 0.1- 0.0001, and coefficient word length from 2 to 12 bits, are designed. If there is no special statement, these specifications are also used to collect simulation data in the later sections. We obtained the suitable values of a_1 and a_2 for various filters with these specifications through multiple trials. Based on the data collected, a_1 and a_2 are found to have close relationship with the filter length. From simulation study, we derived the following equations to compute the weighting coefficients, a_1 and a_2:

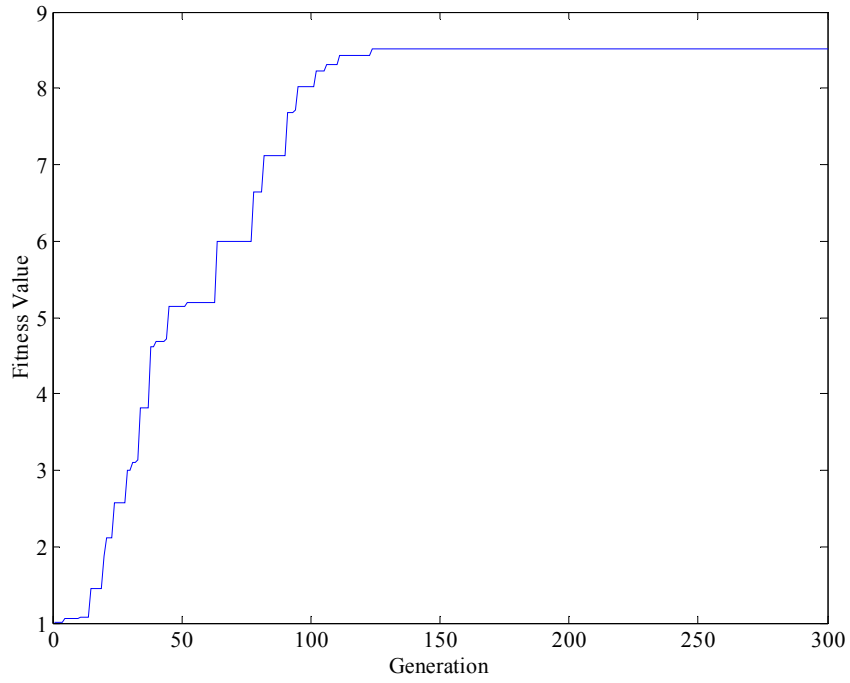
$$a_1 = \frac{\log_{10} N_T}{2^6} S_S$$
$$a_2 = \log_{10} N_T \ , \tag{2.11}$$

where N_T is the total length of all subfilters. To express theses parameters using equations is better than to give fixed values or suitable ranges for the choice of these values. It is more flexible and can avoid multiple trials to find the correct values. With the definition of the fitness function such as EQ. (2.9), the GA not only minimizes the peak ripples of the overall filter but also minimizes the total number of SPoT terms and the number of bits for all subfilters; consequently, the arithmetic operations of the filter is minimized.

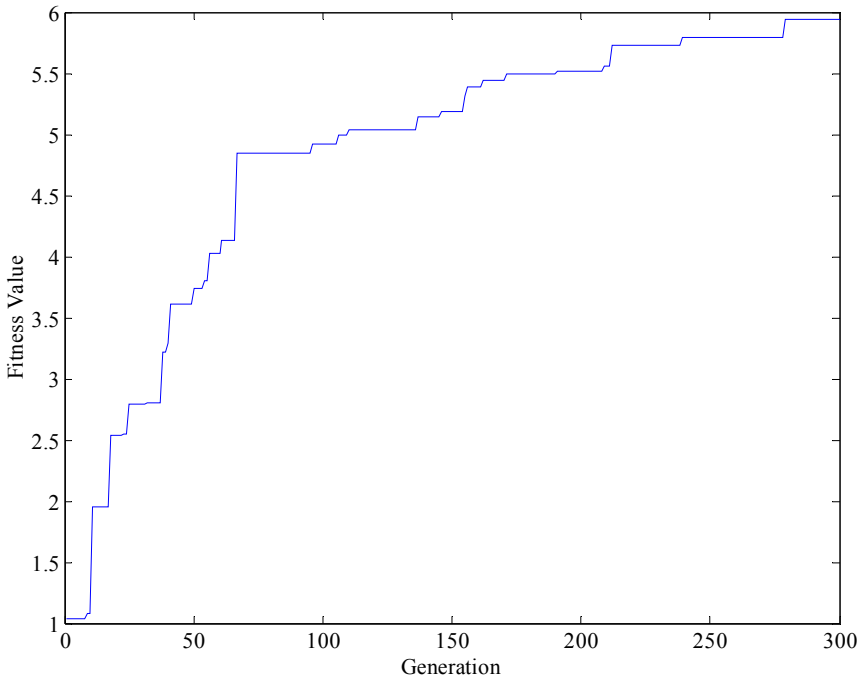
C. Replacement strategies

Replacement is an important stage in the evolutionary process of the GA, which decides how to replace the current population to form the new population after the offspring is produced by genetic operations. In our implementation, generation-replacement [34] is used as the main replacement strategy. In this strategy, each population generates the

same number of new chromosomes to form the population in the next generation that replaces all individuals in the previous generation. To increases the convergence speed, the generation-replacement strategy is combined with the elitist strategy [34] where two best chromosomes in old generation are copied into the succeeding one. Steady-state reproduction [34] is another strategy for replacement, where only a few worst chromosomes of old generation are replaced to produce the next generation. According to our observation, if the population size is larger than 100, the generation-replacement has higher fitness value and convergence speed, resulting in better optimization performance. It can be seen from the simulation results of Fig. 2.4, where the first 300 generations are observed in a design of an FIR filter of order 20. In Fig. 2.4 (a), the generation-replacement strategy is used and the population and mating size are both 200. In Fig. 2.4 (b), the steady-state reproduction is applied and the population and mating size are 600 and 200, respectively. It is clear that the performance of the generation-replacement is better than that of steady-state reproduction.



(a) Generation-Replacement



(b) Steady-State Reproduction

Fig. 2. 4 The convergence results by using Generation-Replacement and Steady-State Reproduction.

2.5.2 Design Example

To illustrate the proposed technique, the design of a linear phase FIR low-pass filter that meets the following specifications is considered:

Normalized passband edge: 0.15

Normalized stopband edge: 0.22

Maximum passband ripple: 0.01

Minimum stopband attenuation: 40 dB

An FIR filter of length 30 with real valued coefficients satisfies the given specifications. The filter length will increase to 53 if each coefficient is quantized into 3 terms of 7-bit SPoT. If such a filter is factorized into 2 subfilters by method in [2], the subfilter lengths of $H_1(z)$ and $H_2(z)$ are both 27. The coefficients of $H_1(z)$ are represented by 2 terms of 5-bit SPoT and $H_2(z)$ is represented by 6-bit SPoT. The minimum stopband attenuation of the overall filter is 40.28 dB. When the filter is designed using the proposed method, the lengths of $H_1(z)$ and $H_2(z)$ are also both 27. Two terms of 5 bits SPoT term is sufficient to represent the coefficients $H_1(z)$ and $H_2(z)$. The stopband attenuation is 40.25 dB. The frequency responses of the two subfilters and overall filter are shown in Fig. 2.5.

If the filter is factorized into three subfilters by using the proposed method, the lengths of $H_1(z)$, $H_2(z)$ and $H_3(z)$ are 19, 17 and 19, respectively. The coefficients of $H_1(z)$ and $H_2(z)$ use 3 terms of 4 bits SPoT, and 2 terms of 4 bits SPoT, respectively. The coefficients of $H_3(z)$ use 2 terms of 3 bits SPoT. The stopband attenuation is 41.46 dB. The frequency responses of three subfilters and overall filter are shown in Fig. 2.6. The coefficient values of three subfilters are listed in Table 2.1.

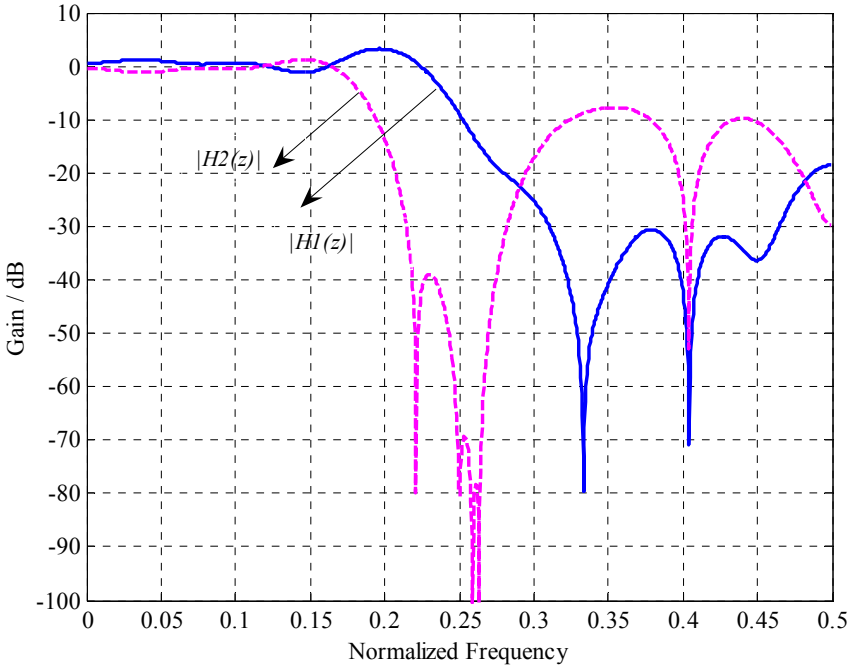
A comparison on hardware cost for the above filters are listed in Table 2.2. From the table, the filter consisting of 3 subfilters designed by the proposed method uses 7.21% less D flip-flops and 24.24% less full adders than the one including 2 subfilters designed by the iterative approach in [2]. This is mainly because the GA optimizes all subfilters simultaneously while the linear programming does it separately. Moreover, the minimization of the coefficient word length is also incorporated as an optimization criterion. It can be seen from Table 2.2 that compared with the filter designed by [2], the proposed method achieves about 20.74% savings in terms of the number of the transistors.

Table 2. 1 List of filter coefficients in three-subfilter structure

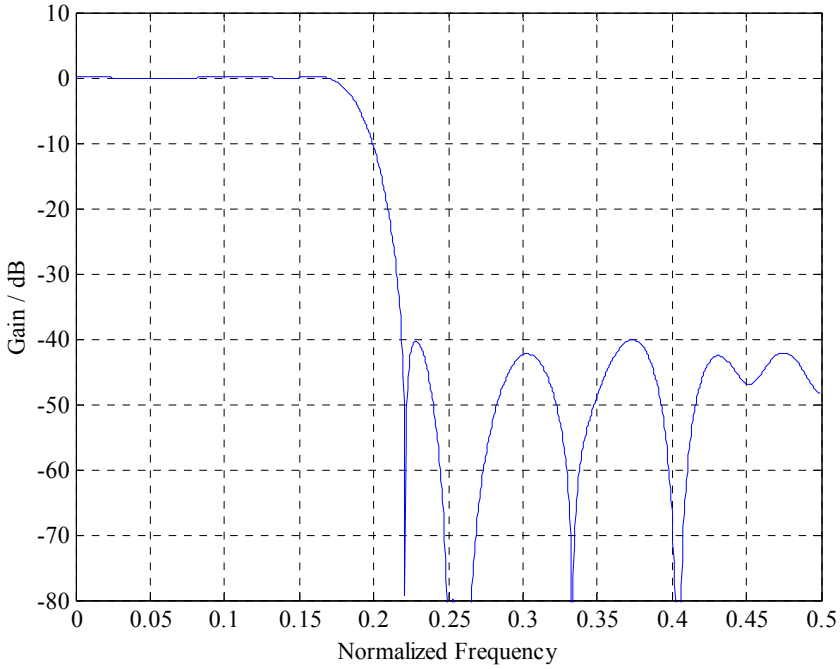
h_1	h_2	h_3
$h(0) = h(18) = 2^0$ $h(1) = h(17) = 2^1$ $h(2) = h(16) = -2^1$ $h(3) = h(15) = 0$ $h(4) = h(14) = -2^0$ $h(5) = h(13) = 2^1$ $h(6) = h(12) = 2^0$ $h(7) = h(11) = -2^1$ $h(8) = h(10) = -2^3+2^0$ $h(9) = -2^3-2^2+2^0$	$h(0) = h(16) = 2^0$ $h(1) = h(15) = 2^1$ $h(2) = h(14) = 2^0$ $h(3) = h(13) = 0$ $h(4) = h(12) = -2^0$ $h(5) = h(11) = 0$ $h(6) = h(10) = 2^2$ $h(7) = h(9) = 2^3$ $h(8) = 2^3+2^1$	$h(0) = h(18) = -2^0$ $h(1) = h(17) = 2^0$ $h(2) = h(16) = 0$ $h(3) = h(15) = 2^0$ $h(4) = h(14) = -2^0$ $h(5) = h(13) = 0$ $h(6) = h(12) = 0$ $h(7) = h(11) = 2^2-2^0$ $h(8) = h(10) = -2^2+2^0$ $h(9) = -2^2-2^0$

Table 2. 2 A comparison of hardware cost among different designs

	Single filter $H(z)$	Iterative approach [2] $H_1(z)$ $H_2(z)$	Proposed $H_1(z)$ $H_2(z)$	Proposed $H_1(z)$ $H_2(z)$ $H_3(z)$
Stopband attenuation (dB)	41.06	40.28	40.25	41.46
No. of bits	7	5,6	5,5	4,4,3
Total No. of SPoT	36	35	33	30
No. of D flip-flops	780	832	806	772
No. of 1-bit full adders	780	920	819	697
No. of transistors	28080	32416	29380	25692

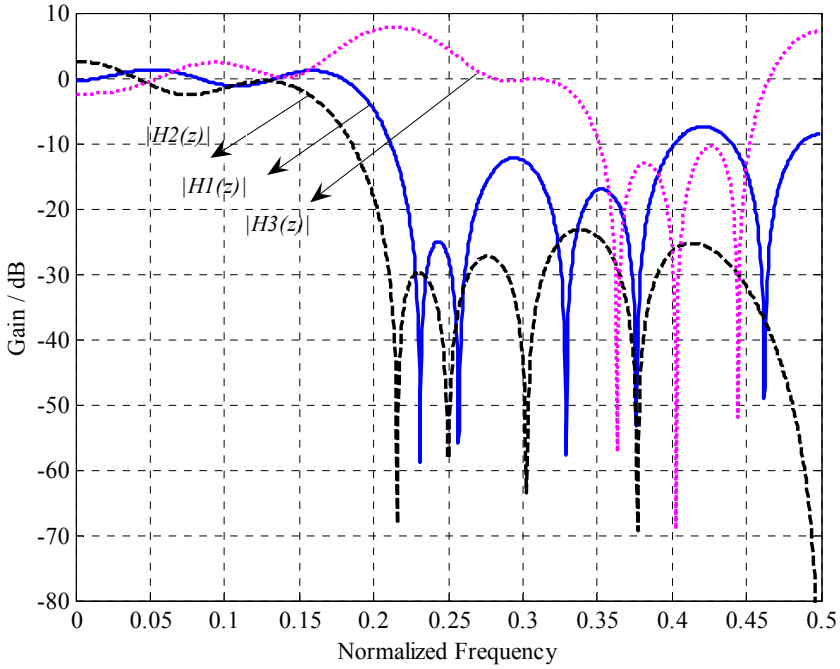


(a) Two subfilters $H_1(z)$ and $H_2(z)$ whose orders are both 26.

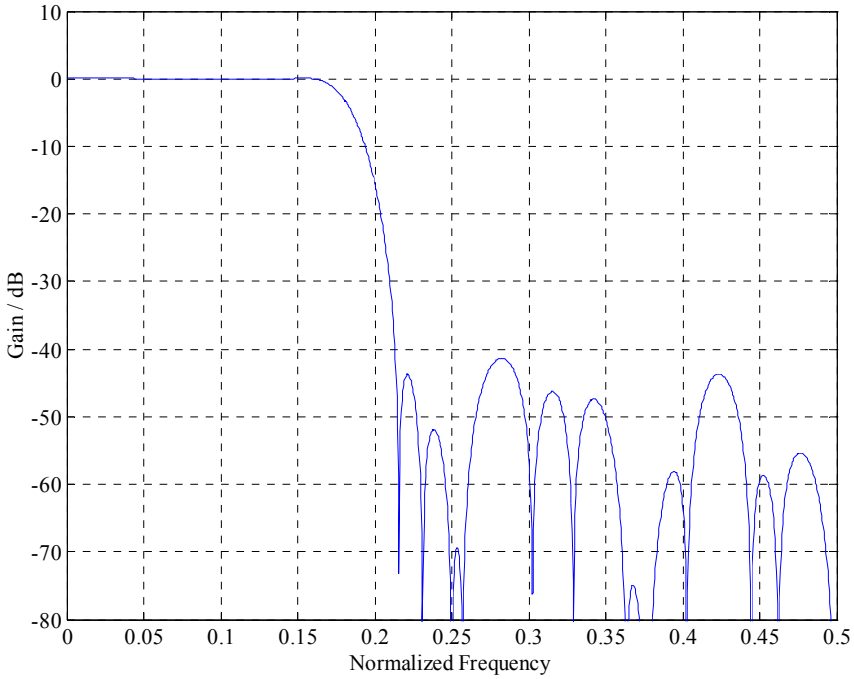


(b) Overall filter whose order is 52.

Fig. 2. 5 The frequency responses of the two subfilters (a) and overall filter (b).



(a) Three subfilters $H_1(z)$, $H_2(z)$ and $H_3(z)$ whose orders are 18, 16, 18, respectively.



(b) Overall filter whose order is 52.

Fig. 2. 6 The frequency responses of the three subfilters (a) and overall filter (b).

2.6 An Adaptive Genetic Algorithm (AGA)

In Sections 2.4 and 2.5, the GA has been introduced and applied in the design of cascade form filters with SPoT coefficients, respectively. The most difficult thing in the implementation of the GA is the choice of control parameters which are problem-dependent and critical to optimization performance. The balance between optimization performance and computational cost should be carefully considered when choosing the suitable parameters for an optimization problem. Moreover, the optimal settings of parameters in the earlier search stages typically become inefficient in the later stages.

To address the problem, an adaptive genetic algorithm (AGA) is proposed in this section. Instead of using fixed population size and probabilities of genetic operations, the proposed AGA automatically adapts the population size and the probabilities of genetic operations to the change of optimization performance and population dynamics during the evolutionary process. It is, therefore, named as “adaptive genetic algorithm”. The parameter-free characteristic avoids multiple trials to find optimal settings for these control parameters.

2.6.1 Adaptive Population Size

The size of the population is one of the most important parameters, which is critical in GA applications. The population size is dependent on the sizes of chromosome and the search space. Increasing the population size increases its diversity and reduces the probability that the GA prematurely converges to local optima, while it increases the computation time required for large population to converge to optimal regions in search

space. To ensure the satisfying optimization performance as well as enhance the calculation efficiency, in the proposed AGA, the population size adaptively varies according to the average improvement of the best fitness values during a specified number of generations. In our implementation, during two successive evolution periods, if the average fitness value of the fittest individuals in the second period is larger than that in the first period, the size of the population pool will decrease. On the other hand, the population size will increase if the average value in the second period is smaller than that in the first period. The adaptive change of the population size (denoted as ΔP) depends on the improvement quantity, which is expressed as

$$\Delta P = \begin{cases} \left\lfloor \dfrac{f_{avgo} - f_{avgn}}{f_{avgn}} \right\rfloor * P_{org}, & \text{if } f_{avgn} - f_{avgo} >= 0 \\ \left\lfloor \dfrac{f_{avgo} - f_{avgn}}{f_{avgn}} \right\rfloor * P_{org}, & \text{if } f_{avgn} - f_{avgo} < 0 \end{cases} \qquad (2.12)$$

$$\text{where } f_{avgo} = \frac{\sum_{i=G_S}^{G_S+G-1} f_{besti}}{G}, \; f_{avgn} = \frac{\sum_{i=G_S+G}^{G_S+2G-1} f_{besti}}{G}.$$

P_{org} is the current population size, f_{besti} is the best fitness value in i^{th} generation, G_S is the starting generation, and f_{avgo} and f_{avgn} are the average of the best fitness values during a pre-specified number (denoted as G) of generations, i.e. from G_S^{th} generation to $(G_S+G-1)^{th}$ generation and the same number of offspring generations, i.e. from $(G_S+G)^{th}$ generation to $(G_S+2G-1)^{th}$ generation, respectively..When ΔP is a positive value, the population size will increase; when ΔP is a negative value, the population size will decrease; when ΔP is zero, the population size will remain unchanged. The absolute value of ΔP is equal to the change of the population size.

At the beginning, the population size is chosen to be small. During the evolutionary process, the population size increases or decreases according to the average improvement of the best fitness values between two consecutive periods each with the same pre-specified number of generations as defined in EQ. (2.12). Upper and lower bounds are specified for the population size, e.g. the lower bound can be set to be 10 and the upper bound to be 1000. If the change of population size calculated from EQ. (2.12) makes the population size larger than the upper bound or smaller than the lower bound, the population size will be set to be the upper bound or lower bound, respectively, which ensures that the population size is always in the reasonable range.

2.6.2 Adaptive Probabilities of Crossover and Mutation

In the evolutionary process of the GA, crossover and mutation are two important genetic operations used to produce new members for the offspring, which occur according to the probabilities of crossover and mutation, denoted as p_c and p_m, respectively. The performance of the GA is largely dependent on the choice of p_c and p_m. The higher the value of p_c is, the faster the introduction of new material is. However, increasing p_c also increases the disruption of good chromosomes. The large value of p_m tends to transform the GA optimization process into a random search, but it also helps prevent the GA from premature convergence.

In the AGA, the method in [30] for the adaptive adjustment of p_c and p_m is adopted, where p_c and p_m increase when the population tends to get stuck at a local optimum and decrease when the population is scattered in the search space

$$p_c = \begin{cases} k_1(f_{\max} - f')/(f_{\max} - f_{avg}), & f' \geq f_{avg} \\ k_3, & f' \leq f_{avg} \end{cases}, \tag{2.13}$$

$$p_m = \begin{cases} k_2(f_{\max} - f)/(f_{\max} - f_{avg}), & f \geq f_{avg} \\ k_4, & f \leq f_{avg} \end{cases}, \tag{2.14}$$

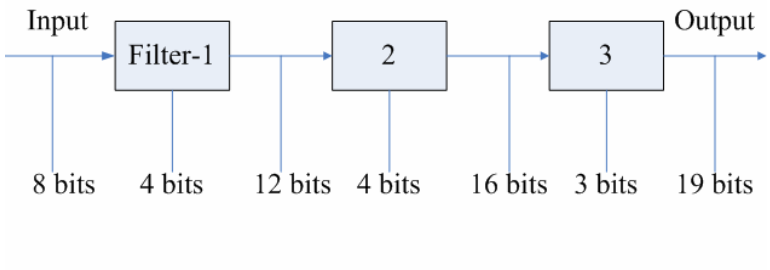
where f_{max} and f_{avg}, respectively, are the maximum and average fitness values in the current population, f' is the larger fitness value of the two chromosomes to be crossed and f is the fitness value of the mutated chromosome. According to [30], the weighting coefficients, k_1 and k_3 are chosen to be 1.0 and k_2 and k_4 to be 0.5.

2.7 AGA for the Design of Low Power High-Speed FIR Filters with Truncation Effect

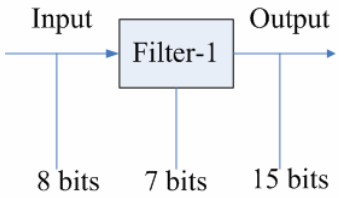
In Section 2.5, a GA has been presented for the design and optimization of high-speed low power FIR filters, where the filters are implemented by cascading several short subfilters. Design example has shown that significant savings on hardware cost can be achieved. However, we would like to know whether there are still rooms for improvement.

Let us examine the filter structure of Fig. 2.2. When the input signal passes through a filter implemented with a series of cascaded subfilters, the output signal of the earlier stage is taken as the input signal of the later stage. Hence, the word length of the overall output signal is the sum of the word lengths of the input signal and all subfilters. It can be illustrated by using the following example, shown in Fig. 2.7. The word lengths of input signals are both 8 bits in the two cases. If the filter is implemented in the direct

form with coefficients of 7 bits SPoT terms, the word length of the output signal is 15 bits; while if the filter is designed with 3 cascaded subfilters with coefficients of 4, 4, and 3 bits, respectively, the word length of the output signal is increased to 19 bits due to the accumulation of the word lengths of the input signal and each cascaded subfilter.



(a) The cascade form filter



(b) The direct form filter

Fig. 2. 7 The word lengths of the output signals in different forms of realization.

The increase in the word length of the output signal will result in an increase in the memory to store the signal. To address this problem, a certain number of the least significant bits are considered to be truncated from the output signal of subfilters. However, this would necessarily introduce quantization errors to the output signal. Moreover, such approach increases the complexity of the optimization process, leading to very long computation time. In this section, a systematic method is proposed, where several least significant bits are truncated from the output signals of the subfilters. To

improve the optimization performance, the AGA proposed in the previous section is utilized to jointly optimize each coefficient in all subfilters.

2.7.1 AGA Implementation for Cascaded FIR Filter Design

The chromosome form and initialization process are the same as the method presented in Section 2.5. The only difference is instead of using a fixed number of SPoT terms for all coefficients in one subfilter, the coefficients are allocated with different numbers of SPoT terms subject to a given total number of SPoT terms.

In a cascade form filter shown in Fig. 2.2, the frequency response of the output signal can be expressed as

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega}), \tag{2.15}$$

where $X(e^{j\omega})$ is the frequency response of the input signal. An impulse signal is used as input to determine the frequency response of the filter. When the impulse signal passes through the system, truncating several least significant bits from the output is performed at each stage of cascaded subfilters if applicable. .

By considering the truncation effect in the filter output, the objective function is formulated as

$$O = \max_{\omega \in [0,\omega_p] \cup [\omega_s,\pi]} \left[k(\omega) \left| Y(e^{j\omega}) - Y_d(e^{j\omega}) \right| \right], \tag{2.16}$$

where $k(\omega)$ is defined as the required ratio in each band, and $Y(e^{j\omega})$ and $Y_d(e^{jw})$ are the frequency response of the output sequence from the designed filter and ideal one, respectively.

Since zero-value coefficients reduce the hardware cost, an additional item to maximize the number of zero-value coefficient is added to the fitness function of EQ. (2.9), shown as

$$fitness = \frac{1}{O} + \frac{a_1}{S_T} + a_2 \times \sqrt[p]{\frac{1}{Diff}} + a_3 \times Z_T, \tag{2.17}$$

where Z_T is the total number of zero-value coefficients.

Since the new terms is related to the number of zero-value coefficients and the maximal number of such coefficients is the total length of all subfilters, the value of a_3 can be a function of N_T, where N_T is defined as the total length of all subfilters. Through similar simulation approach, it is found that the definition of a_1 and a_2 are the same as in EQ. (2.9) and a_3 can be defined as

$$a_3 = \frac{\log_{10} {N_T}^2}{N_T}. \tag{2.18}$$

2.7.2 Truncation Effect on the Cascaded Structure

It is important to understand the characteristics of truncation, e.g. how many bits can be truncated from the output of different stages. To this end, filters with different specifications are designed using the method presented in this section with length ranging from 21 to 65, peak ripple from 0.01 to 0.001 and the word lengths from 2 to 12 bits. From simulation study, some useful observations are concluded below.

1. The maximal number of bits to be truncated increases with the increase of the number of cascaded subfilters. However, the difference among the word lengths of the overall output in the filters with different number of subfilters is very small. Obviously, more bits to be truncated lead more savings of hardware cost. However,

the hardware cost may be high if a filter is factorized into too many subfilters. It is caused by the accumulation of the word lengths of the input signal and all subfilters. Therefore, as shown in Fig. 2.3, factorizing a filter into too many subfilters, e.g. larger than 3 subfilters, may require relatively high hardware cost. However, the relationship between the hardware cost and the number of subfilters is not a linear one. Thus, our conclusion here is not contradict with Table 2.2 which shows that the hardware cost of the filter consisting of 3 cascaded subfilters is less than that consisting of 2 cascaded subfilters. It is shown in Section 2.5 that there is an optimal point where the filter can be implemented with the minimal hardware cost. It is interesting to find that the optimal points in the designs with truncation and without truncation are usually similar. In other words, the quantity relationship between the number of subfilters and the hardware cost may not be changed via truncation.

2. The word length of the overall output in the direct form filter is often shorter than that in the cascade form filter. However, by using the proposed approach, the cascade form filter can be designed with nearly the same word length of the overall output as that in the corresponding direct form filter.

3. The truncation sensitivities of the subfilters located in different cascaded stages are different. It is meaningful to find how many bits can be truncated from each cascaded subfilters. Since the truncation effect in an earlier stage will propagate to all later stages which are stationed after it, the earlier stage has less truncation margin and more sensitive than the later stages. It is found that the first

subfilter is most sensitive, from which nearly no bit can be truncated. An empirical equation to estimate the truncation margin will be given in Section 2.7.3.

2.7.3 Optimal Truncation Margin

There are lots of design parameters which affect the truncation margin, i.e. the coefficient word lengths, filter lengths, the number of subfilters, and real valued coefficients. Since the earlier stages may have less truncation margin than those in later stages, the ratio (j/p) can be considered as one of factors that affect the truncation margin if j is defined as j^{th} subfilter in a chain of cascaded filters and p is the total number of subfilters. Based on the same reason, it is possible that the ratio (N_{sub}/N) can be used as an indication for truncation margin if N_{sub} is the sum of the lengths of 1st to j^{th} subfilters (i.e. $N_{sub} = \sum_{i=1}^{j} N_i$, where N_i is the length of the i^{th} subfilter) and N is the length of the overall filter. Besides the above two ratios, it is easy to conclude that the truncation margin should be proportional to the number of bits used. Putting all three factors together and based on data collected from the design of filters with the specifications given in the previous section, we propose an empirical equation below where TM_j represents the maximal number of bits which may be truncated from the output of j^{th} cascaded subfilter.

$$TM_j = \left\lfloor \left(\frac{j}{p}\right)^2 \left[\sqrt{\frac{2^{B_j}}{\max|h_j|}} \left(\sum_{i=1}^{j} \frac{N_i}{N} \right)^2 + \frac{p}{j} \right] \right\rfloor, \quad \text{for } j = 1, 2, ..., \left\lfloor \frac{p}{2} \right\rfloor,$$
$$TM_j = \left\lceil \left(\frac{j}{p}\right)^2 \left[\sqrt{\frac{2^{B_j}}{\max|h_j|}} \left(\sum_{i=1}^{j} \frac{N_i}{N} \right)^2 + \frac{p}{j} \right] \right\rceil, \quad \text{for } j = \left\lfloor \frac{p}{2} \right\rfloor + 1, ..., p, \tag{2.19}$$

where h_j presents the coefficients of the j^{th} subfilter in continuous space computed using the method proposed in Section 2.3, B_j is the number of bits to represent the coefficients of the j^{th} subfilter, and i and j are i^{th} and j^{th} subfilters, respectively. EQ. (2.19) is an experimental equation which can help to estimate the approximate truncation margin.

Three examples with relative short, medium and long filter lengths are used to show the veracity of EQ. (2.19). The specifications of the three filters are listed in Table 2.3. The comparisons on truncation margin between simulation and computation results are illustrated in Table 2.4. The simulation results are the maximal truncation margin that we can achieve using the method proposed in Section 2.7.1, and the computation results is from EQ. (2.9). It is clear that the equation can provide an approximate estimation on truncation margin where the error is no more than 1 bit.

Table 2. 3 The specifications of three filters with short, medium and long lengths

	Overall filter length	Normalized passband edge	Normalized stopband edge	Maximum passband ripple	Minimum stopband attenuation
Short filter	21	0.15	0.27	0.01	40 dB
Medium filter	35	0.15	0.24	0.01	40 dB
Long filter	53	0.15	0.22	0.01	40 dB

Table 2. 4 A comparisons of truncation margin between simulation and computation results

(a) Short filter

	2 subfilters	3 subfilters	4 subfilters
Simulation results (No. of bits)	0,5	0,2,3	0,0,*4*,*5*
Calculation results (No. of bits)	0,5	0,2,3	0,0,3,4

(b) Medium filter

	2 subfilters	3 subfilters	4 subfilters	5 subfilters
Simulation results (No. of bits)	0,3	0,1,4	0,0,2,3	0,0,2,2,3
Calculation results (No. of bits)	0,3	0,*2*,*5*	0,0,2,3	0,0,2,2,3

(c) Long filter

	2 subfilters	3 subfilters	4 subfilters	5 subfilters	6 subfilters
Simulation results (No. of bits)	0,2	0,2,*2*	0,0,2,3	0,0,1,2,*4*	0,0,1,2,2,3
Calculation results (No. of bits)	0,2	0,2,3	0,0,2,3	0,0,1,2,3	0,0,1,2,2,3

2.7.4 Design Example

To illustrate the proposed technique, the design of the linear phase FIR low-pass filter in Section 2.5 is considered. When the filter is designed with three subfilters using the method proposed in this section, the lengths of $H_1(z)$, $H_2(z)$ and $H_3(z)$ are 21, 19 and 15, respectively. The coefficients of $H_1(z)$ and $H_2(z)$ use average 2 terms of 4 bits SPoT and the coefficients of $H_3(z)$ use average 2 terms of 3 bits SPoT. The stopband attenuation is 40.47 dB. The frequency responses of three subfilters and the overall filter are shown in Fig. 2.8. The initial size of the population pool and mating pool are both 50. The lower and upper bounds are set to be 10 and 1000, respectively. Smaller population than 10 is usually used in a variation of the GA, i.e. micro-GA that works in a different way. Larger population than 1000 requires considerable computational cost. Therefore, the limitation of population size is chosen to be within 10-1000. Generation-Replacement combined with the elitist strategy is used as Replacement Strategies; and selection mechanism is Roulette Wheel Selection, which are used in all algorithms proposed in the thesis except

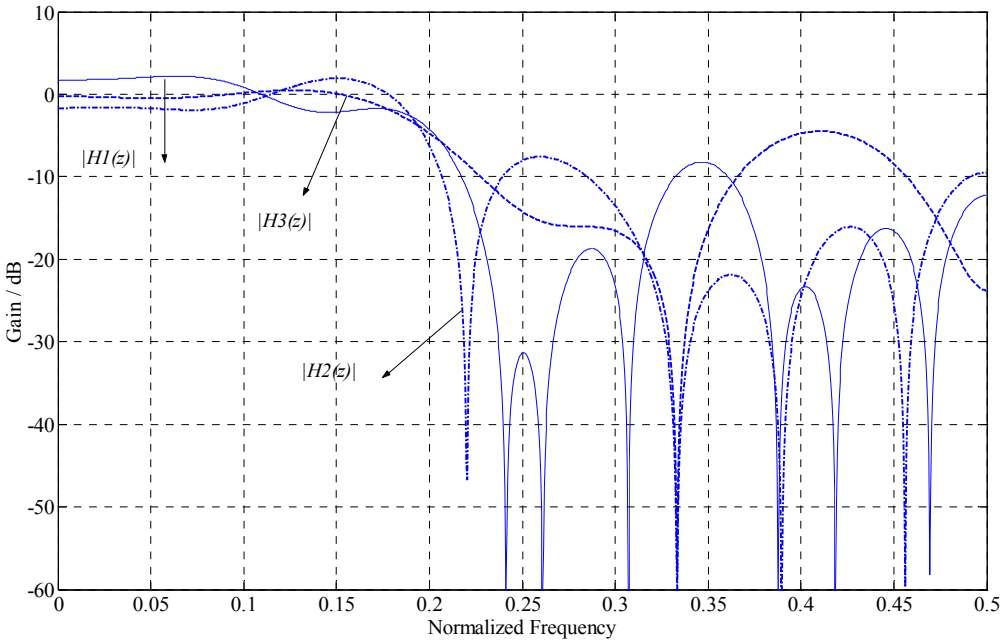
the MGA in Chapter 6 where the tournament selection strategy is used. The cycle of evolution is repeated until the desired objective value is obtained, i.e. 0.01 in this example, or the best fitness value of the population remains unchanged in 2000 generations. By using the proposed method, zero, two and two bits can be truncated from the output of $H_1(z)$, $H_2(z)$, and $H_3(z)$, respectively. This decreases the word length of the overall output and reduces the hardware requirement in implementation.

A comparison of hardware cost is given in Table 2.5. It can be seen from the table that the system designed by the GA with 3 subfilters uses 7.93% less D flip-flops and 36.30% less full adders than the one optimized using the iterative approach with 2 subfilters in [2]. Compared with the method in [2], the proposed method achieves about 30.49% savings in terms of the number of the transistors; compared with the method proposed in Section 2.5, 12.28% savings can be achieved.

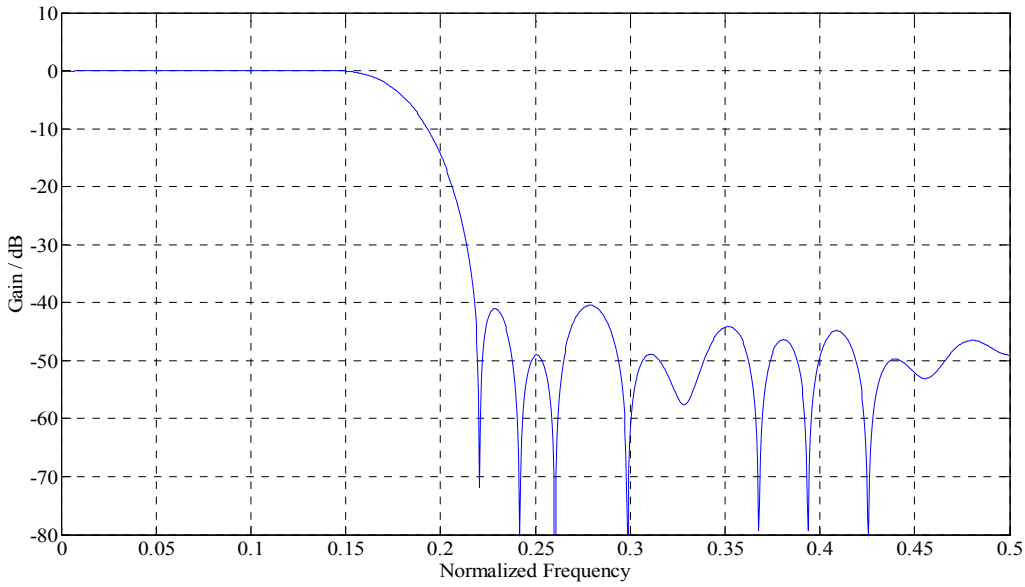
We also designed the filter with 2, 4, 5 and 6 subfilters. The frequency responses of these filters are shown in Figs. 2.9-2.12. Table 2.6 shows the comparisons of hardware cost in terms of the number of D-flip flops, 1-bit full adders and transistors for the designs of pre-truncation and post-truncation. It can be seen that the filter consisting of 3 cascaded subfilters requires the least hardware cost. The coefficients of the filters with 2, 3, 4, 5 and 6 subfilters are listed in Table 2.7.

2.8 Conclusion

In this chapter, the design schemes based on evolutionary optimization algorithms are presented for the design of high-speed low power FIR filters. The high-speed and low power features are achieved by factorizing a long filter into several cascaded subfilters each with coefficients constrained to sum or difference of power-of-two terms. The proposed method reduces the hardware cost significantly as the information which is related to hardware requirement is affiliated to the fitness function as an optimization criterion. To reduce the word length of the overall output, several of the least significant bits are truncated from the outputs of cascaded subfilters. An adaptive genetic algorithm (AGA) is proposed to improve the convergence performance of the GA, which adaptively adjusts the population size and the probabilities of crossover and mutation during the evolutionary process according to the optimization performance and population dynamics. Multiple trials to find suitable values for these parameters can be avoided. With the help of the proposed methods, considerable savings on hardware cost can be achieved.

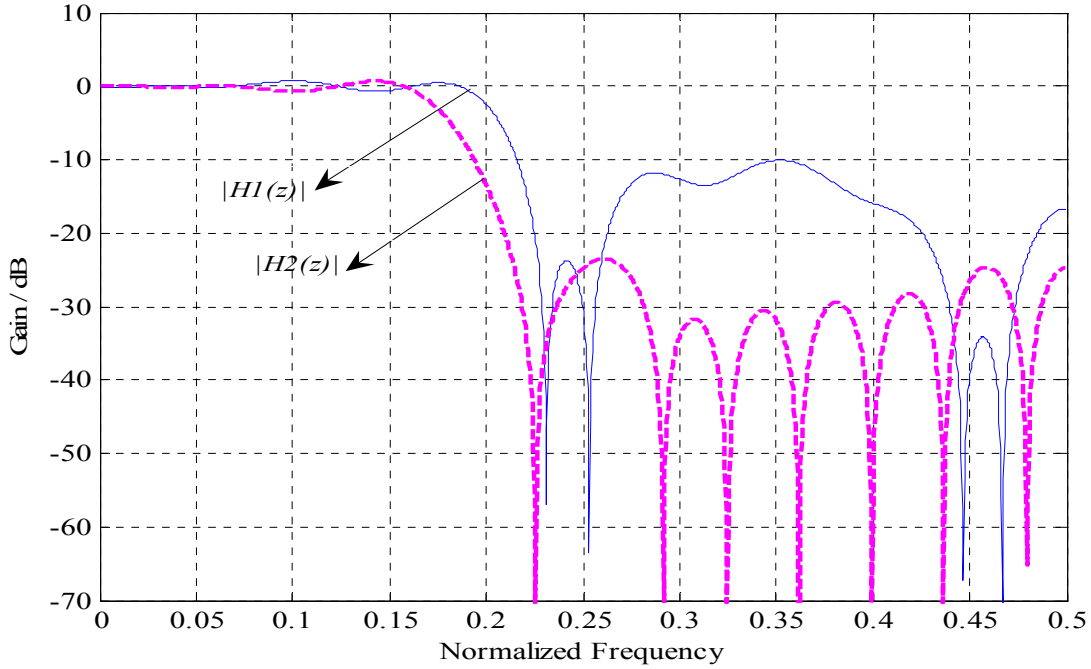


(a) Three subfilters $H_1(z)$, $H_2(z)$ and $H_3(z)$ whose orders are 20, 18 and 14, respectively.

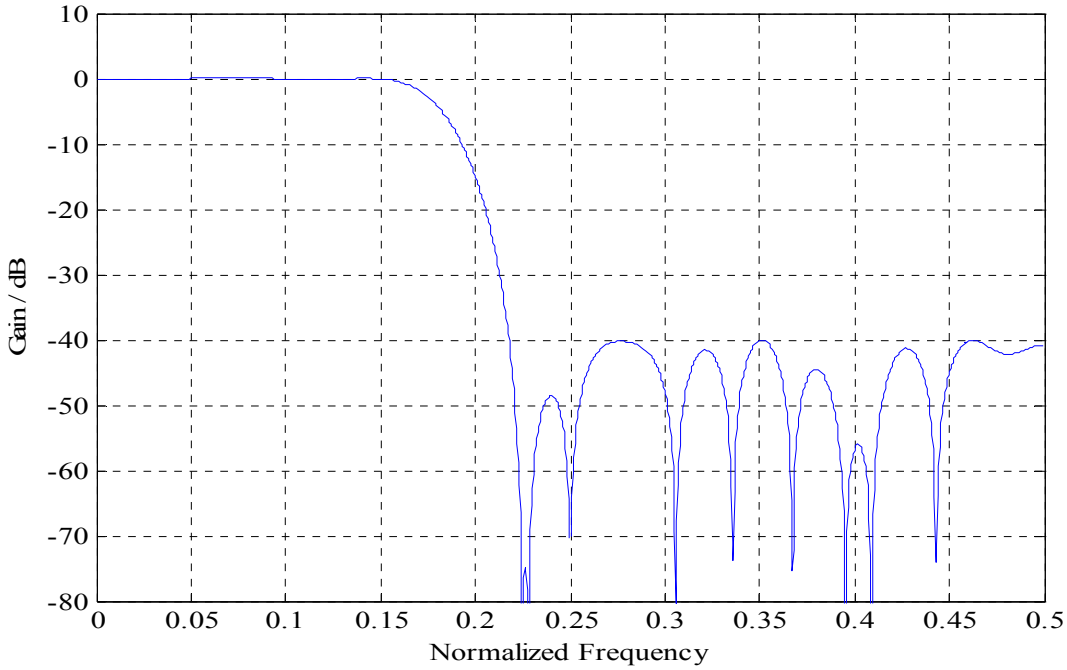


(b) The overall filter whose order is 52.

Fig. 2. 8 The frequency responses of the three subfilters (a) and overall filter (b).

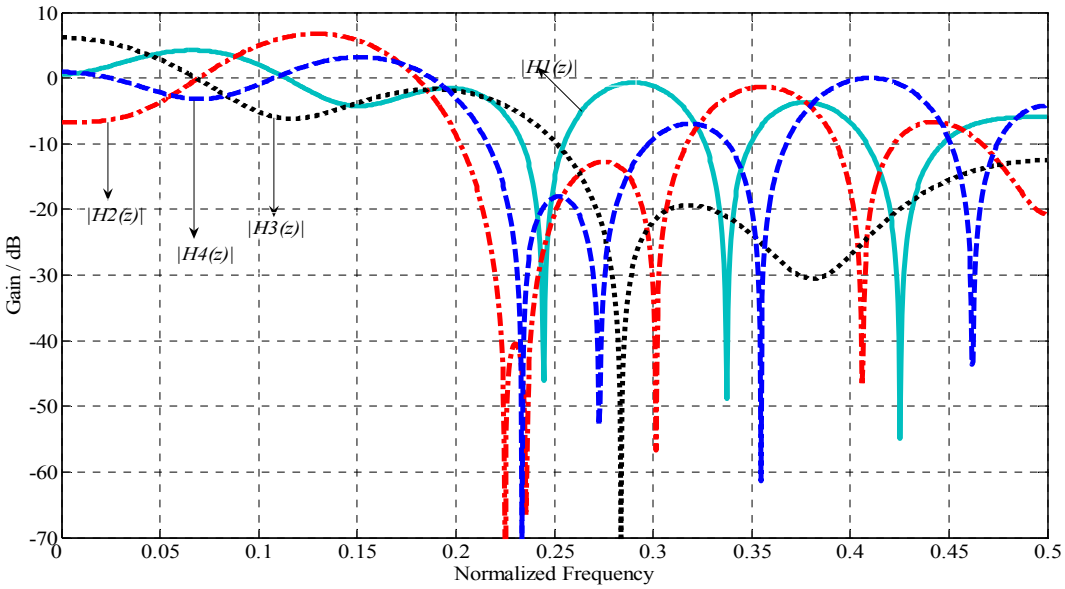


(a) Two subfilters $H_1(z)$ and $H_2(z)$ whose orders are both 26.

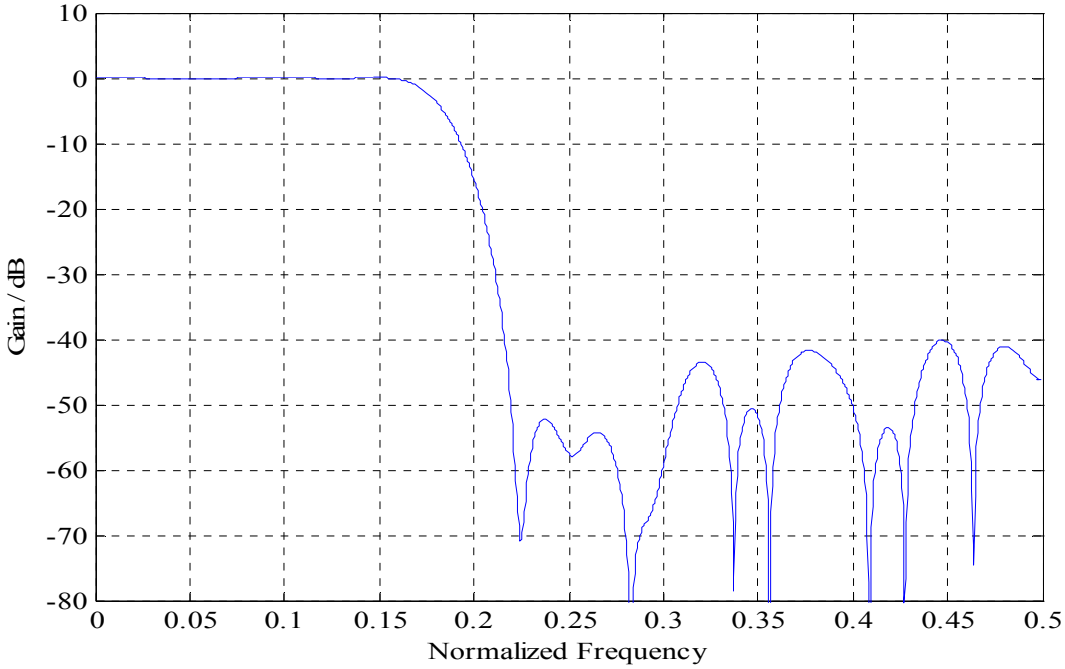


(b) The overall filter whose order is 52.

Fig. 2. 9 The frequency responses of the two subfilters (a) and overall filter (b).

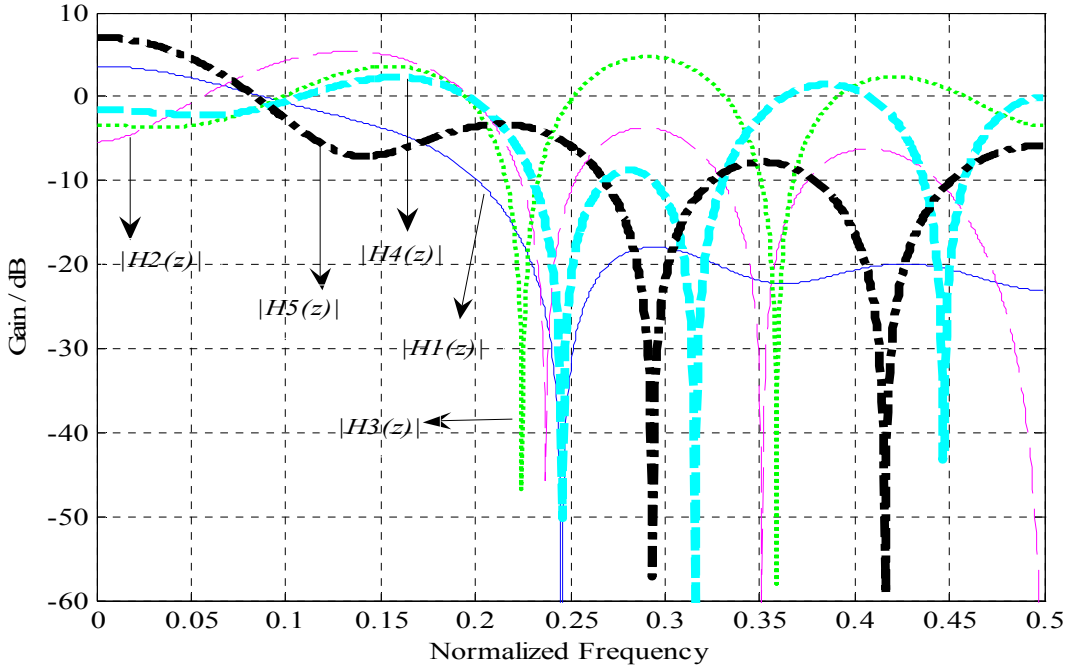


(a) Four subfilters $H_1(z)$, $H_2(z)$, $H_3(z)$ and $H_4(z)$ whose orders are 14, 14, 12 and 12, respectively.

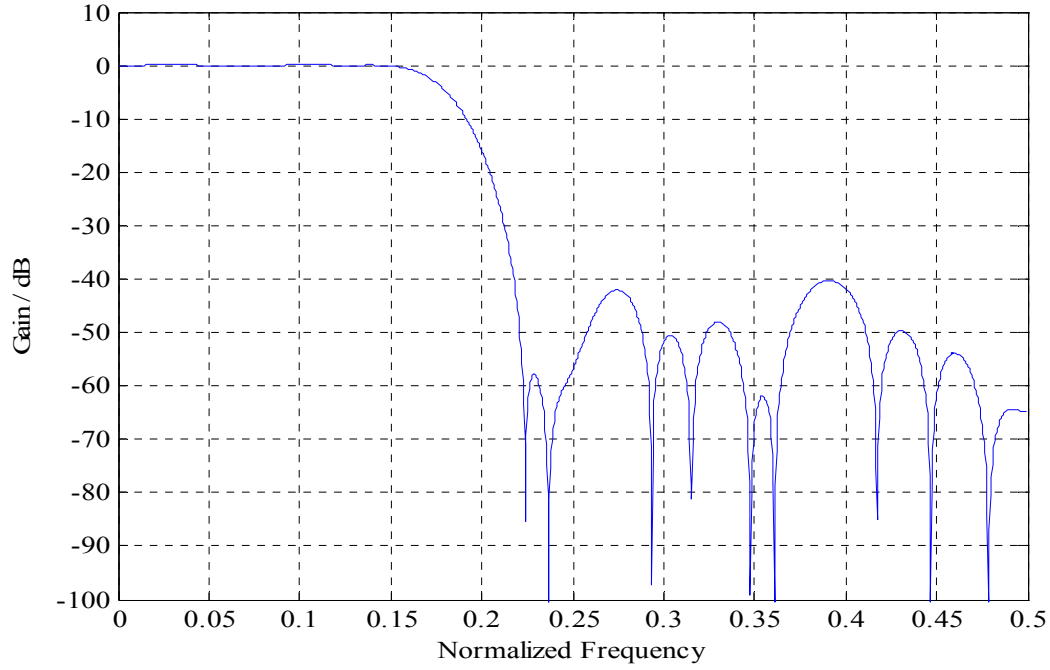


(b) The overall filter whose order is 52.

Fig. 2. 10 The frequency responses of the four subfilters (a) and overall filter (b).

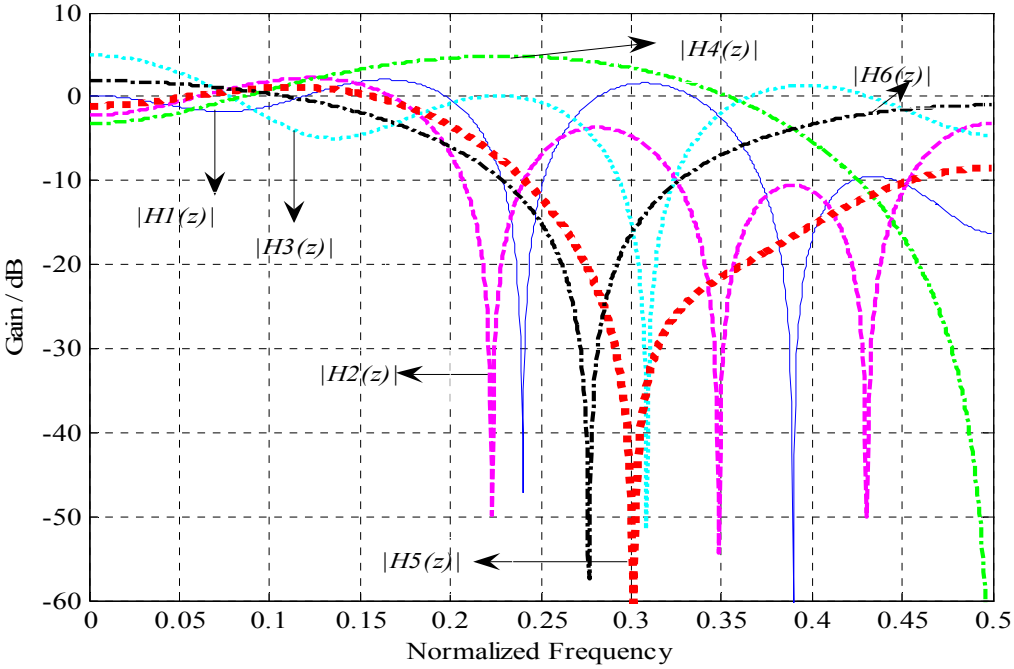


(a) Five subfilters $H_1(z)$, $H_2(z)$, $H_3(z)$, $H_4(z)$ and $H_5(z)$ whose orders are 12, 10, 10, 10 and 10, respectively.

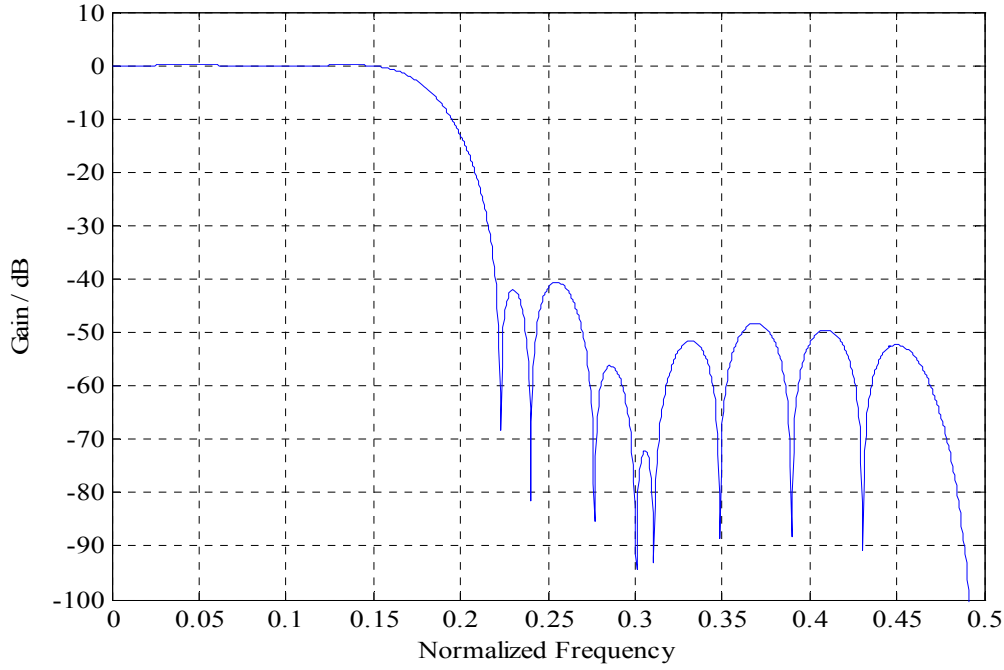


(b) The overall filter whose order is 52.

Fig. 2. 11 The frequency responses of the five subfilters (a) and overall filter (b).



(a) Six subfilters $H_1(z)$, $H_2(z)$, $H_3(z)$, $H_4(z)$, $H_5(z)$ and $H_6(z)$ whose orders are 10, 10, 8, 8, 8 and 8, respectively.



(b) The overall filter whose order is 52.

Fig. 2. 12 The frequency responses of the six subfilters (a) and overall filter (b).

Table 2. 5 A comparison of hardware cost among different designs

	Single filter $H(z)$	Iterative approach [2] $H_1(z)$ $H_2(z)$	Proposed in Section 2.5 $H_1(z)$ $H_2(z)$ $H_3(z)$	Proposed in Section 2.7 $H_1(z)$ $H_2(z)$ $H_3(z)$
Stopband attenuation (dB)	41.06	40.28	41.46	40.47
No. of bits	7	5,6	4,4,3	4,4,3
Total No. of SPoT	36	35	30	30
No. of D flip-flops	780	832	772	766
No. of 1-bit full adders	780	920	697	586
No. of transistors	28080	32416	25692	22536

Table 2. 6 A comparison of hardware cost between pre-truncation and post-truncation

No. of subfilters	2	3	4	5	6
No. of bits	5,5	4,4,3	3,4,4,2	4,4,2,3,3	4,4,3,3,3,2
Stopband ripple(dB)	-40.00	-40.47	-40.17	-40.15	-40.02
No. of D-flip flops for pre-truncation	806	794	844	934	1024
No. of D-flip flops for post-truncation	806	766	820	894	952
No. of 1-bit full-adders for pre-truncation	731	646	966	1136	997
No. of 1-bit full-adders for post-truncation	691	586	898	1033	921
No. of transistors for pre-truncation	26916	24440	33800	39280	36108
No. of transistors for post-truncation	25796	22536	31704	36076	33404

Table 2. 7 List of the coefficients of filters with 2 (a), 3 (b) and 4 (c), 5 (d) and 6 (e) subfilters

(a) Two subfilters

h_1	h_2
$h(0) = h(26) = 2^0$	$h(0) = h(26) = 2^0$
$h(1) = h(25) = 0$	$h(1) = h(25) = 0$
$h(2) = h(24) = -2^0$	$h(2) = h(24) = 0$
$h(3) = h(23) = -2^0$	$h(3) = h(23) = -2^0$
$h(4) = h(22) = 2^0$	$h(4) = h(22) = -2^0$
$h(5) = h(21) = 2^0$	$h(5) = h(21) = 0$
$h(6) = h(20) = 0$	$h(6) = h(20) = 2^1$
$h(7) = h(19) = -2^1$	$h(7) = h(19) = 2^1$
$h(8) = h(18) = 0$	$h(8) = h(18) = -2^0$
$h(9) = h(17) = 2^1+2^0$	$h(9) = h(17) = -2^2$
$h(10) = h(16) = 2^0$	$h(10) = h(16) = -2^1$
$h(11) = h(15) = -2^1$	$h(11) = h(15) = 2^2+2^1$
$h(12) = h(14) = -2^3-2^0$	$h(12) = h(14) = 2^4-2^0$
$h(13) = -2^4-2^0$	$h(13) = 2^4+2^2-2^0$

(b) Three subfilters

h_1	h_2	h_3
$h(0) = h(20) = 2^1$	$h(0) = h(18) = -2^0$	$h(0) = h(14) = 0$
$h(1) = h(19) = -2^1$	$h(1) = h(17) = 0$	$h(1) = h(13) = 2^0$
$h(2) = h(18) = -2^0$	$h(2) = h(16) = 0$	$h(2) = h(12) = -2^0$
$h(3) = h(17) = 0$	$h(3) = h(15) = 2^2$	$h(3) = h(11) = 0$
$h(4) = h(16) = 0$	$h(4) = h(14) = 0$	$h(4) = h(10) = -2^0$
$h(5) = h(15) = 0$	$h(5) = h(13) = -2^2$	$h(5) = h(9) = 0$
$h(6) = h(14) = 0$	$h(6) = h(12) = -2^2$	$h(6) = h(8) = 2^2+2^1$
$h(7) = h(13) = -2^1$	$h(7) = h(11) = 2^2$	$h(7) = 2^2+2^0$
$h(8) = h(12) = 2^2+2^0$	$h(8) = h(10) = 2^3+2^0$	
$h(9) = h(11) = 2^3+2^2+2^0$	$h(9) = 2^3+2^2-2^0$	
$h(10) = 2^3+2^2+2^1+2^0$		

(c) Four subfilters

h_1	h_2	h_3	h_4
$h(0) = h(14) = 2^2-2^0$	$h(0) = h(14) = 2^2$	$h(0) = h(12) = 2^0$	$h(0) = h(12) = -2^1$
$h(1) = h(13) = 2^0$	$h(1) = h(13) = -2^0$	$h(1) = h(11) = 2^2-2^0$	$h(1) = h(11) = 2^0$
$h(2) = h(12) = -2^2+2^0$	$h(2) = h(12) = -2^2+2^0$	$h(2) = h(10) = 2^1$	$h(2) = h(10) = 0$
$h(3) = h(11) = 2^2-2^0$	$h(3) = h(11) = -2^2+2^0$	$h(3) = h(9) = 2^0$	$h(3) = h(9) = 2^0$
$h(4) = h(10) = 0$	$h(4) = h(10) = -2^2-2^1-2^0$	$h(4) = h(8) = 2^1$	$h(4) = h(8) = 0$
$h(5) = h(9) = -2^2+2^0$	$h(5) = h(9) = 2^1$	$h(5) = h(7) = 2^3$	$h(5) = h(7) = -2^1-2^0$
$h(6) = h(8) = -2^2-2^1-2^0$	$h(6) = h(8) = 2^3$	$h(6) = 2^3+2^0$	$h(6) = -2^1-2^0$
$h(7) = -2^2-2^1-2^0$	$h(7) = 2^3+2^1$		

(d) Five subfilters

h_1	h_2	h_3
$h(0) = h(12) = 2^0$ $h(1) = h(11) = 2^0$ $h(2) = h(10) = 0$ $h(3) = h(9) = 2^0$ $h(4) = h(8) = 2^3 - 2^1$ $h(5) = h(7) = 2^3+2^2+2^1+2^0$ $h(6) = 2^3+2^2+2^1+2^0$	$h(0) = h(10) = 2^1$ $h(1) = h(9) = -2^2$ $h(2) = h(8) = -2^2$ $h(3) = h(7) = 2^1$ $h(4) = h(6) = 2^2+2^0$ $h(5) = 2^3$	$h(0) = h(10) = -2^1$ $h(1) = h(9) = 2^1$ $h(2) = h(8) = 2^1+2^0$ $h(3) = h(7) = -2^1-2^0$ $h(4) = h(6) = -2^0$ $h(5) = -2^1-2^0$

h_4	h_5
$h(0) = h(10) = 2^2-2^0$ $h(1) = h(9) = -2^2$ $h(2) = h(8) = 0$ $h(3) = h(7) = 0$ $h(4) = h(6) = 2^2 - 2^0$ $h(5) = 2^2+2^1+2^0$	$h(0) = h(10) = 2^0$ $h(1) = h(9) = 2^2-2^0$ $h(2) = h(8) = 0$ $h(3) = h(7) = 2^2-2^0$ $h(4) = h(6) = 2^2 + 2^1$ $h(5) = 2^2+2^1+2^0$

(e) Six subfilters

h_1	h_2	h_3
$h(0) = h(10) = 2^2+2^1$ $h(1) = h(9) = -2^1$ $h(2) = h(8) = -2^3-2^1$ $h(3) = h(7) = 2^3-2^0$ $h(4) = h(6) = 2^3+2^2-2^0$ $h(5) = 2^3+2^0$	$h(0) = h(10) = -2^1$ $h(1) = h(9) = 2^2+2^1$ $h(2) = h(8) = 2^1$ $h(3) = h(7) = -2^2+2^0$ $h(4) = h(6) = -2^3-2^0$ $h(5) = -2^3+2^0$	$h(0) = h(8) = 2^2-2^1$ $h(1) = h(7) = 0$ $h(2) = h(6) = -2^0$ $h(3) = h(5) = 2^2+2^1+2^0$ $h(4) = 2^2-2^0$

h_4	h_5	h_6
$h(0) = h(8) = 0$ $h(1) = h(7) = 0$ $h(2) = h(6) = -2^1$ $h(3) = h(5) = 2^0$ $h(4) = 2^2+2^1$	$h(0) = h(8) = -2^0$ $h(1) = h(7) = -2^0$ $h(2) = h(6) = 0$ $h(3) = h(5) = 2^2+2^1$ $h(4) = 2^2+2^1$	$h(0) = h(8) = 0$ $h(1) = h(7) = 0$ $h(2) = h(6) = 0$ $h(3) = h(5) = -2^1-2^0$ $h(4) = -2^0$

Chapter 3

Design of Frequency Response Masking Filter Using an Oscillation Search Genetic Algorithm

3.1 Introduction

The frequency-response masking (FRM) technique [36-48] is one of the most computationally efficient ways for the synthesis of arbitrary bandwidth sharp linear phase FIR digital filters. It utilizes an interpolated bandedge shaping filter to form the sharp transition-band of the overall filter. A pair of masking filters together with the complement of interpolated bandedge shaping filter construct the arbitrary bandwidth of the overall filter. The sparse coefficient vector of the bandedge shaping filter results great savings in arithmetic operations at the cost of longer group delay when synthesizes a very sharp FIR filter.

As discussed in the previous chapter, the coefficients of an FIR filter can be quantized into SPoT values leading to a so-called multiplication-free implementation [1]. However, the quantization process requires considerably large computer resources and takes very long time to search for a set of optimal SPoT coefficients, which is especially serious for

the design of FRM based FIR filter. To achieve the global optimal solutions, more than 3 subfilters have to be jointly designed. Recently, several nonlinear optimization techniques were introduced for the design of FRM filters [49-53]. These methods are able to jointly optimize all subfilters in a single stage FRM approach and produce considerable savings in terms of the number of arithmetic operations in comparison with traditional iterative design methods [36-38]. However, these methods cannot be directly applied in the design of FRM filters with discrete coefficients. In [54], a method based on iterative mixed integer programming (MILP) is proposed to design FRM filter with SPoT coefficients. It deals with the bandedge shaping filter and masking filters in two separate steps leading to a sub-optimum solution. In [13], the GA is applied in the design of FRM filters. However, since there are several subfilters to be jointly designed in the synthesis of FRM filters which makes the optimization very complicated with a large number of decision variables. The possibility of entrapment in local optima, therefore, is very high. When the GA is utilized in such complex applications, it usage is limited by the high probability of premature convergence as well as low convergence speed caused by searching large solution space. A sub-optimal solution is usually produced if there is no exterior strength to lead GA out of local optima.

To solve a complex optimization problem, the most effective way is to design a tailor-made algorithm that suits the needs of it. To this end, a hybrid GA is proposed in this chapter, which jointly designs all subfilters each with SPoT coefficients in the FRM structure. An algorithm named as oscillation search algorithm (OS) is developed, which is tailor-made for the design of digital filters in discrete space. The proposed hybrid

algorithm is formed by integrating the GA and OS, named as oscillation search genetic algorithm (OSGA). The whole optimization is based on the GA. The OS algorithm is utilized to optimize some fitter chromosomes during the evolutionary process of the GA. The application of the OS algorithm improves the convergence performance of the GA by preventing premature convergence and reducing computational complexity.

In the OSGA, instead of using the AGA presented in Chapter 2, the GA with fixed control parameters is employed. To adjust the population size and probabilities of genetic operations in the AGA, more function evaluations should be conducted. In the FRM filter design, the function evaluation involves the calculation of the frequency response of each subfilter and the synthesis of the overall filter. Since the transition band of the FRM filter is very sharp, it is necessary to use very fine grid of frequency points for the calculation of frequency response. Due to the complex structure and very fine grid used in the calculation, the computational cost for function evaluation is very high. Therefore, additional computation time to adjust control parameters is not preferred in FRM filter design.

It is shown by means of example that large savings in computation resources can be achieved by the proposed method. The coefficient word length in each subfilter has also been reduced significantly.

This chapter is organised as follows: The FRM technique is introduced in Section 3.2. The OSGA for the design and optimization of FRM filters in SPoT space is discussed in Section 3.3. Section 3.4 is dedicated to the design example. A summary is given in Section 3.5.

3.2 Frequency-Response Masking Technique

The FRM technique employs an interpolated bandedge shaping filter to form the sharp transition width for an FIR filter and utilizes the delay-complementary concept to realize arbitrary bandwidth. Fig. 3.1 shows one of the possible realization structures for FRM. In this figure, $H_a(z)$ is referred as bandedge shaping filter which is a symmetrical impulse response linear phase low-pass filter with odd length N_a and its complementary filter $H_c(z)$ can be expressed as

$$H_c(z) = z^{-\frac{N_a-1}{2}} - H_a(z). \tag{3.1}$$

The frequency response of $H_a(z)$, is shown in Fig. 3.2(a). A pair of complementary interpolated bandedge shaping filters $H_a(z^M)$ and $H_c(z^M)$ can be derived by replacing each delay element of both $H_a(z)$ and $H_c(z)$ by M delays. The frequency responses of $H_a(z^M)$ and $H_c(z^M)$ are shown in Fig. 3.2(b). The transition-band widths of $H_a(z^M)$ and $H_c(z^M)$ are a factor of M narrower than that of $H_a(z)$. Two masking filters $H_{Ma}(z)$ and $H_{Mc}(z)$, whose frequency responses are shown in Figs.3.2(c) and 3.2(e), are cascaded to $H_a(z^M)$ and $H_c(z^M)$, as shown in Fig. 3.1, to remove periodic repetitions of $H_a(z^M)$ and $H_c(z^M)$ in the stopband, respectively. The outputs of $H_a(z^M)H_{Ma}(z)$ and $H_c(z^M)H_{Mc}(z$) are summed to form the overall system, denoted by $H(z)$, as shown in Figs. 3.2(d) and 3.2(f). The z-transform transfer function of the overall filter is given by

$$H(z) = H_a(z^M)H_{Ma}(z) + H_c(z^M)H_{Mc}(z). \tag{3.2}$$

Note that the lengths of $H_{Ma}(z)$ and $H_{Mc}(z)$ in (3.2) are assumed to be the same. If they are not, zero coefficients should be padded into the short filter. The joint design of all subfilters with SPoT coefficients is a nonlinear process since the objective function based on EQ. (3.2) is nonlinear.

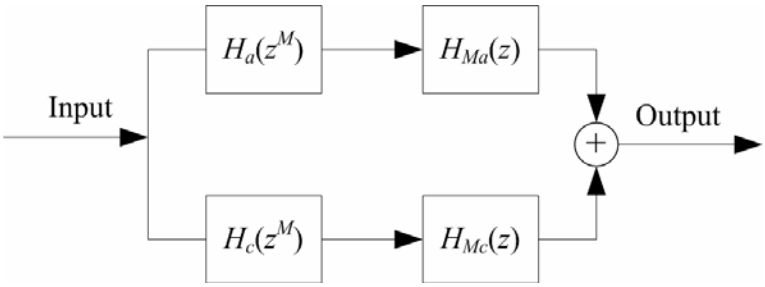


Fig. 3. 1 A realization structure for FRM approach.

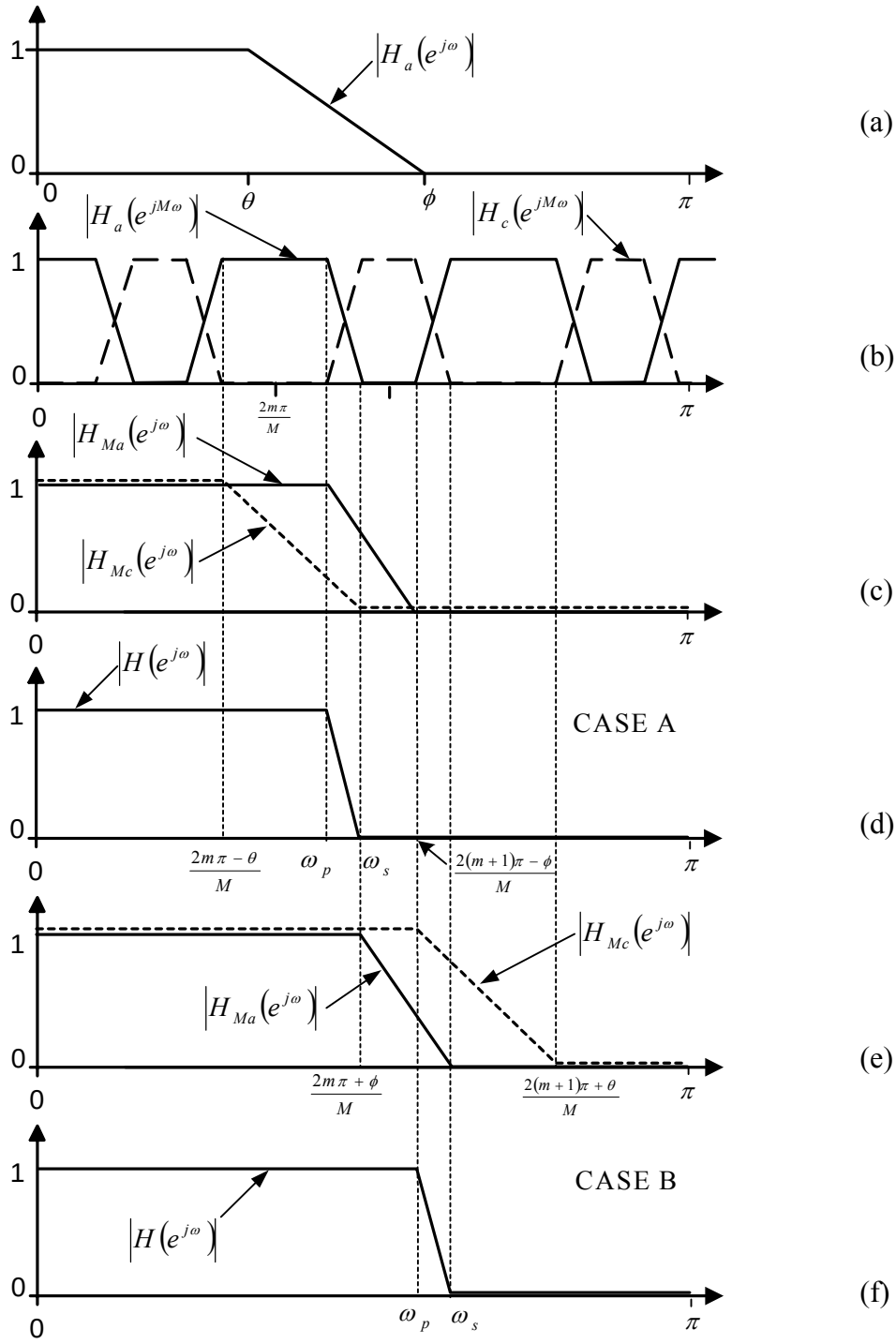


Fig. 3. 2 The frequency responses of various subfilters in the FRM technique.

3.3 Oscillation Search Genetic Algorithm (OSGA)

As a stochastic search method, the GA finds an optima solution through evolution without incorporating any rules of the problem to be optimized and the whole process is problem-independent. This is a great advantage and also a disadvantage of the GA. The GA does not always evolve towards a good solution; it only evolves away from bad circumstances. Therefore, the GA risks finding a suboptimal solution and has low convergence speed in complex applications. It is possible to boost GA's performance and improve its convergence speed if some properties of our problem can be taken into consideration. Three properties of filter coefficients are considered. First, although direct rounding of real valued coefficients deteriorates the frequency response, they usually have the same sign as the optimal discrete coefficients. Second, it is well known that the centre coefficient has larger absolute value and the coefficients near to the beginning and end are smaller. For example, if 10 bits are used to represent the filter coefficients, it is impossible for the first coefficient value $h(0)$ to be about 1023, while it is also impossible for the centre coefficient to be less than 100 in most situations. Third, in a symmetrical filter the frequency response is usually more sensitive to the approximation of the coefficients located near the centre. Little change of these coefficients can largely influence the frequency response of the filter, while similar changes of those coefficients near to the beginning and end can only slightly influence its frequency response. Therefore, it is reasonable to optimize the coefficients of a filter from high coefficient sensitivity to low sensitivity. An oscillation search algorithm is developed based on the

above properties, whose basic principle is to find better coefficient values by oscillating around the initial values. In the oscillation search scheme, each subfilter in FRM structure is optimized individually. Each time the algorithm only optimizes one coefficient of one subfilter and moves to the next coefficient upon the completion of the optimization for the current one. However, globally optimal solutions are possibly achieved if all variables are jointly optimized. The OS algorithm that optimizes each coefficient individually difficultly finds global optimal solutions. To address this, it is integrated with the GA to generate a hybrid algorithm, OSGA, where the GA is used as the basis of the global optimization and OS is applied to optimize elitist members during the evolutionary process. The detailed information on the implementation is given in the following sub-sections.

3.3.1 The Implementation of GA

In the OSGA, the GA is employed in the whole optimization process. The coefficients of the subfilters in the FRM structure are encoded and concatenated to represent the chromosomes as a string of ternary encoding digits. By this way, the coefficients of all the subfilters can be jointly optimized through the evolutionary process of the GA.

The main objective of the optimization in the design of FRM filters is to find the coefficients of each subfilter which can minimize the peek ripple in all bands. Hence, the objective function, O, can be defined as the same as that in EQ. (2.8), where $H(e^{j\omega})$ and $H_d(e^{j\omega})$ are the frequency responses of the $H(z)$ in (3.2) and desired filter, respectively. Besides the minimization of the overall ripple, the hardware cost is also to be minimized

by decreasing the number of bits and the number of SPoT terms and increasing the number of zero-value coefficients. The fitness function is defined as

$$f = \frac{1}{O} + \frac{a_1}{S_T} + a_2 \times Z_T + \frac{a_3}{Diff_{\max}}, \tag{3.3}$$

where the definitions of S_T and Z_T are the same as those in EQ. (2.17) and $Diff_{\max}$ is expressed as

$$\begin{aligned} &Diff_{\max} = \max(Diff_a, Diff_{Ma}, Diff_{Mc}), \\ &\text{where } Diff_P = \max|h_P| - \min|h_P|. \end{aligned} \tag{3.4}$$

Here, h_p represents the coefficients of the p^{th} subfilters, where p can be a, Ma or Mc; in other words, h_p denotes the coefficients of the bandedge shaping filter or two masking filters. The three positive weighting coefficients are defined as

$$a_1 = 0.1S_S,\ a_2 = \frac{5}{N_T},\ a_3 = 2, \tag{3.5}$$

where S_S is the pre-specified maximal number of SPoT terms and N_T is the total length of all subfilters.

3.3.2 Oscillation Search (OS) Algorithm

The basic principle of the OS algorithm is to find better coefficient values by oscillating around the initial values. In the OS scheme, all subfilters in FRM structure are optimized individually. Each time the algorithm optimizes only one coefficient of one subfilter and moves to the next coefficient after the optimization for the current one is finished. The coefficient to be optimized is gradually increased until the maximal bound is reached and then decreased until the minimal bound is reached. The optimal value of the coefficient is the one with the largest objective value during the increasing and decreasing processes,

which will be used in the subsequent optimization process. In a subfilter, the optimization order is from high coefficient sensitivity to low sensitivity. An optimization round is defined as the process during which all subfilters are optimized once. If an improvement of the objective value can be obtained in the current round, a new one will be started upon the completion of the current one. The OS algorithm is considered to reach convergence if there is no improvement during a whole round.

During the evolutionary process of the GA, the OS algorithm is applied when one of the two conditions is met:

a) For every a pre-specified number (G_{OS}) of generations, the OS will join according to a probability (p_{os});

b) If the fitness improvement cannot be obtained in a specific number of generations.

Here, p_{os} controls the involvement of the OS algorithm, which is introduced to avoid disturbing the GA evolution. If p_{os} is chosen to be 1, the OS algorithm will be applied after every G_{OS} generations. This may disturb the GA evolution. We can avoid this by choosing a larger value of G_{OS}. However, the fixed application frequency of the OS algorithm leads to the loss of flexibility. The use of p_o, offers more flexibility to the search. The setting of p_{os} is related to the choice of G_{OS}. It is usually chosen within 0.5-0.8 when G_{OS} is within 100-500.

The objective function of the OS algorithm is formulated as

$$O_{OS} = -20\log_{10}(O). \tag{3.6}$$

In the OS algorithm, better solutions have larger objective values. According to the definition of the O, smaller O means smaller peak ripple, and also better design. Thus, we

need an inverse form of O to express its objective function. Here, we define it as EQ. (3.6) because such definition can directly tell us the peak ripple level in dB measurement.

The three subfilters $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ in FRM structure can be characterized by the z-transform transfer function of EQ. (2.1). For symmetrical filters, we just need to optimize half of N or (N+1) coefficients, i.e. $h(n)$, $n = 0,\ldots,(N\text{-}1)/2$ for odd length filters or $h(n)$, $n = 0,\ldots,(N\text{-}2)/2$ for even length filters. For each subfilter, the coefficients are optimized one by one from high coefficient sensitivity to low sensitivity. The optimization is performed from $h((N\text{-}1)/2)$ for odd length filter or $h((N\text{-}2)/2)$ for even length filter to $h(0)$.

The steps are shown as follows:

1) Choose a "fitter" chromosome from the GA population.
2) Calculate its OS objective value using EQ. (3.6).
3) Optimize one coefficient of $H_a(z)$ at a time from $h((N\text{-}1)/2)$ to $h(0)$. Assume that the coefficient to be optimized is denoted as $h(i)$, where $i = (N\text{-}1)/2, \ldots, 1, 0$. Increase the value of $h(i)$ by 1 and calculate the OS objective value. If the new objective value is larger than the best one found so far, the new value is recorded as the best one and the set of coefficient values of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are used in the subsequent optimization. The search step is set to be 1, which avoids missing better solutions. Repeat the process until the maximum bound of $h(i)$ is reached. In parallel, the initial value of $h(i)$ is also decreased by 1 at each time until the minimal bound is reached to check if better solutions can be found. We define the maximal and the minimal bounds of the OS search as

$$h_{\max}=\begin{cases}\min[h_{org}(i)+2^{b},2^{B}-1] & \text{for } h_{org}(i)\geq 0\\ \min[h_{org}(i)+2^{b},0] & \text{for } h_{org}(i)<0\end{cases},$$
$$h_{\min}=\begin{cases}\max[h_{org}(i)-2^{b},0] & \text{for } h_{org}(i)\geq 0\\ \max[h_{org}(i)-2^{b},1-2^{B}] & \text{for } h_{org}(i)<0\end{cases}, \quad (3.7)$$

where $h_{org}(i)$ is the initial value of $h(i)$, B is the word length of the coefficient, and b is used to control the search space of the OS. Working in the way of local search, the OS algorithm searches the neighborhood area of $h_{org}(i)$. The maximal and minimal bounds are employed to limit the search. Enlarging the search space by increasing the value of b requires more computation time for the OS process. However, small search space associated with a small value of b may make the OS miss better solutions. The suitable choice of b is based on the consideration of trade-off between computation time and and solution quality. In our implementation, b is set to be 3. The search space is within $h_{org}(i)\pm 2^3$ or the permitted maximal/minimal values.

After all coefficients $h(i)$ in $H_a(z)$, where i = 0,…,(N-1)/2, are optimized, go to the next step.

4) Repeat Step 3 for the optimization of $H_{Ma}(z)$ and $H_{Ma}(z)$.

5) If there is no improvement in an optimization round, the search terminates and goes to Step 6 to record data; otherwise a new optimization round will be started by going to Step 3.

6) Record coefficient values of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ which have the best OS objective value found through the above steps. Encode them to form a string of ternary digits and construct a chromosome of GA. Replace the worst chromosome in the population with this one.

Through Step 1-6, it is possible to improve the quality of the GA population by refining better individuals and replacing worse ones. By this way, it is likely to find a global optimal solution. Furthermore, the OS algorithm can be used to improve the initial population obtained either by rounding the real valued coefficients or by randomly generating. Since it is developed according to the properties of filter coefficients, it can reduce the computational cost required by purely stochastic search of the GA.

3.4 Design Example

To illustrate the proposed technique, let us consider the design of a narrow transition width low-pass filter that is also taken as an example in [13]. The filter meets the following specifications: the normalized passband and stopband edges are 0.3 and 0.305, respectively; the permitted maximum passband and stopband deviations are 0.01. The estimated value of M corresponding to the lowest complexity is 6, and the lengths of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ with infinite coefficients are 67, 19 and 33, respectively [36]. To meet the specifications, the filter lengths of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ have to be increased to be 69, 21 and 35, respectively, when each subfilter is quantized with 12 bits SPoT coefficients using linear programming [36]. The numbers of SPoT terms in $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are 92, 29 and 57, respectively. The stopband attenuation of the overall filter is 40.64 dB.

If the filter is designed using the OSGA, the probabilities of crossover and mutation are set to be 0.7 and 0.01, respectively. The OS algorithm is applied to the top 5%

individuals in the population for every 200 generations with a probability of 0.6 or when there is no fitness improvement for 300 generations. The cycle of evolution is repeated until the desired objective value is obtained, i.e. 0.01 in this example, or the fitness improvement cannot be achieved for 2000 generations. The lengths of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are 67, 19 and 33, respectively, which are the same as those with real valued coefficients. $H_a(z)$ uses 10 bits SPoT and $H_{Ma}(z)$ and $H_{Mc}(z)$ both use 11 bits SPoT. The stopband attenuation of the overall filter is 40.21 dB. The total number of SPoT terms used in $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are 75, 32 and 50, respectively. The frequency responses of $H_a(z^6)$, $H_{Ma}(z)$, $H_{Mc}(z)$ and the overall filter are shown in Figs. 3.3 - 3.5, respectively. It is clear that proposed algorithm can effectively reduce the coefficient word length. The coefficient values of the subfilters are listed in Table 3.1.

To evaluate the performance of each chromosome in the population, the fitness function has to be evaluated for ($P \times G$) times during the whole optimization process, where P is the population size and G is the total number of generations. The calculation of the frequency response for all subfilters and the synthesis of the overall filter are involved in the objective function. Due to the sharp transition band, very fine grid of frequency points is required in calculation, which increases the complexity of function evaluation. To partially solve the problem, a small population is preferred to reduce the computational cost. However, in the GA, a small population usually leads to high probability of premature convergence, especially when the solution space is large. Therefore, in [13], a big population pool and mating pool are employed in the design of the above filter, namely, 4000 and 800, respectively. With the same lengths of $H_a(z)$,

$H_{Ma}(z)$ and $H_{Mc}(z)$, the evolutionary process evolved to 38 dB after 4287 generations, where all coefficients are represented by average 2 terms of 11 bits SPoT. With the help of the OSGA, the population pool size and mating pool size can be reduced to 200. We achieve 38.12 dB after 1505 generations and the coefficients of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are represented by 8, 9 and 9 bits SPoT, respectively. The frequency responses of $H_a(z^6)$, $H_{Ma}(z)$, $H_{Mc}(z)$ and the overall filter are shown in Figs. 3.6 - 3.8. Table 3.2 shows the comparisons among the designs from different methods, i.e. the method in [36], GA [13] and proposed OSGA. The coefficient values of the subfilters are listed in Table 3.3. Compared with the results in [13], the following improvements can be claimed:

1) Require much fewer generations.
2) Use much smaller population. Fewer generations coupled with smaller population size mean less computational cost.
3) Reduce the coefficient wordlengths for all subfilters.

Although the number of bits and computation time are largely reduced, the total number of SPoT terms in each subfilters is slightly more than the one using average 2-term by 2, 3 and 5 terms, respectively. The increase is mainly caused by the OS that searches the neighborhood by increasing or decreasing 1 in each step. If minimizing the coefficient wordlength is a major goal, our method is most effective. The excellent performance of the OSGA makes it promising to solve other integer problems.

3. 5 Conclusion

In this chapter, a hybrid optimization scheme is proposed for the design of FRM filters in a discrete space. The coefficients of the subfilters are quantized to signed power-of-two (SPoT) values in order to replace multiplications with a limited number of shift-and-add operations. The hybrid algorithm (OSGA) is generated by integrating the genetic algorithm (GA) with an oscillation search (OS) algorithm. The optimization process is based on the GA. The OS that is developed based on the properties of filter coefficients is used as a separate process to refine the elitist individuals achieved from the GA. All subfilters are simultaneously designed each with SPoT coefficients. It has shown that the proposed OSGA is both more efficient and more effective than the GA by requiring less computational cost and identifying higher quality solutions.

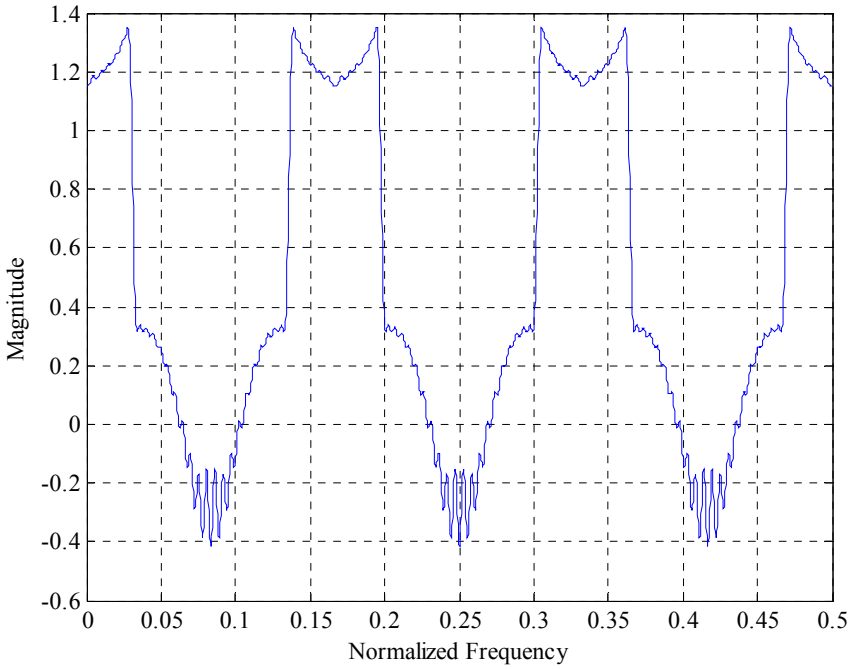


Fig. 3. 3 The frequency response of $H_a(z^6)$ with the overall filter stopband attenuation of 40.21 dB.

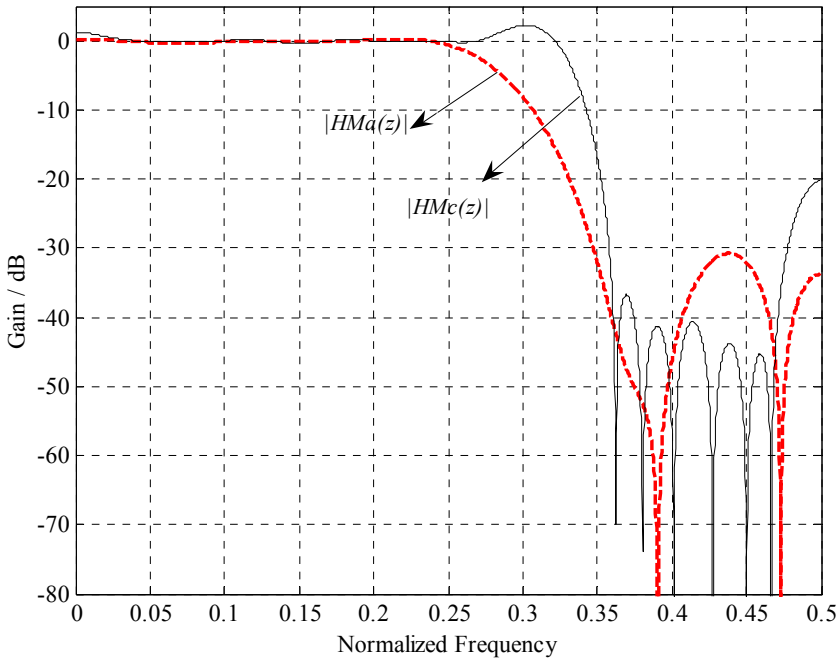


Fig. 3. 4 The frequency responses of $H_{Ma}(z)$ and $H_{Mc}(z)$ with the overall filter stopband attenuation of 40.21 dB.

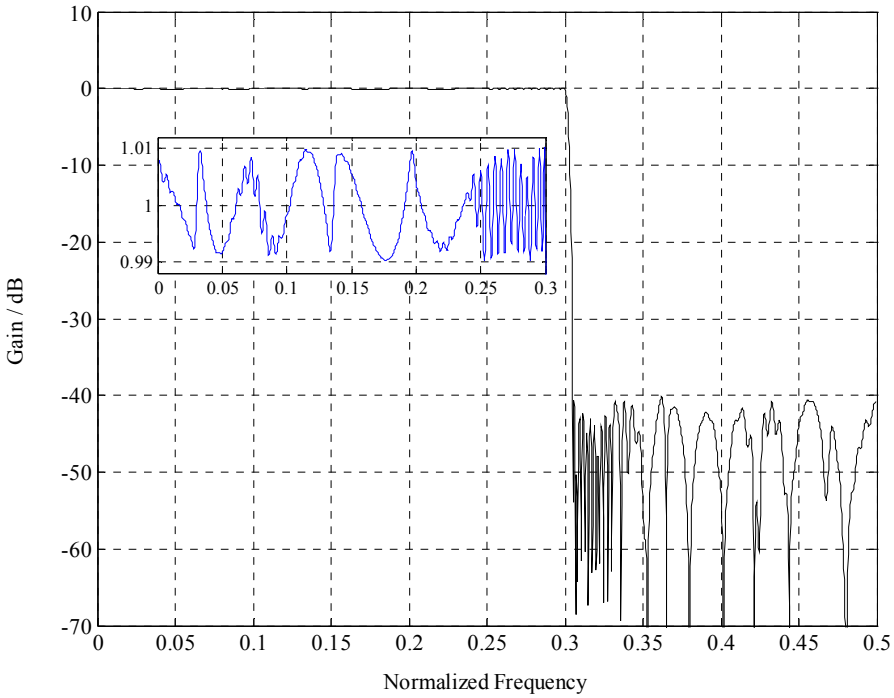


Fig. 3. 5 The frequency response of the overall filter with the overall filter stopband attenuation of 40.21 dB.

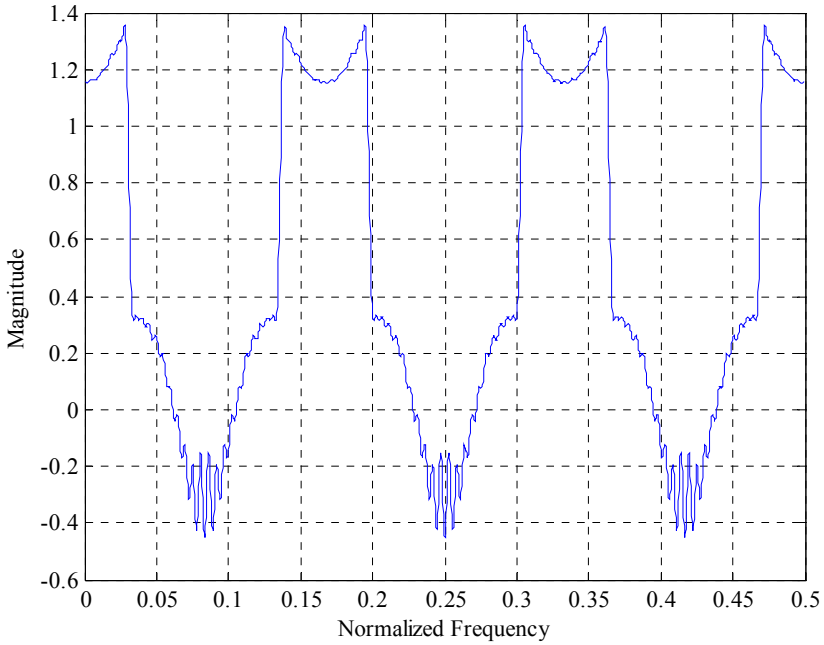


Fig. 3. 6 The frequency response of $H_a(z^6)$ with the overall filter stopband attenuation of 38.12 dB.

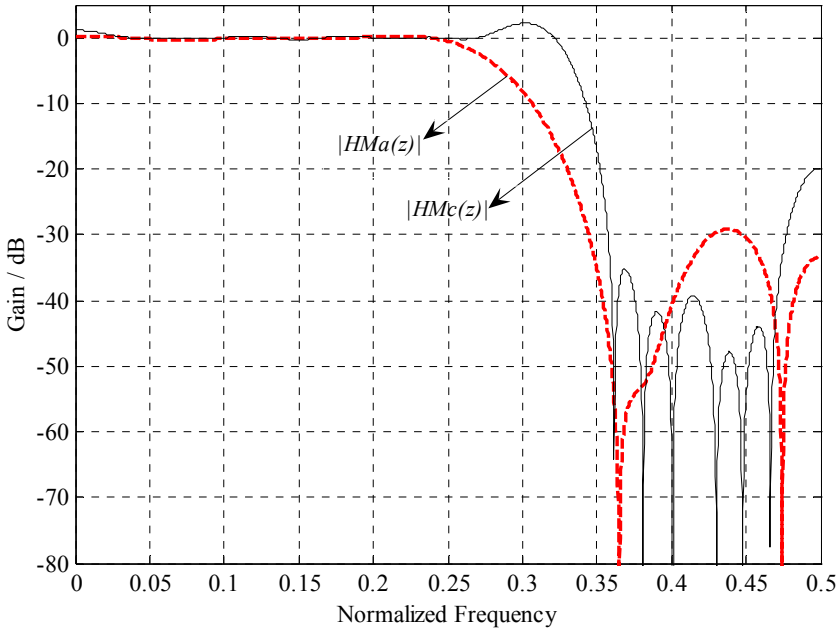


Fig. 3. 7 The frequency responses of $H_{Ma}(z)$ and $H_{Mc}(z)$ with the overall filter stopband attenuation of 38.12 dB.

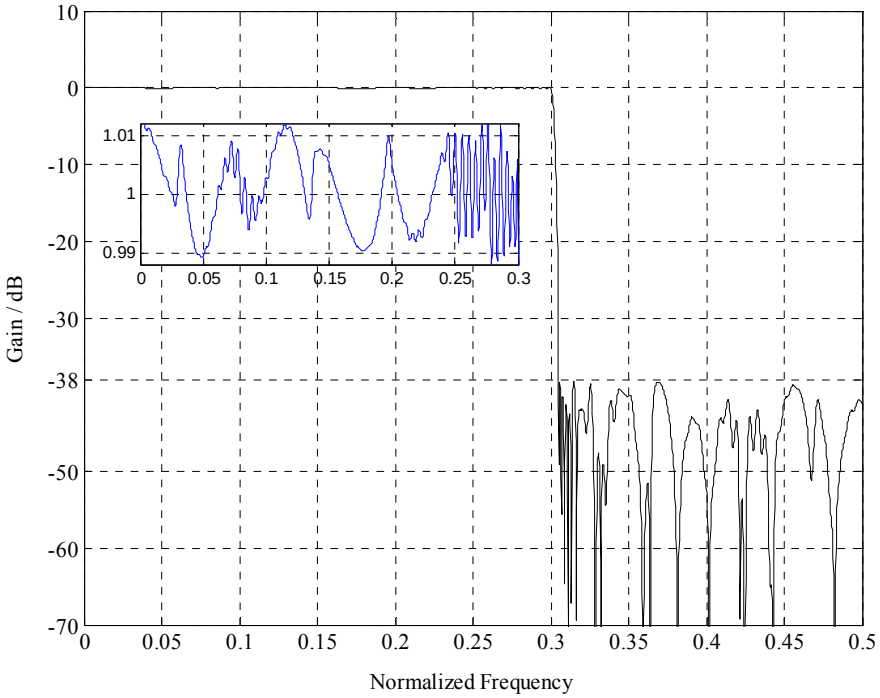


Fig. 3. 8 The frequency response of the overall filter with the overall filter stopband attenuation of 38.12 dB.

Table 3. 1 List of filter coefficients of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ with the overall filter stopband attenuation of 40.21 dB

$h_a(n)$		
$h(0) = h(66) = 2^3+2^2-2^0$ $h(1) = h(65) = -2^4+2^2$ $h(2) = h(64) = 2^3-2^0$ $h(3) = h(63) = -2^3-2^2+2^0$ $h(4) = h(62) = 2^3+2^1$ $h(5) = h(61) = -2^0$ $h(6) = h(60) = 2^2$ $h(7) = h(59) = -2^3-2^0$ $h(8) = h(58) = -2^2+2^0$ $h(9) = h(57) = 0$ $h(10) = h(56) = 2^3+2^0$ $h(11) = h(55) = 2^2-2^0$	$h(12) = h(54) = -2^3+2^1$ $h(13) = h(53) = -2^4+2^2$ $h(14) = h(52) = -2^1$ $h(15) = h(51) = 2^4-2^2-2^0$ $h(16) = h(50) = 2^4-2^2$ $h(17) = h(49) = -2^2$ $h(18) = h(48) = -2^4-2^1$ $h(19) = h(47) = -2^4+2^2$ $h(20) = h(46) = -2^4+2^2-2^0$ $h(21) = h(45) = 2^5-2^3+2^0$ $h(22) = h(44) = 2^3-2^1$ $h(23) = h(43) = -2^5+2^3+2^0$	$h(24) = h(42) = -2^5-2^1$ $h(25) = h(41) = 2^2$ $h(26) = h(40) = 2^5+2^3+2^0$ $h(27) = h(39) = 2^5+2^3-2^0$ $h(28) = h(38) = -2^5+2^2$ $h(29) = h(37) = -2^6-2^4-2^2-2^0$ $h(30) = h(36) = -2^5+2^3-2^0$ $h(31) = h(35) = 2^4+2^0$ $h(32) = h(34) = 2^9-2^7+2^4+2^2+2^1$ $h(33) = 2^9+2^2$
$h_{Ma}(n)$		
$h(0) = h(18) = -2^3+2^1$ $h(1) = h(17) = 2^5+2^1+2^0$ $h(2) = h(16) = 2^3-2^0$ $h(3) = h(15) = -2^6+2^4$	$h(4) = h(14) = 2^5-2^2$ $h(5) = h(13) = 2^7-2^3-2^2+2^0$ $h(6) = h(12) = -2^7-2^2+2^0$	$h(7) = h(11) = -2^7-2^5-2^2+2^0$ $h(8) = h(10) = 2^9+2^7-2^5+2^3+2^0$ $h(9) = 2^{10}+2^7+2^5+2^3-2^1$
$h_{Mc}(n)$		
$h(0) = h(32) = 2^5$ $h(1) = h(31) = -2^4+2^2+2^0$ $h(2) = h(30) = -2^4-2^2+2^0$ $h(3) = h(29) = 2^5+2^3-2^0$ $h(4) = h(28) = -2^1$ $h(5) = h(27) = -2^6+2^4$	$h(6) = h(26) = 2^6+2^4+2^2+2^0$ $h(7) = h(25) = 2^3+2^1$ $h(8) = h(24) = -2^6-2^4-2^0$ $h(9) = h(23) = 2^6+2^5-2^2$ $h(10) = h(22) = 2^5+2^3+2^2-2^0$ $h(11) = h(21) = -2^7-2^4-2^0$	$h(12) = h(20) = 2^7+2^5-2^1$ $h(13) = h(19) = 2^6-2^3-2^0$ $h(14) = h(18) = -2^8-2^5-2^3+2^0$ $h(15) = h(17) = 2^9+2^5+2^1$ $h(16) = 2^{10}+2^9-2^7+2^4+2^3$

Table 3. 2 A comparison among different designs from the method in [36], GA [13] and OSGA

(a) Filter specifications

	Ripple	Filter length	No. of bits	No. of term
Method [36]	38.12 dB	67, 21, 35	11, 11, 11	72, 27, 44
GA [13]	38 dB	67, 19, 33	11, 11, 11	Avg. 2
OSGA	38.12	67, 19, 33	8, 9, 9	70, 23, 39

(b) GA parameters

	Population pool size	Mating pool size	Generations
GA[13]	4000	800	4287
OSGA	200	200	1505

Table 3. 3 List of filter coefficients of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ with the overall filter stopband attenuation of 38.12 dB

$h_a(n)$		
$h(0) = h(66) = 2^2+2^0$ $h(1) = h(65) = -2^3+2^0$ $h(2) = h(64) = 2^2$ $h(3) = h(63) = -2^2-2^0$ $h(4) = h(62) = 2^2+2^1$ $h(5) = h(61) = -2^0$ $h(6) = h(60) = 2^2-2^0$ $h(7) = h(59) = -2^2-2^0$ $h(8) = h(58) = -2^0$ $h(9) = h(57) = 0$ $h(10) = h(56) = 2^2+2^0$ $h(11) = h(55) = 2^0$	$h(12) = h(54) = -2^2+2^0$ $h(13) = h(53) = -2^2-2^1$ $h(14) = h(52) = 0$ $h(15) = h(51) = 2^2+2^1$ $h(16) = h(50) = 2^2+2^1$ $h(17) = h(49) = -2^1$ $h(18) = h(48) = -2^3-2^0$ $h(19) = h(47) = -2^2-2^1$ $h(20) = h(46) = 2^2+2^1$ $h(21) = h(45) = 2^3+2^2+2^0$ $h(22) = h(44) = 2^1$ $h(23) = h(43) = -2^3-2^2+2^0$	$h(24) = h(42) = -2^4$ $h(25) = h(41) = 2^2-2^0$ $h(26) = h(40) = 2^4+2^2+2^0$ $h(27) = h(39) = 2^4+2^2$ $h(28) = h(38) = -2^4+2^0$ $h(29) = h(37) = -2^5-2^3-2^1$ $h(30) = h(36) = -2^4+2^2-2^0$ $h(31) = h(35) = 2^3-2^0$ $h(32) = h(34) = 2^7+2^6+2^4-2^1-2^0$ $h(33) = 2^7+2^6+2^5+2^4+2^3+2^1+2^0$
$h_{Ma}(n)$		
$h(0) = h(18) = -2^1$ $h(1) = h(17) = 2^3+2^0$ $h(2) = h(16) = 2^1$ $h(3) = h(15) = -2^3-2^2$	$h(4) = h(14) = 2^3-2^0$ $h(5) = h(13) = 2^5-2^1$ $h(6) = h(12) = 2^5-2^0$	$h(7) = h(11) = -2^5-2^3-2^0$ $h(8) = h(10) = 2^7+2^5-2^2-2^0$ $h(9) = 2^8+2^5+2^3+2^0$
$h_{Mc}(n)$		
$h(0) = h(32) = 2^3$ $h(1) = h(31) = -2^2+2^0$ $h(2) = h(30) = -2^2-2^0$ $h(3) = h(29) = 2^3+2^1$ $h(4) = h(28) = 0$ $h(5) = h(27) = -2^3-2^2$	$h(6) = h(26) = 2^5-2^3-2^0$ $h(7) = h(25) = 2^1$ $h(8) = h(24) = -2^4-2^2$ $h(9) = h(23) = 2^5-2^3-2^0$ $h(10) = h(22) = 2^3+2^1+2^0$ $h(11) = h(21) = -2^5-2^2$	$h(12) = h(20) = 2^5+2^3-2^0$ $h(13) = h(19) = 2^3+2^2$ $h(14) = h(18) = -2^8-2^3-2^1$ $h(15) = h(17) = 2^7+2^3+2^0$ $h(16) = 2^8+2^6+2^5+2^2+2^1$

Chapter 4

Design of Modified FRM filters Using the Genetic Algorithm and Simulated Annealing

4.1 Introduction

In Chapter 3, we have proposed a hybrid algorithm OSGA for the design of FRM based FIR filter. A great benefit of the FRM approach is the significant reduction in the number of multiplication which can be as high as 98% as reported in [37]. This feature makes the FRM technique one of the best candidates for the design of high speed FIR filters.

It was demonstrated in [55-56] that it is possible to increase the speed of an FRM filter with a modified structure and SPoT techniques. The modified FRM structure utilizes several cascaded short filters to replace the long bandedge shaping filter resulting in faster speed and less hardware. However, the design of such a filter is rather complicated owing to the lack of a systematic design procedure in factorizing a long filter into several short filters and in searching for a set of optimal SPoT coefficients.

In Chapter 2, the design of cascade form filters with SPoT coefficients has been discussed. By utilizing the methods proposed in Chapter 2 together with the linear programming, all subfilters in the modified FRM structure can be designed in SPoT space. The drawbacks

are that the bandedge shaping filters and two masking filters are designed separately and the optimizations for all subfilters are done in an iterative way that does not guarantee a global optimal solution. It is interesting to know whether there is a way to design all subfilters simultaneously.

In Chapter 3, by integrating the GA and OS algorithm, a hybrid algorithm OSGA has been generated for the design of FRM filters. However, the observation from simulation shows that it is possible to find better method for the design of modified FRM filters due to the following reasons. There are two levels of subfilters in the modified FRM structure: first level is the bandedge shaping filter and a pair of masking filters and the second level is the short cascaded subfilters. In such complex structure, the OS algorithm may not bring enough diversity to the population. Moreover, the coefficient properties for direct form filters are not always true for cascade form filters.

To address the problem, a new hybrid genetic algorithm, GSA, is proposed in this chapter. The GSA combines the GA with the SA technique to produce an optimal solution. The GA is used as the basis of the algorithm and the SA is applied, when necessary, to optimize a certain number of fitter chromosomes to improve the convergence of GA. Due to the same reason discussed in Chapter 3, AGA is not used in the GSA. It is shown, by means of examples, that the proposed GSA can significantly reduce the coefficient word length in comparison with other methods.

This chapter is organized as follows. A modified FRM structure is introduced in Section 4.2. In Section 4.3, the hybrid genetic algorithm, GSA, is presented. Following that, the

GSA is applied for simultaneous design of all subfilters in the modified FRM structure. The design example is given is Section 4.5. Section 4.6 is dedicated to a conclusion.

4.2 A Modified FRM Structure

In a single stage FRM design, the length of the bandedge shaping filter $H_a(z)$ can be much longer than those of two masking filters $H_{Ma}(z)$ and $H_{Mc}(z)$. In some designs, the filter length of $H_a(z)$ can be 2 to 4 times longer than those of $H_{Ma}(z)$ or $H_{Mc}(z)$. With the unmatched filter length among $H_a(z)$, $H_{Ma}(z)$, and $H_{Mc}(z)$, the operation speed of an FRM filter mainly depends on the longest filter if all subfilters are implemented in a direct-form without employing pipeline. To improve the throughput of the overall system, a modified FRM structure is presented in [56] where the long bandedge shaping filter is replaced by several cascaded short filters. The z-transform transfer function of the modified FRM filter can be written as

$$H(z) = [H_{Ma}(z) - H_{Mc}(z)]\prod_{i=1}^{p} H_{ai}(z^M) + H_{Mc}(z) \times z^{-\frac{M(N_{a1}+N_{a2}+\ldots+N_{ap}-p)}{2}}, \tag{4.1}$$

where p is the number of short filters factorized from $H_a(z)$ and N_{ai} (i=1,2,…, p) is the lengths of the corresponding short filters.

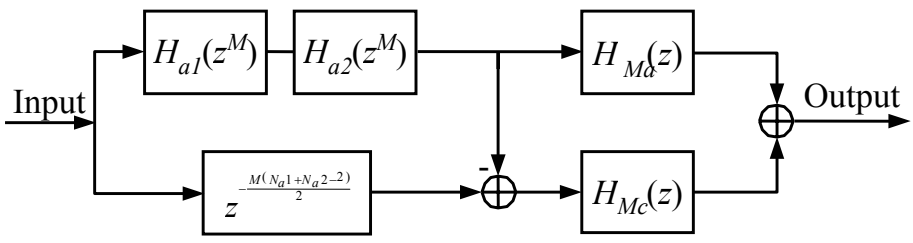


Fig. 4. 1 A realization structure for a modified frequency response masking approach.

Fig. 4.1 shows one of possible realization structures for the modified FRM approach, where $H_a(z)$ is factorized into two short cascaded subfilters, $H_{a1}(z)$ and $H_{a2}(z)$, respectively. The process of factorizing the bandedge shaping filter into several short filters is a time consuming process and is done in a heuristic manner, except for factorizing the long filter into two where interleave method [2] can be applied. That hinders the factorization from reaching optimal. Furthermore, a filter with SPoT coefficients is highly desired if high-speed is one of the design goals because the multiplication of data with a SPoT coefficient can be done with the help of a few adders. To this end, we present an efficient hybrid algorithm in the next section that optimizes all subfilters jointly in a discrete space.

4.3 A Hybrid Genetic Algorithm (GSA)

When conventional GA is used to solve the problem, it may not produce an optimum solution based on reasons given below. First, the chromosomes that represent the coefficients of all subfilters are very long due to the number of filters involved, e.g. at least four filters to be jointly optimized in the modified FRM filter. The population size has to be large enough to produce a useful result. This leads to very long computation time. Second, the determination of a good chromosome relies on the evaluation of a fitness function which involves the calculation of frequency responses of at least 4 subfilters. The large search space combined with the evaluation of a complicated fitness function puts an even high demand on computer resources leading to unacceptable

computation time. Third, the cascaded connection of bandedge shaping filters coupled with ripple compensation effect in an FRM filter makes the coefficients very sensitive to the changes of chromosome, in other words, the change of chromosome should be carried out gradually and large probabilities of crossover and mutation should be avoided. This is equivalent to an increase in population size prolonging computation time.

To address the above problems, it is necessary to reduce the population size in order to shorten the computation time. One of the possible ways is to introduce new operations for the chromosomes during the GA process while keeping the continuity of the evolution. We seek the help of another artificial intelligence algorithm, simulated annealing. The SA technique has shown its effectiveness of optimization in the factorization of a long filter in Section 2.3. By integrating the main features of SA into the GA process, an effective hybrid algorithm named as GSA is created. The introduction of the SA moves the GA away from local optima and prevents the GA from the premature convergence. At the same time, it helps to reduce population size of the GA.

In the hybrid GA, the optimization process is mainly governed by GA. The SA comes into the picture only when there is a sign showing slowdown in convergence of the GA process, i.e. no improvement for a pre-specified number of generations during the GA evolution. The SA can be considered as an external force that drives the GA away from a local minimum. As a result, the population size can be significantly reduced without affecting the effectiveness of the optimization algorithm. The role of the SA is to create variations of population by changing the collocations of digits to find optima configurations. In the GA approach, the coefficients of all subfilters are represented by a

chromosome which contains three digits, i.e. '1', '-1' and '0'. The GA operates on chromosomes by performing crossover and mutation that affects the number of '1s', '-1s' and '0s' in a chromosome. Unlike the GA, the SA does not change the number of '1s', '-1s' and '0s' in chromosomes obtained from the GA. It merely swaps the bits within a chromosome to check whether such actions lead to a better fitness value for a given chromosome. In the GSA, the GA can be viewed as a vertical evolution process while the SA can be seen as a horizontal evolution process.

There are two levels of convergence criteria in a GSA process. The first one is to check if the local optimum is reached and SA should be applied to current population to prevent the premature of the GA. The SA starts only if there is no improvement for a pre-specified number (denoted as G_{SA}) of generations during GA evolutions. Too small G_{SA} may destroy the evolution process of the GA while too large G_{SA} reduces the benefit of the SA. The reasonable range for G_{SA} is 100-500 for our problem. A more effective way is to change G_{SA} during the optimization process, i.e. randomly pick a value from 100 to 500 as G_{SA} during each convergence check. The second level convergence criterion is used to control the termination of the optimization process. The process terminates if one of the following two conditions is satisfied in the order given:

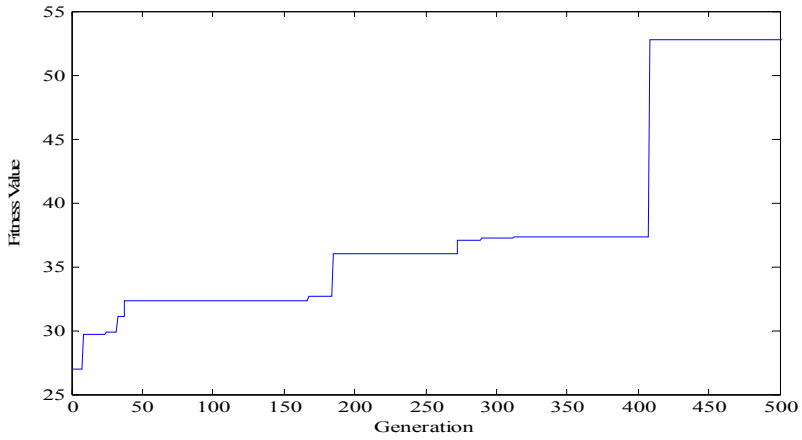
1. The desired objective value is reached.
2. There is no improvement in fitness value for a pre-specified maximum number of iterations.

The main steps of the algorithm are summarized as follows:

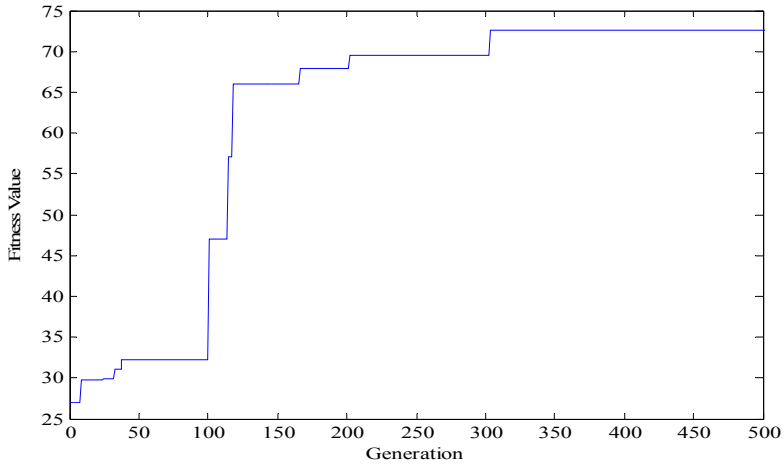
1. Create an initial population by randomly generating a set of feasible solutions (chromosomes).
2. Evaluate the fitness value for each chromosome in the population.
3. Apply GA operators, i.e. reproduction, crossover and mutation, to generate new populations.
4. Apply Replacement Strategies which combine Generation-Replacement with the elitist strategy, where two best chromosomes from the current population are copied to replace two worst chromosomes in the new population.
5. Check the first level convergence criterion. If it is satisfied, the SA is involved. Otherwise go to Step 3. The SA process includes two steps: first, we make a pre-specified number of copies of chromosomes whose fitness values are high among all chromosomes in the current population; second, apply the SA to optimize these chromosomes. If their fitness values are improved, we use them to replace the old chromosomes with worse fitness values in the current population.
6. Check the second level convergence criterion to decide if the optimization process should be continued, i.e. go to Step 3 or stopped.

To illustrate the effectiveness of the proposed GSA, let us design a linear phase low-pass FIR filter with normalized passband and stopband edges at 0.15 and 0.25, respectively. The maximum passband ripple is 0.01 and the minimum stopband attenuation is 40 dB. The filter length is 31 and all coefficients are represented using average 2 terms of 7 bits SPoT. The filter is synthesized by the GA and GSA, respectively, where G_{SA} is set to be 50 and the two level convergence conditions are checked for every 20 generations. The

convergence trends for the GA and GSA within the first 500 generations are shown in Fig. 4.2. At the first 100 generations, the two processes have nearly the same convergence trends, since they use the same initial population which is calculated from the real valued coefficients. At the 100th generation, the GSA finds much better solution due to the involvement of the SA, while the GA does not show any improvement on the best fitness value until 168th generation. It is clear that the GSA provides much better convergence performance than that of the GA.



(a) GA



(b) GSA

Fig. 4. 2 The convergence trends of the GA (a) and GSA (b) within the first 500 generations.

4.4 GSA for the Design of Modified FRM Filters

When the GSA is utilized to design the modified FRM filters, the chromosome is the encoded coefficients of all subfilters in the form of ternary encoding strings. Filter coefficients are allocated with different number of SPoT terms while keeping the total number of SPoT terms fixed.

Assume that $H_a(z)$ is factorized into p numbers of short filters, where p depends on the difference between the lengths of bandedge shaping filter and the longer masking filter. EQ. (2.8) is used as the objective function, where $H(e^{j\omega})$ and $H_d(e^{j\omega})$ are the frequency responses of the $H(z)$ in (4.1) and the desired filter, respectively. The fitness function is defined as

$$f = \frac{1}{O} + \frac{a_1}{S_T} + a_2 \times Z_T. \tag{4.2}$$

There are only three items in EQ. (4.2) while 4 items are included in EQ. (3.3). The fourth item in EQ. (3.3) is used to minimize the word lengths of the subfilters. In the modified FRM structure the bandedge shaping filter is factorized into several short filters. If the bandedge shaping filter and two masking filter are viewed as the first level subfilters, the short cascaded filters are viewed as the second level. To minimize their word lengths together according to the comparisons on the difference of their maximal and minimal absolute values, therefore, may not produce an optimal solution.

In the GSA, when there is no improvement in the best fitness value for a pre-specified number of generations in the GA evolution process, the SA algorithm is applied in the

current population. The energy E in the SA is defined to be the same as the objective function in EQ. (2.8) as

$$E = \max_{\omega \in [0,\omega_p] \cup [\omega_s,\pi]} [\, k(\omega) \,|\, H(e^{j\omega}) - H_d(e^{j\omega}) \,|\,]. \tag{4.3}$$

With this definition of E, the SA is able to find better chromosome configurations according to the minimax error criterion.

In the SA process, a fitter chromosome is chosen from current population as an initial solution. Its energy value is calculated as initial energy. The initial value of temperature T is set to be the initial energy value. The SA process is similar to that presented in Section 2.3. The major step is repeated here. Beginning with the initial solution and parameters of E and T, the reordering of bits in a chromosome is performed in an iterative manner. In the reordering process, a random length subsequence is firstly chosen from a random position in the chromosome. Then it is either replaced with the same subsequence in reversed order or is shifted to another random position between two genes in the chromosome with equal probability [21-22]. The energy value based on reordered chromosome is calculated. The new energy value and the corresponding chromosome configuration will be recorded as new best results to replace the current ones (such iteration process is called as a successful one) only when the energy value is smaller than the current best value or when a generated random number uniformly distributed within (0, 1) is less than the value of e-ΔE/T (where ΔE denotes the change of the energy value). For each successful iteration process, the temperature T would decrease by a specified factor. The reordering process is repeated until there is no more improvement in the energy value for the given number (L_{end}) of continuous iteration process or the

temperature is smaller than a specified ending temperature (T_{end}). The choices of L_{end} and T_{end} have been discussed in Section 2.3.2. A modified FRM filter consists of at least 4 subfilters and the effective length is very long. Hence, L_{end} is set to be 300 and T_{end} to be 10E-3, which is also validated using simulation examples.

4.5 Design Example

A. Example 1

To illustrate the proposed approach, let us consider the design of a narrow transition band low-pass FIR filter and compare our results with those in [56-58]. The filter meets the following specifications: normalized passband and stopband edges are at 0.2 and 0.21, respectively. The passband and stopband ripples are both 0.01. A minimax design needs 195 real coefficients to satisfy the above specifications. Using original FRM approach, the filter lengths of $H_a(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are 55, 17 and 31, respectively. The optimum interpolation factor is 4. Each coefficient is quantized into three terms of 10 bits powers-of-two value.

Since the bandedge shaping filter $H_a(z)$ consists of about 3 times more taps than the masking filter $H_{Ma}(z)$ and $H_{Mc}(z)$, $H_a(z)$ can be factorized into three short filters. The lengths of factorized filters $H_{a1}(z)$, $H_{a2}(z)$ and $H_{a3}(z)$ are 21, 19 and 17, respectively. The GSA is employed to optimize all subfilters, i.e. $H_{a1}(z)$, $H_{a2}(z)$, $H_{a3}(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$. The coefficients of $H_{a1}(z)$ use average 2 terms of 6 bits SPoT, $H_{a2}(z)$ uses average 2 terms of 5 bits SPoT and $H_{a3}(z)$ uses average 3 terms of 5 bits SPoT. The coefficients of $H_{Ma}(z)$

and $H_{Mc}(z)$ need average 2 terms of 7 bits SPoT. The total number of SPoT terms used in $H_{a1}(z)$, $H_{a2}(z)$, $H_{a3}(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$ are 19, 17, 21, 18 and 32 terms, respectively. The population pool size and the mating pool size are both 300. The crossover and mutation probabilities are set to be 0.6 and 0.001, respectively. For every 100 generations, the convergence conditions are checked and SA process is applied into the top 10% of current population when there is no improvement on GA evolution process for 200 generations. The cycle of evolution is repeated until the desired objective value is obtained, i.e. 0.01 in this example, or the fitness value of the population remains unchanged for 3000 generations. The stopband attenuation of the overall filter is 40.05 dB. The frequency responses of $H_a(z)$ and its subfilters, masking filters, and overall filter are shown in Figs. 4.3 - 4.5 respectively. Tables 4.1 and 4.2 list the coefficient values of $H_{a1}(z)$, $H_{a2}(z)$, $H_{a3}(z)$, $H_{Ma}(z)$ and $H_{Mc}(z)$. The convergence curve is shown in Fig. 4.6. The optimization performance in the earlier stage, i.e. fitness<80, is much better than that in the later stage. It can be seen from Fig. 4.6 that the evolution process evolved to 40.05 dB after 5600 generations when the convergence condition was met, i.e. the desired objective value is achieved.

For comparison, we also design the FRM filter using two other methods. First, the SA technique introduced in Section 2.3 is utilized to factorize the long bandedge shaping filter and linear programming is applied to quantize all coefficients into SPoT space in an iterative manner. The bandedge shaping filter with infinite precision coefficients and two masking filters with SPoT coefficients are designed firstly, and then $H_a(z)$ is factorized into p subfilters using the method presented in Section 2.3 followed by quantizing the

coefficients of each short filter into SPoT values using linear programming, finally the two masking filters are designed separately by taking the other subfilters as prefilters. Second, the GA based method proposed in Section 2.5 is applied for the design of bandedge shaping filter $H_a(z)$ with SPoT coefficient values. The two masking filters are designed separately by taking the other subfilters as prefilters in an interleaved procedure. The detailed information on the two methods can refer to [57] and [58], respectively. Table 4.3 is the comparison of word lengths of subfilters designed by using different methods, such as linear program in [56], SA [57] and GA [58] based methods, and the proposed GSA. It is shown that the GSA significantly reduces the coefficient word length for each subfilter in a modified FRM structure. This is mainly due to the joint optimization of all subfilters in the modified FRM structure in a discrete space.

B. Example 2

The design of an FRM filter specified in Section 3.4 is used as the example. The specifications are repeated here, i.e. the normalized passband and stopband edges are 0.3 and 0.305, respectively; the permitted maximum passband and stopband deviations are 0.01. It is reported in [36] that the lowest complexity can be achieved when the interpolation factor is chosen to be 6 or 9. In Section 3.4, we have presented the design with interpolation factor of 6. Here, we will show the design with interpolation factor of 9, where the lengths of $H_a(z)$, $H_{ma}(z)$ and $H_{mc}(z)$ with real valued coefficients are 45, 38 and 30, respectively, [36]. When the filter coefficients are quantized into SPoT by MILP, the lengths of $H_a(z)$, $H_{ma}(z)$ and $H_{mc}(z)$ are 47, 43, 35, respectively. The word lengths of each coefficient in $H_a(z)$, $H_{ma}(z)$ and $H_{mc}(z)$ are of 11, 12 and 12 bits with 3 terms of

SPoT, respectively. When the filter is designed using the OSGA and GSA, similar solutions are achieved with stopband attenuation of 40.27 dB. The lengths of $H_a(z)$, $H_{ma}(z)$ and $H_{mc}(z)$ are 45, 41 and 33, respectively, and the word lengths of each coefficient in $H_a(z)$, $H_{ma}(z)$ and $H_{mc}(z)$ are of 10, 11 and 11 bits with average three terms of SPoT, respectively. If the filter is designed using conventional GA, the stopband attenuation of the final solution is 38.59 dB with the same coefficient word length. Table 4.4 lists the designs achieved by different methods. To find the average performance of the OSGA and GSA, ten independent runs of each algorithm are performed for each design. Table 4.5 lists the best and worst solutions achieved by the OSGA and GSA. The sizes of population pool and the probabilities of crossover and mutation are the same as those in Example 1. It can be seen from the table that the best solutions from the OSGA and GSA are similar, while the worst solution from OSGA is better than that from GSA. The average performance of OSGA is, therefore, better than that of GSA. It is because that the OS algorithm works based on the rule of filter coefficients, while the SA algorithm works in a stochastic model. When all subfilters are implemented in the direct form, therefore, the optimization performance of the OSGA is better than that of the GSA. However, if not all the subfilters are implemented in direct form, the performance of GSA is better than that of the OSGA. If the filter in Example 1 is design using the OSGA, the desired filter meeting the given specifications cannot be achieved with the same word lengths as those from GSA.

4.6 Conclusion

In this chapter, a novel hybrid genetic algorithm has been proposed to jointly optimize all subfilters in a modified FRM structure in a discrete space. The proposed GSA integrates the main features of SA into the GA optimization process, where GA handles the overall optimization while SA is applied to prevent GA from the premature convergence. The FRM filters designed by the proposed GSA show reductions in hardware cost due to the fact that the total number of SPoT terms is included as one of terms in the fitness function during the optimization. The overall filter with powers-of-two coefficients is suitable for the low power VLSI implementations of high speed digital filters.

Table 4. 1 List of filter coefficients of $H_{a1}(z)$, $H_{a2}(z)$ and $H_{a3}(z)$

$h_{a1}(n)$	$h_{a2}(n)$	$h_{a3}(n)$
$h(0) = h(20) = 2^4-2^0$ $h(1) = h(19) = -2^2$ $h(2) = h(18) = -2^4+2^2$ $h(3) = h(17) = 2^0$ $h(4) = h(16) = -2^3$ $h(5) = h(14) = -2^2+2^0$ $h(6) = h(14) = -2^2+2^0$ $h(7) = h(13) = -2^4-2^2+2^0$ $h(8) = h(12) = 2^0$ $h(9) = h(11) = 2^5$ $h(10) = 2^5+2^2+2^0$	$h(0) = h(18) = -2^0$ $h(1) = h(17) = -2^3$ $h(2) = h(16) = 2^3-2^1$ $h(3) = h(15) = 2^3-2^0$ $h(4) = h(14) = 2^2-2^0$ $h(5) = h(13) = -2^1$ $h(6) = h(12) = -2^0$ $h(7) = h(11) = 2^3+2^0$ $h(8) = h(01) = 2^4+2^2-2^0$ $h(9) = 2^4+2^0$	$h(0) = h(16) = 2^3+2^1$ $h(1) = h(15) = 2^2+2^1$ $h(2) = h(14) = 2^2+2^1$ $h(3) = h(13) = -2^2+2^0$ $h(4) = h(12) = -2^1$ $h(5) = h(11) = -2^0$ $h(6) = h(10) = 2^4-2^2-2^0$ $h(7) = h(9) = 2^4+2^3+2^1+2^0$ $h(8) = 2^4+2^3+2^2+2^0$

Table 4. 2 List of filter coefficients of $H_{Ma}(z)$ and $H_{Mc}(z)$

$h_{Ma}(n)$	
$h(0)=h(16)= 2^2+2^1$ $h(1)=h(15)= -2^2$ $h(2)=h(14)= -2^2$ $h(3)=h(13)= -2^2$	$h(4)=h(12)= -2^4-2^1$ $h(5)=h(11)= 2^4+2^2-2^0$ $h(6)=h(10)= 2^5+2^2$ $h(7)=h(9)=2^6-2^3-2^2$ $h(8)= 2^6+2^5-2^3$
$h_{Mc}(n)$	
$h(0)=h(30)=-2^0$ $h(1)=h(29)= 2^0$ $h(2)=h(28)= 2^0$ $h(3)=h(27)=-2^0$ $h(4)=h(26)=-2^2$ $h(5)=h(25)= 2^2-2^0$ $h(6)=h(24)= 2^2-2^0$ $h(7)=h(23)=-2^1$	$h(8)=h(22)=-2^3-2^1$ $h(9)= h(21)= 2^2+2^0$ $h(10)=h(20)= 2^3+2^2-2^0$ $h(11)=h(19)=-2^2+2^0$ $h(12)=h(18)=-2^5+2^2$ $h(13)=h(17)= 2^3-2^0$ $h(14)=h(16)= 2^6+2^4-2^1$ $h(15) = 2^6+2^5+2^4+2^3+2^2+2^0$

Table 4. 3 A comparison on the word lengths of subfilters designed by using different methods

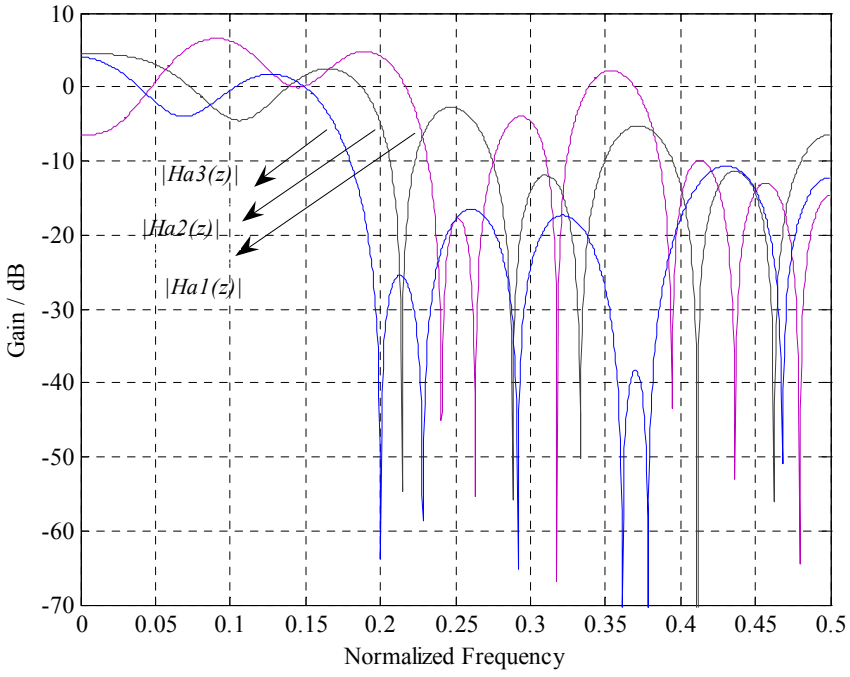
			Method in [56]	SA in [57]	GA in [58]	GSA
$H_{a1}(z)$	Filter length		19	21	21	21
	Coefficient word length	Average No. of SPoT term	3	3	3	2
		No. of bits	11	5	5	6
$H_{a2}(z)$	Filter length		19	19	19	19
	Coefficient word length	Average No. of SPoT term	2	3	3	2
		No. of bits	9	6	5	5
$H_{a3}(z)$	Filter length		19	17	17	17
	Coefficient word length	Average No. of SPoT term	2	3	3	3
		No. of bits	9	7	6	5
$H_{Ma}(z)$	Filter length		17	17	17	17
	Coefficient word length	Average No. of SPoT term	3	3	3	2
		No. of bits	10	8	8	7
$H_{Mc}(z)$	Filter length		31	31	31	31
	Coefficient word length	Average No. of SPoT term	3	3	3	2
		No. of bits	10	8	8	7

Table 4. 4 A comparison among the designs achieved by using different methods. No. Gen is the required number of generations to achieve the final solutions.

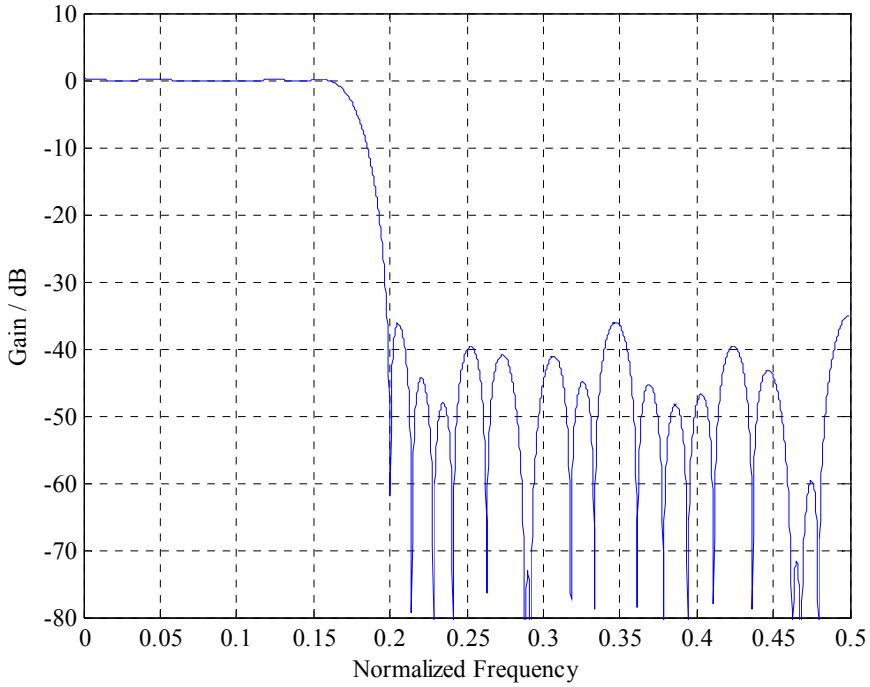
	Stopband ripple (dB)	No. of Gen.	Filter Length			Coefficient word length			Average No. of SPoT term		
			$H_a(z)$	$H_{ma}(z)$	$H_{mc}(z)$	$H_a(z)$	$H_{ma}(z)$	$H_{mc}(z)$	$H_a(z)$	$H_{ma}(z)$	$H_{mc}(z)$
GA	-38.59	3965	45	41	33	10	11	11	3	3	3
GSA	-40.27	1989	45	41	33	10	11	11	3	3	3
OSGA	-40.27	1763	45	41	33	10	11	11	3	3	3
MILP	-40.47		47	43	35	11	12	12	3	3	3

Table 4. 5 A comparison of the average performance between OSGA and GSA over ten independent runs. Successful runs refer to those which converge to the desired solutions while unsuccessful runs refer to the runs which cannot find desired solutions.

	Stopband ripple (dB)			No. of runs		
	Best	Worst	Average	Best	Successful runs	Unsuccessful runs
GSA	-40.27	-39.15	-39.99	3	7	3
OSGA	-40.27	-39.84	-40.12	4	7	3



(a) Three subfilters



(b) Overall filter

Fig. 4. 3 The frequency responses of the three subfilters (a) and overall filter (b) of $H_a(z)$.

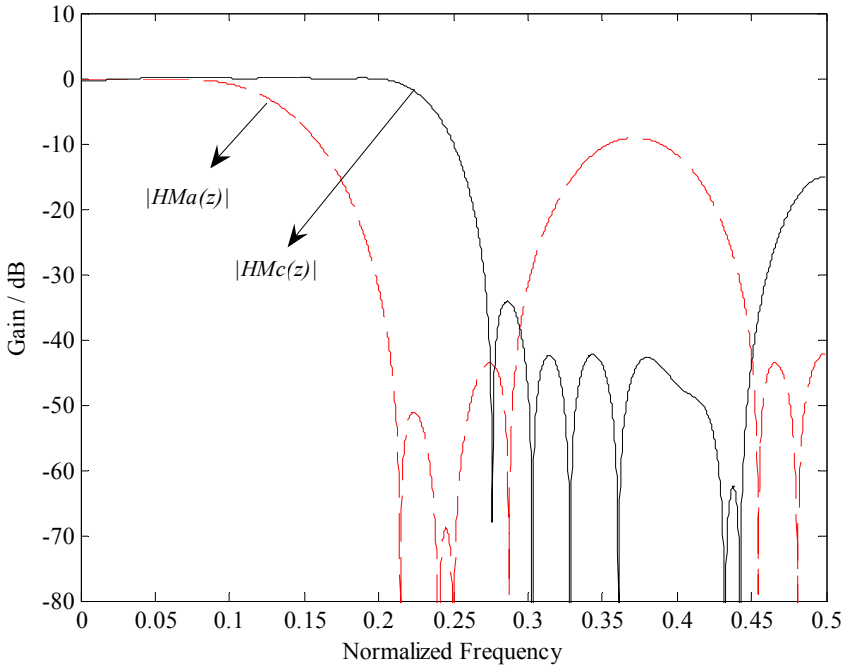


Fig. 4. 4 The frequency responses of $H_{Ma}(z)$ and $H_{Mc}(z)$.

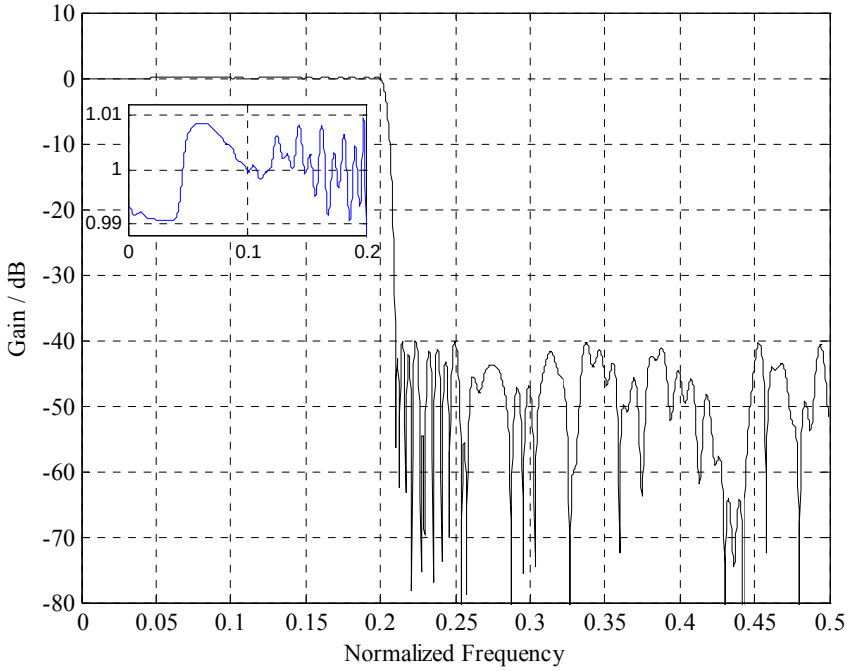


Fig. 4. 5 The frequency response of the overall filter.

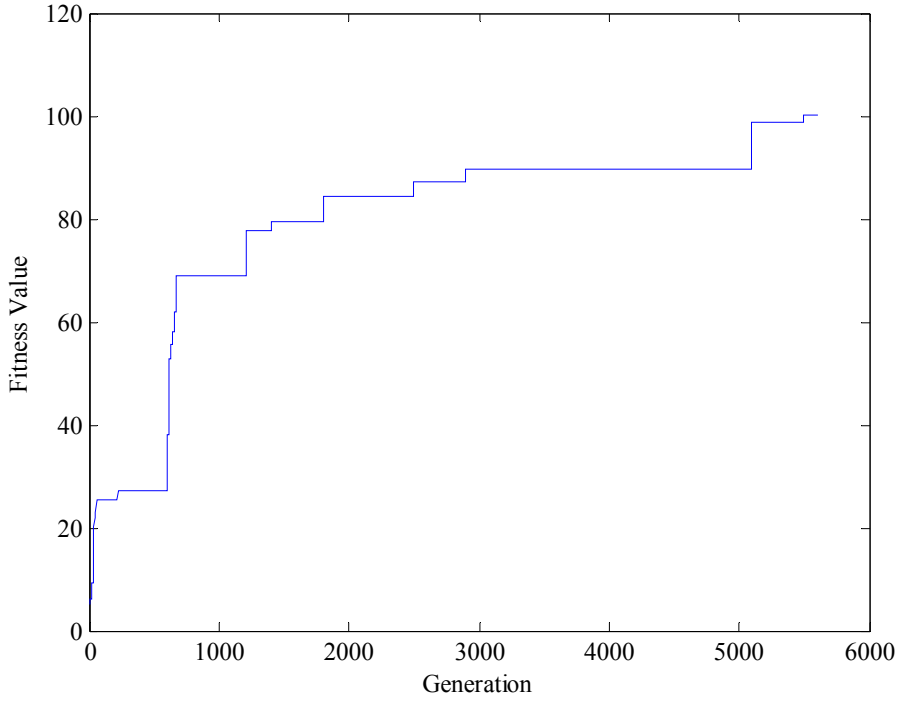


Fig. 4. 6 The convergence trend of the GSA.

Chapter 5

An Efficient Hybrid Genetic Algorithm for the Optimal Design of FIR Filters

5.1 Introduction

The design of FIR filters with SPoT coefficients is a complex process, where numerous local optima lead to high probability of premature convergence. The convergence speed is also very low, especially when the number of decision variables is large, e.g. the synthesis of FRM filters with very sharp transition band. In Chapters 3 and 4, two novel hybrid algorithms are proposed for the design of FRM filters with SPoT coefficients, which integrate the GA with the OS and SA algorithms, respectively. The results reported in the chapters are much better than those from the existing methods. However, due to the complicated computation of the objective function in FRM filter design, complex implementation of the optimization algorithm is not preferred. This why the AGA presented in Chapter 2 is not adopted in the OSGA and GSA.

In many DSP systems, the filter is not required to have sharp bandwidth. Compared with the FRM filters that have very sharp transition band, general FIR filters with relative broad transition bands have simple objective functions. Without the need of conducting

complicated computation of objective function, the research for general FIR filter design is focused on the solution quality and algorithm stability.

To this end, a hybrid genetic algorithm, AGSTA, is proposed, which is generated by combining the AGA, SA and tabu search (TS) algorithms. Compared with the GSA presented in Chapter 4, two enhancements are proposed in the AGSTA..

1) The AGA is used as the basis of the hybrid algorithm AGSTA, while the conventional GA is used in the GSA. Adaptive population size and probabilities of genetic operations not only improve the optimization performance but also increase the convergence speed.

2) The concept of "Tabu" in the TS technique is utilized to reduce the search space by considering the properties of filter coefficients. During the evolution process, the tabu-check is performed for each new solution. All infeasible solutions rejected by tabu-check will be repaired according to the repair mechanism before they are released to the new population.

In Chapter 3, we proposed a hybrid algorithm, which employs the OS algorithm to boost the performance of the GA. The OS algorithm is developed based on the coefficient properties of FIR filters. Since the TS technique is applied in the AGSTA to reduce the search space also according to the coefficient properties, the OS algorithm associated with the TS may limit the population diversity. It can be seen from the simulation study that the SA algorithm working in a stochastic model, is more suitable to be applied in the AGSTA.

The rest of the chapter is organized as follows. The hybrid genetic algorithm AGSTA is presented in Section 5.2. To show the effectiveness of the AGSTA, several design examples are given in Section 5.3. Section 5.4 is dedicated to a summary.

5.2 A Hybrid Genetic Algorithm (AGSTA)

In the previous chapters, the algorithms based on the GA and SA have been presented for the design of FIR filters. As discussed before, with the advancement of low cost and high speed computers, the GA, as a tool for search and optimization, has reached a mature stage. However, there are mainly three disadvantages, which prevent the GA from being applied in more applications, which have been covered in the previous chapter. First, the function of the GA can be visualized as a balanced combination of exploration of new regions in the search space and exploitation of already sampled regions. This balance, which critically controls the performance of the GA, is determined by the right choice of control parameters, i.e. the population size and the probabilities of crossover and mutation. The optimal control parameters are largely dependent on the optimization problem. Usually, multiple trials are needed to find optimal parameters for a special problem. Furthermore, parameter settings optimal in the earlier stages of the search typically become inefficient during the later stages. Hence, fixed parameters may deteriorate the efficiency of the GA. Second, the high probability of premature convergence is the major disadvantage of the conventional GA. It is especially serious when the GA is applied in complex applications with high problem dimension and large

search space. In these applications, the GA tends to produce a sub-optimal solution if there is no exterior "strength" or "instruction" to lead the GA out of local optima. Third, the convergence rate of GA is low if the search space is large. Numerous repetitive evaluations of candidature solutions should be carried out during the optimization process before the final solution can be found. This is the leading reason to explain the low convergence speed of the GA. To address these issues, a genetic approach based hybrid algorithm is proposed in this chapter. The main features of the AGA, SA and TS algorithms are integrated to yield a hybrid scheme that is abbreviated to AGSTA.

5.2.1 The Overview of AGSTA

In the AGSTA, the optimization process is mainly governed by the AGA proposed in Chapter 2. The adaptation of these control parameters not only improves the optimization performance of the GA but also avoids multiple trials of finding optimal control parameters. The SA comes to the picture when there is an indication that the AGA has entrapped in local optima, i.e. when better solutions cannot be found through a large number of evolution loops. The SA is served as exterior strength to help the AGA escape from the local optima, which is similar to the GSA. Some of better individuals in the population are chosen to perform the SA optimization and the new solutions found by the SA replace the worse members in the population. With the help of the SA, the population quality is improved by introducing better members and removing worse ones.

The large search space is one of the major reasons causing low convergence speed. This prompts us to consider if we can reduce the search space by avoiding impossible solutions. As pointed in Chapter 3, the coefficients located in the middle part have larger

absolute values and coefficients near to the beginning and end have smaller absolute values. According to the properties of filter coefficients, the concept of tabu in the TS technology is introduced to reduce the search space in the AGSTA. In each generation, tabu-check is performed on the new solutions in the population. If a new solution belongs to tabu, it may be accepted or rejected based on the judgment of aspiration criteria. The aspiration criteria allow overriding of tabu status. If its fitness value is better than the best one which has been found so far, it will be accepted; otherwise it is accepted according to the acceptance probability. The rejected candidates are repaired to make them feasible after tabu-check.

There are two level convergence criteria in the proposed algorithm. The first one is used to check whether AGA has entrapped in local optimum and SA should be applied. In the implementation, SA is applied if better solutions cannot be found for a pre-specified number of evolution iterations. The second level convergence criterion is used to control the termination of the whole search, which is defined as the same as that in the GSA.

The main steps of the algorithm are summarized as follows:

1) Specify the control parameters in the AGSTA, i.e. initial population size, maximal and minimal population size, and initial probabilities of genetic operations.
2) Create an initial population. The initial population is achieved from the real valued coefficients of the desired filter. The filter is designed with the given specifications in continuous space and quantized according to the specific word length. By this way a filter with discrete coefficient values can be obtained. The

initial population of filters is produced by perturbing the encoded ternary bits of the discrete coefficients.

3) Evaluate the fitness value for each chromosome in the population.
4) Apply AGA operations, i.e. reproduction, crossover and mutation, to generate offspring.
5) Check the first level convergence criterion to decide whether the AGA has reached a local optimum. If the condition is satisfied, apply the SA to a pre-specified number of better individuals in the offspring. The individuals with worse fitness values in population are replaced by the new members obtained from the SA.
6) Apply tabu-check to each individual found in this loop. If it is rejected by tabu-check, the repair mechanism is applied to make it feasible. The repaired one will be released to the offspring and replace the original one.
7) Copy two best chromosomes from the old population to replace two worst chromosomes in the offspring. Update the new population using the offspring.
8) Check the second level convergence criterion to decide whether the optimization process continues or terminates.

Fig. 5.1 shows the flow chart of the proposed AGSTA.

The implementation of the AGA and SA algorithm has been presented in Chapters 2 and 4, respectively. The objective function and fitness function in the GA and the energy function in the SA are defined as the same as EQ. (2.8), EQ. (2.17) and EQ. (4.3),

respectively. In the next sub-section, the implementation on how to reduce the search space by using the tabu search technology is presented.

5.2.2 Tabu-Check and Repair Mechanism

The diversity of population associated with large search space slows down the convergence speed of the GA. To speed up it, one efficient way is to reduce the search space by avoiding impossible solutions. Tabu search is an efficient optimization method that has been successfully utilized to solve a number of combinatorial optimization problems. Some researchers have combined Tabu Search and Genetic Algorithm to form hybrid tabu search genetic algorithm [62]. In the AGSTA, the concept of "tabu" is utilized to reduce the search space according to the properties of filter coefficients. For SPoT based FIR filter design, the tabu move can be defined as a string with the same length of a chromosome, where the earlier bits of the coefficients that is located near to the beginning and end are set to be "1" and the other parts of the string are set to be "*" according to the coefficient properties. For example, if a 10-order systematic lowpass filter is designed and each coefficient is represented using 5 bits SPoT terms, the tabu move can be defined as "111** 111** 11*** ***** ***** *****".

In the AGSTA, tabu-check is applied to each of new individuals in the population. If it belongs to tabu, this individual may be accepted or rejected according to aspiration criteria which allow overriding of tabu status. If its fitness value is better than the best one found so far, it will be accepted；otherwise it can only be accepted according to an acceptance probability P_a. To reduce the search space as well as keep the population

diversity, we define an adaptive acceptance probability that is adjusted according to the population diversity. Here, the population diversity is reflected using the ratio of the sum of the real fitness values and the imaginary values assuming that all members have maximal fitness. A weighting coefficient that is larger than 1 is used in the denominator to reduce the ratio to make it suitable to define the acceptance probability. It was chosen to be 2.65 according to simulation study. Therefore, the adaptive acceptance probability is defined as

$$p_a = \frac{\sum_{i=1}^{C_P} f_i}{2.65 \times \max_{i=1}^{C_P} f_i \times C_P}, \tag{5.1}$$

where C_P is the population size and f_i is the fitness value of the i^{th} individual. Although we cannot claim that it is the optimal definition, it can work very well in simulation. Larger p_a indicates smaller diversity of population. To increase the population diversity, the new member that belongs to tabu is accepted with a larger probability; to increase the convergence speed, it is accepted with a small probability.

After tabu-check, the rejected members are repaired, where the OR operation is performed between the rejected chromosome and a repair string. The repair string has the same length as the chromosome and is constructed in such a way that has value of 0 in the genes expected to be 0 and has 1 in the other parts. If the previous example is considered, the repair string can be defined as: 00011 00011 00111 11111 11111 11111.

Fig.5.2 shows the procedure of Tabu-check and repair.

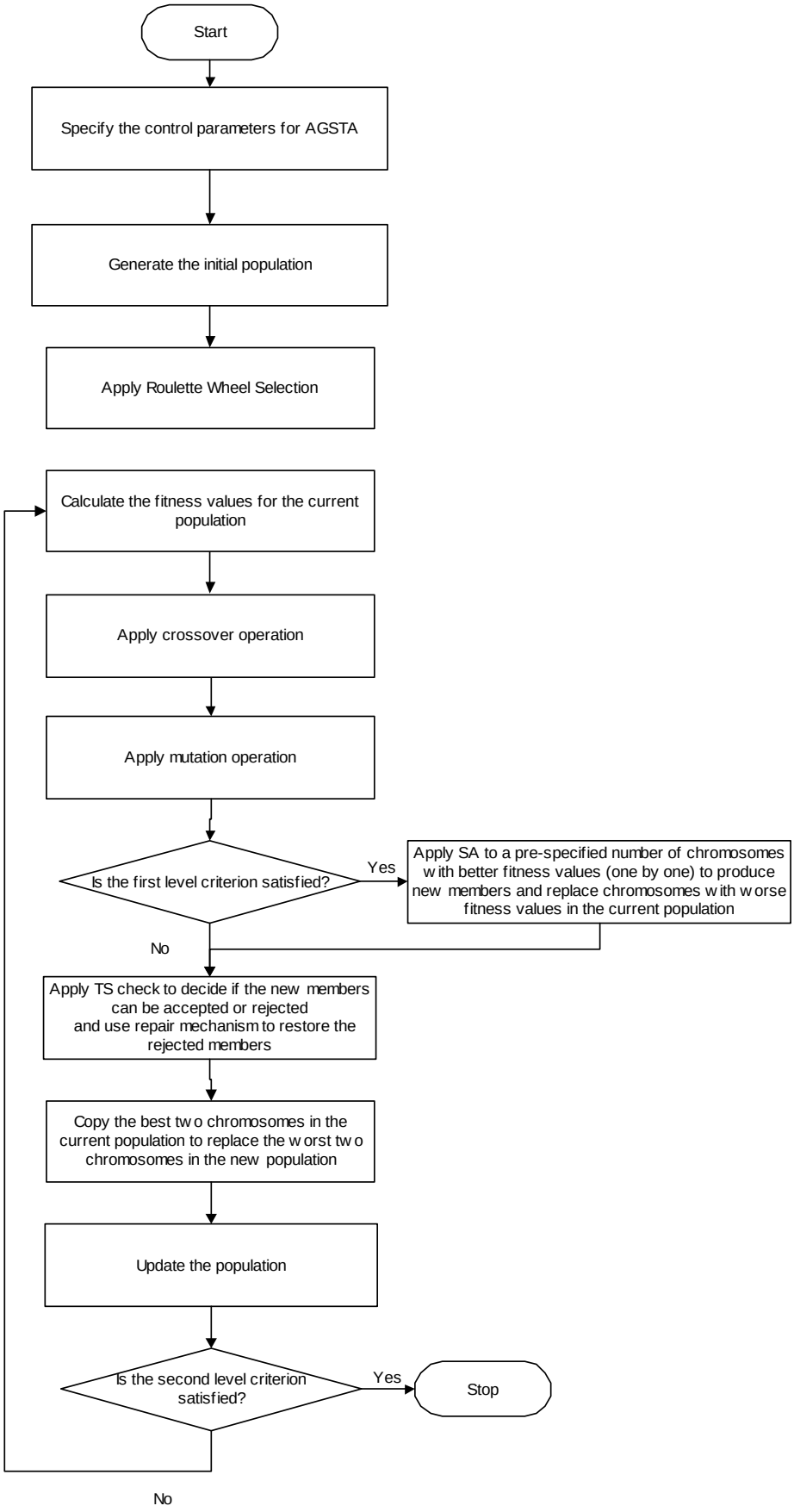


Fig. 5. 1 The flow chart of AGSTA.

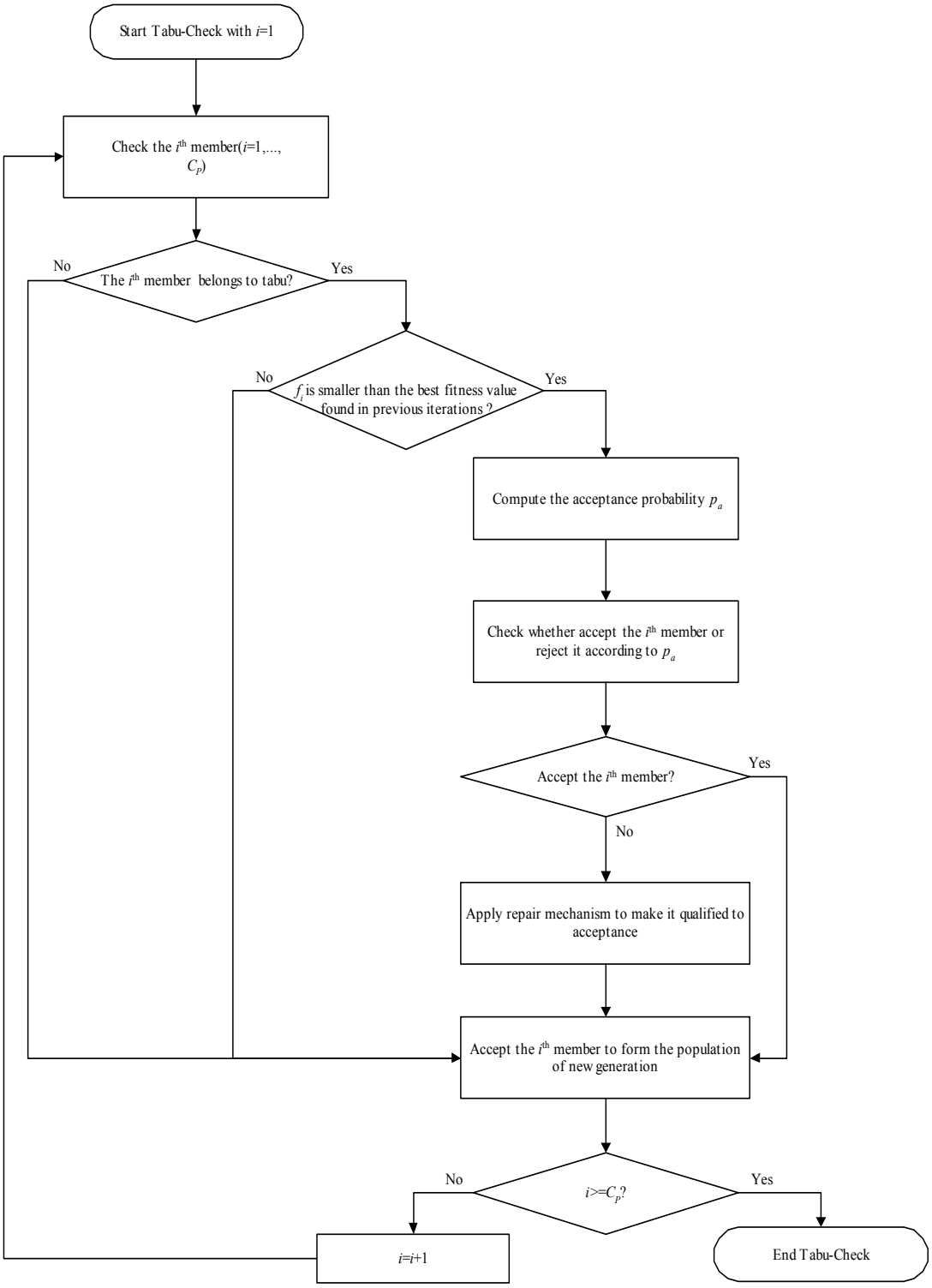


Fig. 5. 2 The flow chart of TS implementation.

5.3 Design Example

To illustrate the proposed technique, let us consider the design of two linear phase FIR low-pass filters. Our results are compared with those obtained from the MILP [1], polynomial-algorithm [8] and SA [10]. The average performance of the GA and AGSTA are also compared based on ten independent runs of each algorithm.

A. Example 1

The first example is designed in [1, 6, 7, 8 and 10] that is a linear phase FIR low-pass filter meeting the following specifications:

Normalized passband edge: 0.15

Normalized stopband edge: 0.25

Maximum passband ripple: 0.01

Minimum stopband attenuation: 40 dB

An FIR filter of length 22 with infinite precision coefficients satisfies the given specifications. If such a filter is designed by the method in [8], the filter length is 23 when the coefficients are represented by average 2 terms of 8-bit SPoT. The minimum stopband attenuation is 41.35 dB. When the filter is designed with the same length of 23 by using the proposed method, the filter coefficients only use average 2 terms of 7 bits SPoT. The stopband attenuation is 40.19 dB. The frequency response of the filter is shown in Fig. 5.3. In the AGSTA, the initial population pool size is 50. The lower bound and the upper bound of population are set to be 10 and 1000, respectively. For every 100 generations, the convergence conditions are checked. If the first level convergence

condition is met such that no further improvement can be achieved for 200 generations in the evolution process, the SA process is applied into the top 10% of current population. The cycle of evolution is repeated until the desired objective value is obtained or the best fitness cannot be improved for 1000 generations.

The comparison of hardware cost between the filters designed by the polynomial-time algorithm [8] and the proposed AGSTA are listed in Table 5.1. It can be seen that the filter designed by the AGSTA uses 6.25% less D flip-flops and 12.5% less full adders than the one obtained from the polynomial-time algorithm. Assuming that each delay element is implemented by D-flip-flop with 8 transistors and all adders are carry ripple adders with 28 transistors for 1-bit full adder cell, the AGSTA achieves about 11.42% savings in terms of the number of the transistors in comparison with the method in [8].

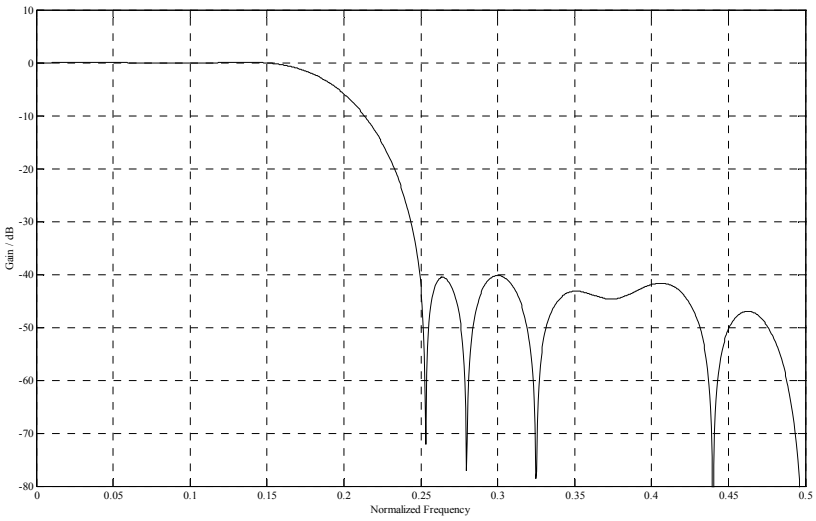


Fig. 5. 3 The frequency response of the filter with length of 23.

Table 5. 1 A comparison of hardware cost between the designs using AGSTA and polynomial-time algorithm [8]

	No. of bits	No. of SPoT terms	D-flip-flop	full adders
Proposed AGSTA	7	20	330	420
Polynomial-Time algorithm	8	22	352	480
Achieved savings	1	2	6.25%	12.50%

In order to compare the AGSTA with the methods proposed in [8] and [10], the same design parameters as those in [8] and [10] are chosen. The word length is 9 excluding sign bit. The filter is designed with different lengths ranging from 27 to 35. Table 5.2 summarizes the results for these designs, where we compare our results with those obtained from MILP [1], polynomial-algorithm [8], SA [10] and GA. It is illustrated from Table 5.2 that the normalized peak ripples of the filters designed by the AGSTA are up to 9.81 dB, 1.29 dB and 7.58 dB and 4.40 dB smaller than those from the MILP [1], polynomial-algorithm [8] and SA [10], and GA, respectively. The frequency responses of the filters with lengths of 27, 28, 29, 31 and 33 designed using the AGSTA are shown in Fig. 5.4. It is clear that the proposed hybrid algorithm AGSTA can find much better solutions than the individual implementation of SA and GA and outperforms the polynomial-algorithm on the improvement of normalized peak ripples.

The filter is also designed with 25-tap as that in [1], [6] and [7]. The coefficients are represented by fixed 2 terms of 9 bits in [1] and [6] and average 2 terms of 9 bits in [7]. According to [7], the filter designed using MILP techniques [1] had a stopband attenuation of 41 dB approximately, the filter designed using the local search technique [6] had a stopband attenuation of 38 dB approximately, and the filter designed using the improved local search technique [7] had a stopband attenuation of 43.8 dB. If the filter is

designed by using the proposed algorithm, the normalized peak ripple is 44.4 dB with the same coefficient word length.

B. Example 2

The second filter was specified in [8 and 10] that meets the following specifications: the normalized passband and stopband edge of 0.15 and 0.22, respectively; the ratio in passband and stopband equals one. With the same coefficient precision, the filter is designed with different lengths ranging from 31 to 41. The results are listed in Table 5.3. It can be seen that the normalized peak ripples of the filters designed by the AGSTA are up to 7.08dB, 0.91dB and 5.52dB smaller than those from MILP, polynomial-algorithm and SA, respectively. The frequency responses of the filters with lengths of 31, 33, 35, 37, 39 and 41 designed using AGSTA are shown in Fig. 5.5.

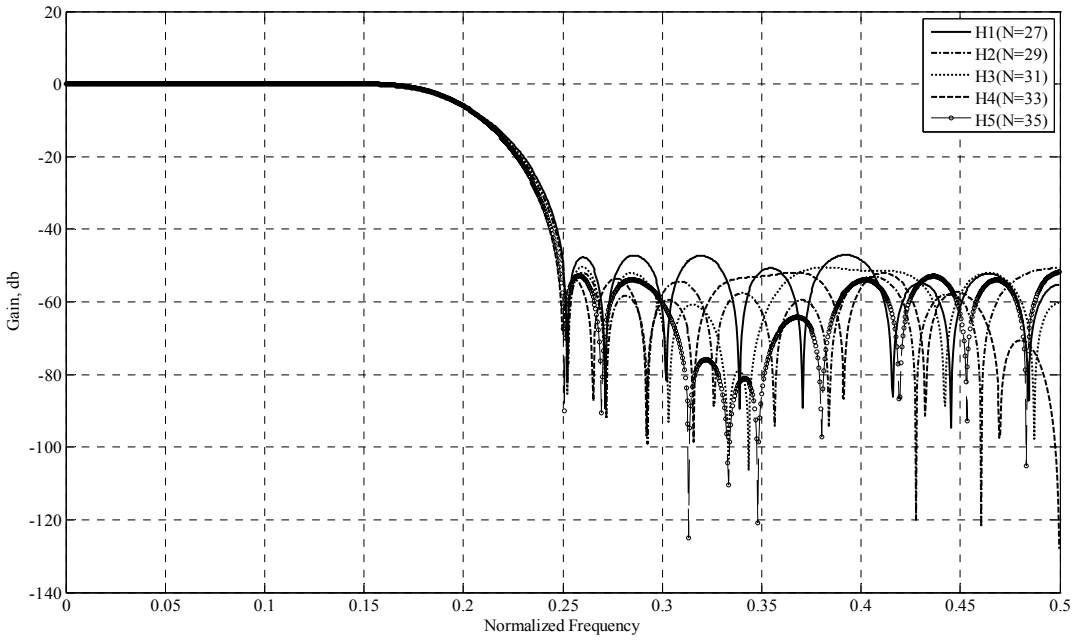


Fig. 5. 4 The frequency responses of the filters with lengths of 27, 28, 29, 31 and 33.

Table 5. 2 A comparison of normalized peak ripples (NPR) among different methods. The word length (excluding sign bit) is 9. The designs from MILP [1] use fixed 2 SPoT terms for each coefficient. Total of SPoT terms is 2*N* for the designs from polynomial algorithm [8], SA [10] and GA.

Filter length *N*	Infinite word length	MILP	Polynomial-algorithm	SA	GA		Proposed AGSTA	
	NPR(dB)	NPR(dB)	NPR(dB)	NPR(dB)	NPR(dB)	No. of SPoT terms	NPR(dB)	No. of SPoT terms
27	-47.74	-40.41	-45.77	-41.3	-44.50	27	-47.06	30
29	-53.04	-41.88	-50.19	-43.1	-46.28	35	-50.68	32
31	-53.35	-41.88	-50.97	-43.1	-46.21	33	-50.46	33
33	-57.84	-42.09	-53.64	-44.7	-48.28	33	-51.90	31
35	-60.12	-44.22	-51.56	-44.7	-48.43	37	-51.84	36

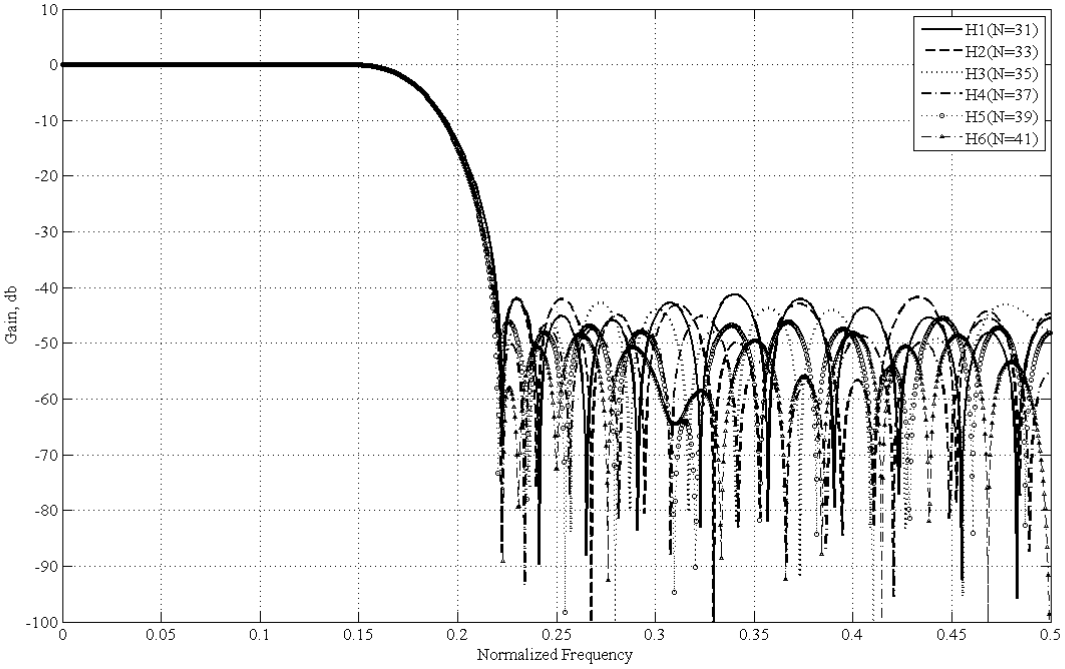


Fig. 5. 5 The frequency responses of the filters with lengths of 31, 33, 35, 37, 39 and 41.

Table 5. 3 A comparison of normalized peak ripples (NPR) among different methods. The word length (excluding sign bit) is 9. The designs from MILP [1] use fixed 2 SPoT terms for each coefficient. Total of SPoT terms is 2*N* for the designs from polynomial algorithm [8] and SA [10].

Filter Length *N*	Infinite word length	MILP	Polynomial-algorithm	SA	Proposed AGSTA	
	NPR(dB)	NPR(dB)	NPR(dB)	NPR(dB)	NPR(dB)	No. of SPoT terms
31	-42.29	-36.86	-40.36	-39.7	-41.27	33
33	-42.82	-37.81	-41.36	-39.7	-41.67	37
35	-44.83	-38.43	-42.31	-41.0	-42.77	39
37	-47.73	-39.61	-44.87	-41.5	-45.13	39
39	-48.31	-40.20	-45.91	-41.6	-45.51	41
41	-51.53	-41.14	-48.20	-42.7	-48.22	42

Due to the probabilistic nature of artificial intelligence based optimization algorithms, the stability of convergence performance in different runs is an important indicator concerning the optimality of the algorithms. Ten independent runs of the GA and AGSTA are performed for every design to achieve their average performance. The following parameters are recorded for each design to assess the convergence performance: the mean number of function evaluations in 10 runs and the mean NPR of the best filters found in the 10 runs. The comparison is based on the average performance, which is listed in Table 5.4. It is shown from this table that the AGSTA can find better solutions with smaller mean values of NPR using much fewer function evaluations. The statistic shows that the solutions found in 10 runs are only slightly different, which encourages the confidence in the quasi-optimality of our results. Furthermore, at least 6 runs out of 10 can converge to the best solution, which illustrates the optimization stability of the

AGSTA. Compared with the GA, the AGSTA achieves not only the improved solution quality but also the considerable reduction of computational effort.

Table 5. 4 A comparison between GA and AGSTA over 10 independent runs. Successful runs refer to the runs where the best solutions can be found.

Filter Length *N*	GA		Proposed AGSTA		
	Mean NPR (dB)	Mean Function Evaluations	Mean NPR (dB)	Mean Function Evaluations	No. of desired run
31	-39.79	387,000	-41.17	219,736	8
33	-39.84	429,500	-41.43	224,745	7
35	-40.85	441,000	-42.59	245,529	8
37	-41.91	478,500	-45.02	274,372	8
39	-42.03	534,500	-45.27	290,530	6
41	-43.46	569,000	-48.12	312,949	8

5.4 Conclusion

A novel hybrid genetic algorithm (AGSTA) is proposed for digital FIR filter design, which integrates the main features of the AGA, SA and TS algorithms. The AGA with adjustable population size and probabilities of genetic operations is used as the basis of the algorithm. The SA is applied when it is implied that the evolutionary process has reached a local optimum. The use of SA algorithm is to help the AGA escape from the local optima and prevent it from premature convergence. The TS is introduced to improve the convergence speed by reducing the search space according to the properties of filter coefficients. It is shown by means of examples that the normalized peak ripples of filters can be reduced with the help of the AGSTA. In comparison with other genetic algorithm, the AGSTA can not only improve the solution quality but also reduce the computational effort.

Due to the excellent optimization performance, the proposed AGSTA, can also be utilized in the design of FRM filters. However, we should note two points in such applications. First, due to the reason pointed in Chapters 3 and 4, the AGA may not be suitable when the computation of fitness function is too complex. Hence, compared with the AGA, the GA may be more competent in FRM filter design. Second, in the modified FRM structure, the bandedge shaping filter is replaced by several cascaded subfilters. However, the coefficient properties of direct form filters may not be applicable in cascade form filters. Thus, the Tabu-check may only be applied to the masking filters.

Chapter 6

A Modified Micro-Genetic Algorithm for the Design of FIR Filters

6.1 Introduction

As a stochastic algorithm, the GA is a robust and powerful optimization technique which has been applied into the design of digital signal processing systems by many researchers [11-15]. In the previous chapters, several novel algorithms based on the GA have been proposed for the design of FIR filters with different design requirements and structures. However, the main drawback of the GA is that long computation time is required for the repetitive evaluations of a large population of candidature solutions. Although designing FIR filters is more than a job that can be done off-line, the computational complexity does become an issue. Therefore, how to find acceptable solutions with the least computational cost is worth being investigated. A micro-genetic algorithm (MGA) [63] has been proposed to overcome the drawbacks of low convergence speed of the GA, which utilizes a very small population to reduce the computation time. However, there is a high likelihood that the MGA entraps into a local optimum when the MGA is applied in

complex applications. To address this issue, the MGA is modified to include a strategy that varies the probabilities of crossover and mutation during the evolutionary process. The modified MGA is utilized for the design of digital FIR filters both in direct realization and cascade realization with SPoT coefficient values. The effectiveness of the modified MGA can be shown by the design examples.

This chapter is organized as follows. The modified micro-genetic algorithm with varying probabilities of crossover and mutation is proposed in Section 6.2. In Section 6.3, the modified MGA is utilized in the design of digital FIR filters in direct realization with SPoT coefficient values. In the following section, the modified MGA is applied for the design of high speed FIR filter in cascade realization with SPoT coefficients. In Chapters 2-6, several algorithms are developed for different design problems and requirements concerning FIR digital filter design with SPoT coefficients. To gain an overall impression on these algorithms, a comparison among the AGA, OSGA, GSA, AGSTA and MGA are given in Section 6.5. A summary is presented in Section 6.6.

6.2 A Modified MGA with Varying Probabilities of Crossover and Mutation

The MGA starts with a population of, say, five randomly generated individuals. The individual with the largest fitness value is copied to the next generation and the other four individuals are determined with the tournament selection strategy. The four individuals are paired randomly and adjacent individuals in a pair compete to produce the children individuals in the subsequent generation using the GA operations,, except that the

crossover probability is set to 1.0 and the mutation probability is 0. After that the convergence of the small population is checked. If the population converges, a new population is generated by randomly initiated four new individuals and the fittest individual from the previous generation. Otherwise the previous generation is used to produce the subsequent generation by genetic operations. In the MGA, the mutation operation is not employed since enough diversity is introduced by generating the new individuals after convergence of a population. The optimization process of the MGA is different from that of the conventional GA. The convergence speed of the small population is very high with the crossover probabilities 1.0 and mutation probabilities zero. The introduction of the new population after convergence and the retention of the previous fittest individual make it possible to find better solution for the problem under optimization.

In the GA, the convergence criteria control how often new population are generated. In our implementation, the convergence condition is defined as the sum of the differences between the fittest individual and each of the rest four individuals are less than 5% in terms of number of bits in the chromosomes, i.e., the individuals in the whole population are moved towards the fittest individual. If the convergence condition is met, a new population is generated by randomly initiated four new individuals and the fittest individual from the previous generation. The MGA process is terminated if there is no improvement on the fitness values for 3000 generations.

However, the performance of the MGA will be affected when the lengths of chromosomes are very long. It is because MGA with a small population size and the

probabilities of crossover and mutation 1.0 and 0 is easy to become premature to a local optimal solution. In the filter design, we usually face such cases where very long chromosomes are needed. To address this problem, the MGA is modified to avoid trapping into local optima by adjusting the probabilities of crossover and mutation in different convergence stages instead of the crossover probabilities 1.0 and mutation probabilities zero for the whole evolutionary process. The probabilities of crossover and mutation will be changed according to the degree that the convergence criterion is met. The crossover probability is set to be 0.9 and the mutation probability is 0.001 when the similarity between the best individuals and the rest is more than 30%. The crossover probabilities is set to be 0.8 and mutation probabilities is 0.01 when the similarity is more than 50%. The probabilities of crossover and mutation are changed to 0.7 and 0.015, respectively, when the similarity is increased up to 70%. The two probabilities are changed to 0.6 and 0.02, respectively, when the differences between the fittest individual and each of the rest 4 individuals are less than 10%. Different probabilities in different evolution stages can prevent premature convergence of the MGA and improve the convergence speed for long chromosomes optimization cases. The principal is to increase population diversity by adjusting operation probabilities with the increase of chromosome uniformity. These values are chosen within the reasonable ranges of the probabilities of genetic operations, i.e. 0.6-0.9 for crossover probability and 0.001-0.02 for mutation probability. The validation is performed by simulation study. With the varying probabilities of crossover and mutation, the MGA can efficiently deal with the optimization problems with long chromosomes.

6.3 MGA for the Design of Digital FIR Filters with SPoT Coefficients

In this section, the modified MGA is applied in the design of digital FIR filters with SPoT coefficient values. The definitions of the objective function and fitness function are the same as those in Chapter 5. The design of a linear phase FIR low-pass filter is considered to illustrate the proposed technique, which meets the following specifications: the normalized passband and stopband edges are 0.18 and 0.27, respectively; the maximum passband ripple is 0.01 and the minimum stopband attenuation is 40 dB.

An FIR filter of length 24 with infinite precision coefficients satisfies the given specifications. If such a filter is designed by the proposed method, the filter length increases to 27. The minimum stopband attenuation of the overall filter is 40.37 dB when each coefficient is quantized into 2 terms of 8-bit SPoT. The frequency response of the filter is shown in Figure. 6.1. The total number of fitness function evaluations is 26,560.

To compare the proposed MGA with the conventional GA, the same filter is designed by the GA. The population pool size and the mating pool size are both 100. The crossover and mutation probabilities are set to be 0.8 and 0.01, respectively. The cycle of evolution is repeated until the fitness value of the population remains unchanged for 1000 generations. A similar set of coefficients was found, after the fitness function was evaluated for 243,700 times.

Table 6.1 lists the details of the filter designed using the modified MGA and GA. It is seen that the MGA takes much less computation time to find the same design than the GA.

Table 6. 1 A comparison between the filters designed by using GA and MGA

	Stopband attenuation	Lengths	No. of bits	No. of fitness function evaluations
GA	40.37 dB	27	8	243,700
MGA	40.37 dB	27	8	26,560

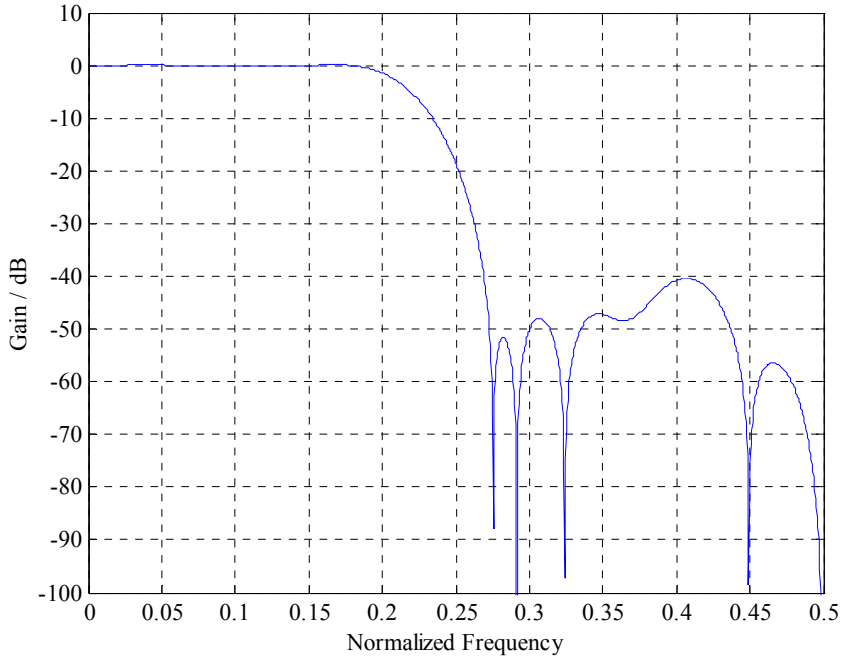


Fig. 6. 1 The frequency response of the direct form filter designed by using MGA.

6.4 MGA for the Complexity Reduction of High-Speed FIR Filters

As discussed in Chapter 2, to design high speed FIR filter, one of the useful approaches is to factorize a long filter into several short filters each with SPoT coefficients. For the

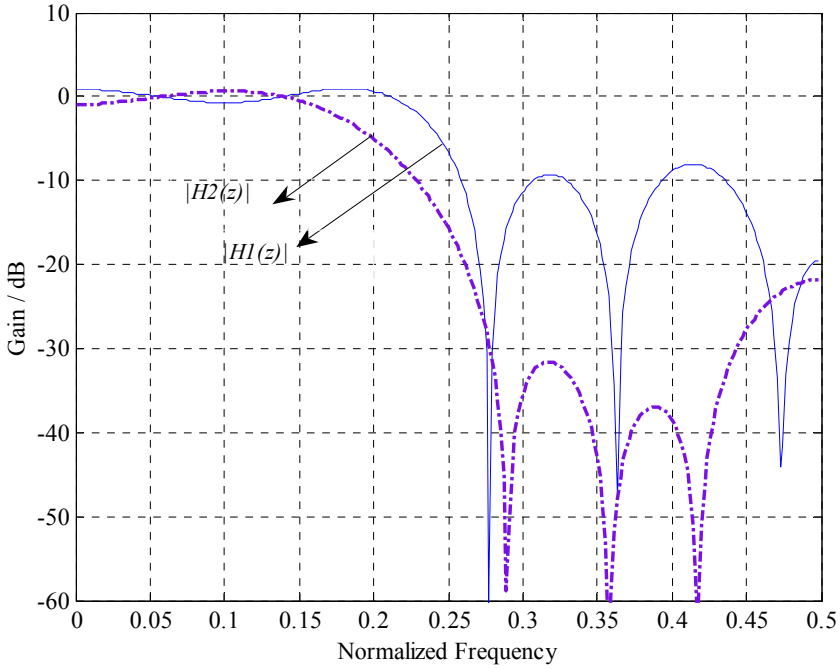
optimal design of a cascade discrete coefficient filter, the GA is a good candidate as it is capable of solving nonlinear optimization problems which are essential for factorizing a long filter and quantizing the coefficients. However, huge amount of computer resource and long computation time are required when conventional GA is applied in such optimization problems. In this section, the modified MGA is applied for the design of cascade form filters with SPoT coefficients, resulting in very short computation time. The objective function and the fitness function are the same as EQ. (2.8) and EQ. (2.17), respectively.

A linear phase FIR low-pass filter that meets the following specifications is designed by using the proposed technique: the normalized passband and stopband edges are 0.15 and 0.27; the maximum passband ripple is 0.01 and the minimum stopband attenuation is 40 dB. An FIR filter of length 19 with infinite precision coefficients satisfies the given specifications. The filter length increases to 21 if each coefficient is quantized into 2 terms of 8-bit SPoT.

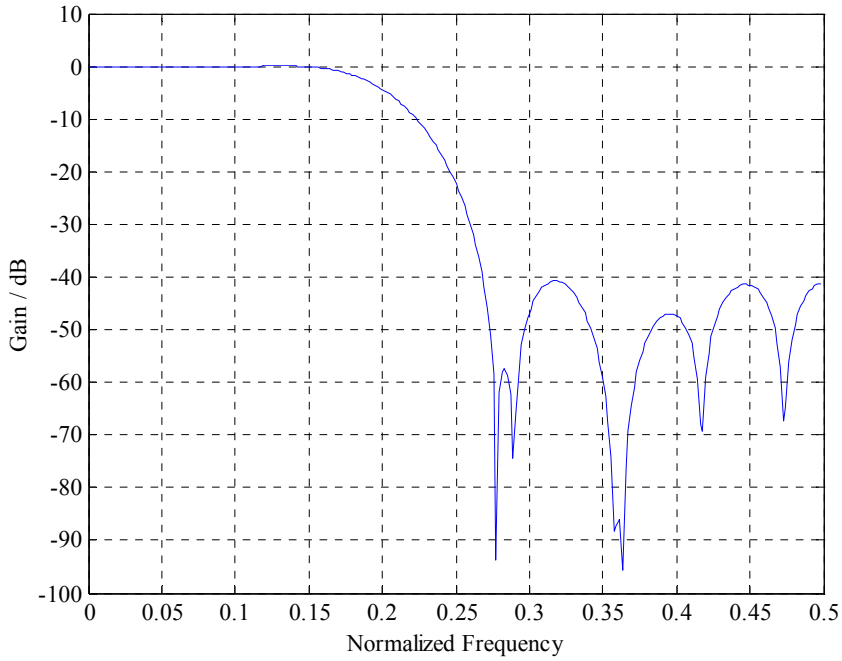
If such a filter is designed with two cascaded subfilters by the proposed method, the subfilter lengths of $H_1(z)$ and $H_2(z)$ are 13 and 9, respectively. The minimum stopband attenuation of the overall filter is 40.81 dB when coefficients of $H_1(z)$ and $H_2(z)$ are represented by 3 terms of 6-bit SPoT and 2 terms of 6-bit SPoT, respectively. The frequency responses of two subfilters and the overall filter are shown in Fig. 6.2. Using the proposed method to factorize the filter into three subfilters, the lengths of $H_1(z)$, $H_2(z)$ and $H_3(z)$ are 9, 7 and 7, respectively. The coefficients of $H_1(z)$, $H_2(z)$ and $H_3(z)$ use 2 terms of 4-bit SPoT, 5-bit SPoT and 4-bit SPoT, respectively. The stopband attenuation is

40.63 dB. The frequency responses of three subfilters and the overall filter are shown in Fig. 6.3. When the same filter is factorized into 4 cascaded subfilters, the lengths of $H_1(z)$, $H_2(z)$, $H_3(z)$ and $H_4(z)$ are 7, 7, 5, and 5, respectively. The coefficients of $H_1(z)$ use 2 terms of 5-bit SPoT, $H_2(z)$ uses 2 terms of 4-bit SPoT, $H_3(z)$ uses 2 terms of 4-bit SPoT, and $H_4(z)$ uses 2 terms of 3-bit SPoT. The stopband attenuation is 40.38 dB. The frequency responses of four subfilters and the overall filter are shown in Fig. 6.4.

Table 6.2 lists the details of each design from the conventional GA and modified MGA. The computation effort for different cases in terms of the number of function evaluations are shown in Table 6.2. It is clear that considerable savings on computational effort can be achieved by using the MGA.

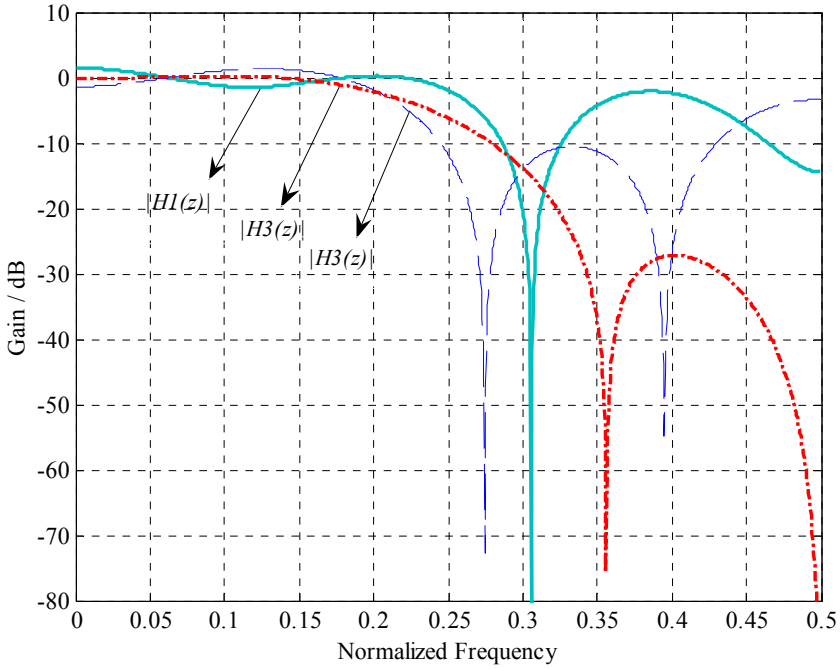


(a) **Two subfilters**

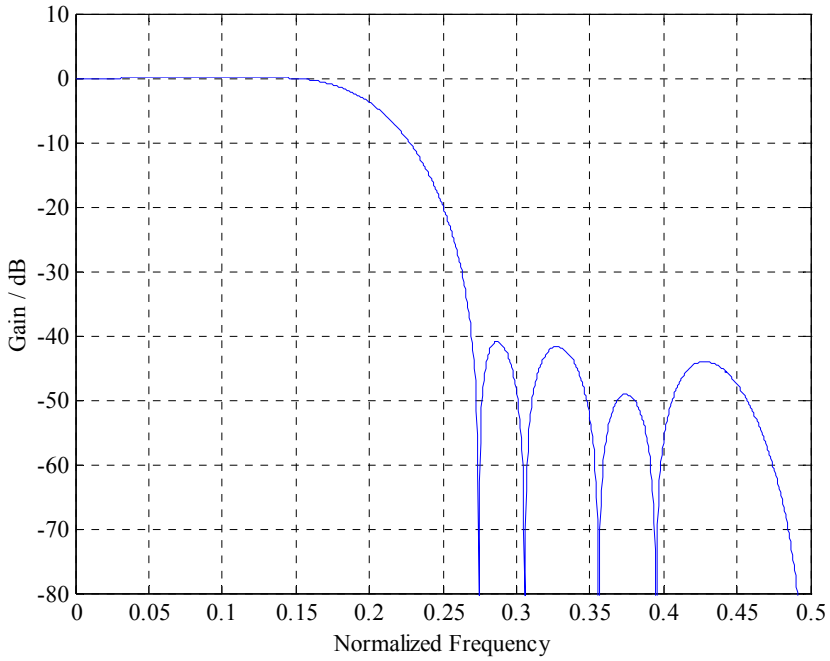


(b) **Overall filter**

Fig. 6. 2 The frequency responses of the two subfilters (a) and overall filter (b).

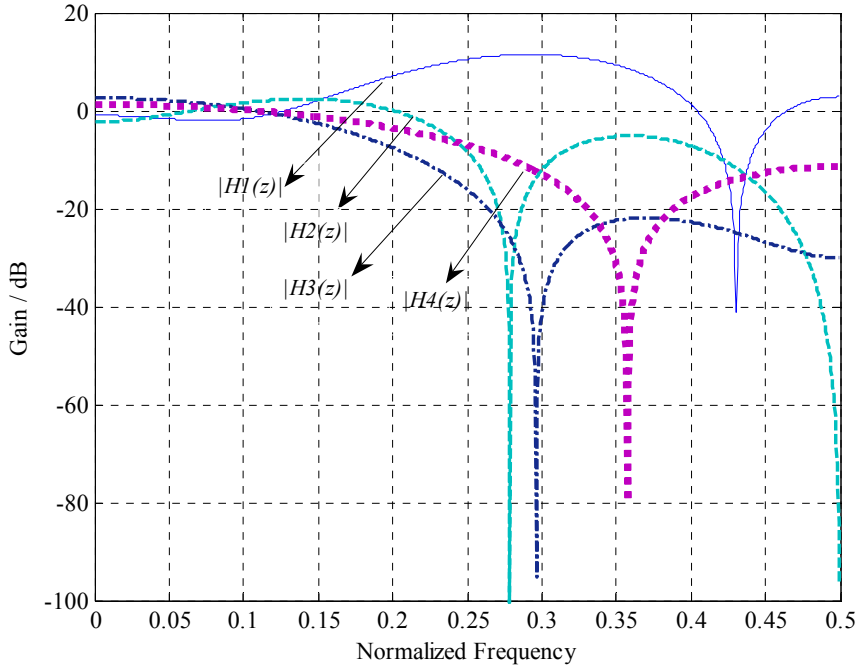


(a) **Three subfilters**

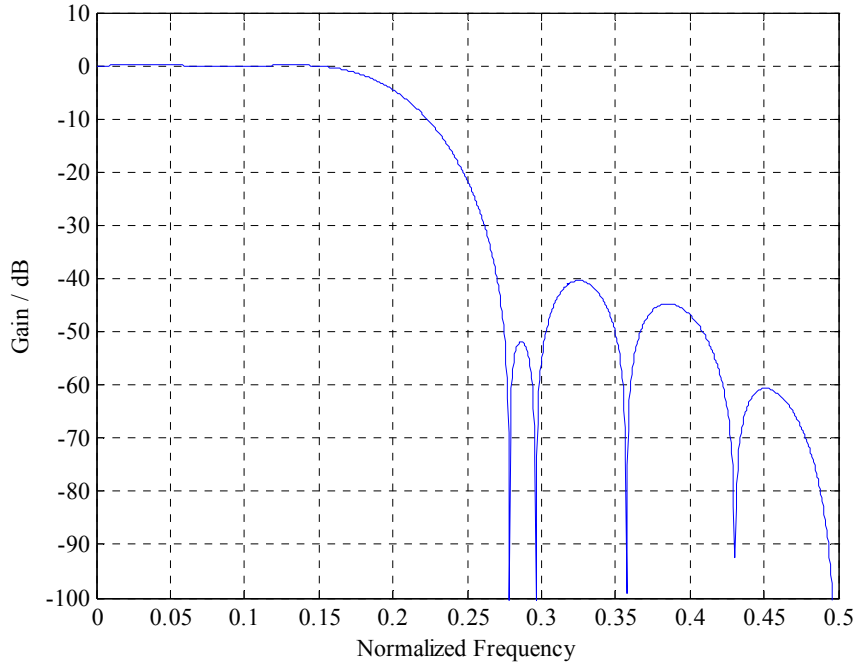


(b) **Overall filter**

Fig. 6. 3 The frequency responses of the three subfilters (a) and overall filter (b).



(a) **Four subfilters**



(b) **Overall filter**

Fig. 6. 4 The frequency responses of the four subfilters (a) and overall filter (b).

Table 6. 2 A comparison among the filters with 2, 3, and 4 subfilters designed by using GA and MGA

No. of factorized filters		2	3	4
Stopband attenuation	GA	40.01 dB	41.54 dB	40.61 dB
	MGA	40.81 dB	40.63 dB	40.38 dB
Lengths	GA	13,9	9,7,7	7,7,5,5
	MGA	13,9	9,7,7	7,7,5,5
No. of bits	GA	6,6	6,4,4	5,4,4,4
	MGA	6,6	4,5,4	5,4,4,3
No. of Function evaluations	GA	267,100	213,900	271,400
	MGA	286,20	21,440	143,60

6.5 A Comparison among Proposed AGA, OSGA, GSA, AGSTA, and MGA

In Chapters 2 - 6 of the thesis, several novel algorithms are proposed for the design of FIR filters with SPoT coefficients. The AGA with adaptive population size and genetic operation probability, which offers enhancement in terms of design performance and efficiency, is proposed in Chapter 2. In Chapters 3 and 4, two new hybrid genetic algorithms named as OSGA and GSA that integrate the GA with an oscillation search algorithm and with the SA, respectively, are developed for the design of FRM FIR filters. In Chapter 5, an efficient hybrid algorithm, AGSTA adopting the GA, SA, and TS techniques, is proposed for the global optimization of FIR filters with SPoT coefficients.

In this chapter, a modified MGA is proposed to considerably increase the convergence speed.

These algorithms are developed for different design problems and requirements concerning FIR digital filter design with discrete coefficients. Therefore, we cannot simply conclude that one is better than another or which one is the best. However, to gain an overall impression, the low-pass linear phase FIR filter specified in Section 5.3 is designed by using all proposed algorithms, i.e. AGA, OSGA, GSA, AGSTA, and MGA. The specifications are repeated here, i.e. the normalized passband and stopband edges of the filter are 0.15 and 0.22, respectively; the ratio in passband and stopband equals one. The filter is designed with different lengths ranging from 31 to 41. The corresponding data from other methods can refer to Section 5.3. Ten independent runs of each algorithm are performed for every design to check the optimization stability. The comparison is based on the average performance. The following parameters are recorded for each design to assess the convergence performance: the mean number of function evaluations in 10 runs and the mean NPR of the best filters found in the 10 runs. The results from GA, AGA, OSGA, GSA, AGSTA, and MGA are listed in Table 6.3. It can be seen from this table that except MGA, all proposed algorithms can achieve better performance than the GA. In this example, the mean values of NPR achieved by the AGSTA are smallest, while the convergence speed of MGA is highest. The average performance of the OSGA is better than that of the GSA. The OS and SA both belong to the class of local search. The OS algorithm works based on the rule of filter coefficients, while the SA algorithm works in a stochastic model. Thus, when they are applied in the design of direct form FIR

filters, the optimization performance of the OSGA is better. It can be seen that the MGA can achieve slightly better solutions than the GA when the filter length is relative short, i.e. shorter than 35. Although the solutions from the MGA are not as good as those from other algorithms, considerable savings on computational effort can be achieved by using the MGA. Thus, the MGA will be more competent when the computational cost is the most important design requirement.

Table 6. 3 A comparison among the designs from the GA, AGA, OSGA, GSA, AGSTA, and MGA over 10 runs

Filter Length *N*	GA		AGA		OSGA	
	Mean NPR (dB)	Mean Function Evaluations	Mean NPR (dB)	Mean Function Evaluations	Mean NPR (dB)	Mean Function Evaluations
31	-39.79	387,000	-40.14	322,473	-40.80	285,710
33	-39.84	429,500	-40.37	335,294	-40.92	308,584
35	-40.85	441,000	-41.10	361,821	-42.06	332,680
37	-41.91	478,500	-43.08	412,769	-44.13	379,276
39	-42.03	534,500	-43.40	464,288	-44.84	423,872
41	-43.46	569,000	-45.95	502,396	-46.71	471,952
Filter Length *N*	GSA		AGSTA		MGA	
	Mean NPR (dB)	Mean Function Evaluations	Mean NPR (dB)	Mean Function Evaluations	Mean NPR (dB)	Mean Function Evaluations
31	-40.68	301,796	-41.17	219,736	-39.81	28,632
33	-40.79	329,837	-41.43	224,745	-39.88	29,180
35	-41.92	340,744	-42.59	245,529	-40.64	31,024
37	-43.82	396,157	-45.02	274,372	-41.47	31,940
39	-44.53	458,521	-45.27	290,530	-41.65	32,980
41	-46.23	489,285	-48.12	312,949	-43.07	33,089

6.6 Conclusion

In this chapter, a modified micro-genetic algorithm is presented for the design of digital FIR filters with SPoT coefficient values. The MGA overcomes the drawback of long computation time by utilizing a small population. To avoid entrapment in local optima, the MGA is modified to include a strategy that varies the probabilities of crossover and mutation during the evolution. The modified MGA can be successfully applied in the design of discrete valued filters in both direct form and cascade form. Compared with the conventional GA, the modified MGA speeds up the optimization process significantly. To gain an overall impression, a comparison among the AGA, OSGA, GSA, AGSTA and MGA proposed in Chapters 2-6 is presented.

Chapter 7

Conclusion

7.1 Summary

It is a well known fact that the coefficients of an FIR filter can be quantized into sum or difference of signed powers-of-two values leading to a so-called multiplication-free hardware implementation where multiplications can be carried out by simply using adders and data shifters. The quantization of each coefficient into SPoT space is a complicated process requiring excessive computer resources. It may have multiple feasible regions and multiple locally optimal points. It is impractical to exhaustively enumerate all of the possible solutions and pick the best one, even on a fast computer. AI based algorithms are good candidatures to solve global optimization problems. In this thesis, design techniques for the SPoT based FIR filters are proposed. Several novel methods based on AI technique are proposed for different design requirements and structures.

In Chapter 2, the methods are proposed for the design of low power high-speed FIR digital filters, where a long filter is factorized into several cascaded short filters each with SPoT coefficient values. The coefficients of all subfilters are quantized into SPoT values

simultaneously with the help of genetic encoding scheme. Since the information which is related to hardware requirement is affiliated to the fitness function as an optimization criterion, the proposed methods reduce the hardware cost significantly. Truncating several of the least significant bits from the outputs of intermediate stages in cascaded subfilters is introduced to reduce the word length of output signal from the overall filter. An empirical equation for truncation margin is given. An adaptive genetic algorithm, AGA with varying population size and varying probabilities of genetic operations, is proposed to overcome the drawbacks of the conventional GA. Significant savings on hardware cost can be achieved when the filters are optimized using the proposed methods. The FRM technique is one of the most computationally efficient ways for the synthesis of arbitrary bandwidth sharp linear phase FIR digital filters. In one stage FRM structure, there are one bandedge shaping filter and two masking filters. The simultaneous quantization of at least three subfilters is very complicated. To address the problem, a novel hybrid algorithm, OSGA, is proposed in Chapter 3, which is formed by integrating an OS algorithm with the GA. The OS algorithm is developed according to the properties of filter coefficients and used to improve the convergence performance of the GA. The example shows that the coefficient word lengths of subfilters can be significantly reduced. The computational cost is much less than those required by the GA.

It is possible to increase the speed of an FRM filter with a modified FRM structure and SPoT techniques, where the long bandedge shaping filter is replaced by several cascaded short filters each with SPoT coefficients. In Chapter 4, an effective hybrid algorithm, GSA, is proposed for the joint design of all subfilters in the modified FRM structure with

SPoT coefficients. In the GSA, the GA handles the overall optimization while the SA is applied to help escape from local optima and to prevent the premature convergence of the GA. It is shown by means of example that the proposed methods can significantly reduce the word lengths of the subfilters in comparison with linear programming.

In Chapter 5, integrating the AGA, SA and TS algorithms leads to a hybrid genetic scheme. The AGA is used as the basis of the hybrid algorithm. The SA algorithm comes to the picture when it is indicated that the AGA may get stuck in a local optimum. The TS technique is used to improve the convergence speed by reducing the search space according to the properties of filters coefficients. Simulation study shows that the normalized peak ripples of filters can be largely reduced with the help of the proposed algorithm. Compared with the GA, the AGSTA achieves not only the improved solution quality but also the considerable reduction of computational effort.

To speed up the design process and overcome the drawbacks of low convergence speed of the conventional GA, a modified micro-genetic algorithm with very small population is proposed in Chapter 6. To improve the high probability of premature convergence usually associated with small population, the MGA is modified to adjust the probabilities of crossover and mutation during the evolution. It was illustrated by examples that the modified MGA is about several times faster than the conventional when it is applied for the design of discrete valued filters in both direct form and cascade form.

7.2 Future Work

Although many optimization techniques have been proposed for the design of FIR filters with SPoT coefficients in the thesis, there are still rooms to improve the design performance from the following aspects.

For optimization algorithms:

1. Develop encoding schemes to reduce the chromosome length as well as the solution space, resulting in high convergence speed. Instead of using fixed chromosome length, variable lengths can be considered.
2. Develop new mechanism of genetic operations to improve convergence performance by considering the properties of the desired filters.

For FIR filter design:

1. In this thesis, the OS algorithm is developed based on the properties of filter coefficients, where the optimization order of one subfilter is from high coefficient sensitivity to low sensitivity. For FRM filters, the sensitivity of the coefficients has been less studied. It is expected that the OS optimization order based on sensitivity would not separate the subfilters. It relays on clearly understanding FRM filter coefficient sensitivity, which will certainly improve the design.
2. How to arrange the subfilters, i.e. the order of subfilters, is important in the design of cascade form filters. Although some researchers have contribute much in this topic. It is still an open issue.
3. The mathematical analysis on the quantization noise with truncation in the design of cascade form FIR filters with discrete coefficients is worth being investigated.

Bibliography

[1] Y. C. Lim and SR Parker, “FIR filter design over a discrete powers-of-two coefficient space,” *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-31, pp. 583-591, Jun. 1983.

[2] Y. C. Lim and B. Liu, “Design of cascade form FIR filters with discrete valued coefficients,” *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-36, pp. 1735-1739, Nov. 1988.

[3] D. C. Munson Jr., “On finite wordlength FIR filter design,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 329, Apr. 1981.

[4] D. M. Kodek,”Design of optimal finite wordlength FIR digital filters using linear programming techniques,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 304-307, Jun. 1980.

[5] Y. C. Lim, “Design of discrete-coefficient-value linear phase FIR filter with optimum normalized peak ripple magnitude,” *IEEE Trans. Circuits Syst.*, vol. CAS-37, pp. 1480-1486, Dec. 1990.

[6] Q. F. Zhao and Y. Tadokoro, “A simple design of FIR filters with powers-of-two coefficients,” *IEEE Trans. Circuits Syst.*, vol. CAS-35, no. 5, pp. 566-570, May 1988.

[7] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Trans. Circuits Syst.*, vol. CAS-36, no. 7, pp. 1044-1047, Jul. 1989.

[8] D. Li, Y. C. Lim, Y. Lian, J. Song, "A polynomial-time algorithm for designing FIR filters with power-of-two coefficients," *IEEE Trans. Signal Processing*, vol. 50, no. 8, pp. 1935-1941, Aug. 2002.

[9] E. J. Diethorn and D. C. Munson Jr., "Finite word length FIR digital filter design using simulated annealing," *Proc. of the IEEE International SymposiumOn Circuits and System*s, San Jose California, May 1986.

[10] N. Benvenuto, M. Marchesi, and A. Uncini, "Application of simulated annealing for the design of special digital filters," *IEEE Trans. Signal Processing*, vol. 40, pp. 323-332, Feb. 1992.

[11] A. Lee, M. Ahmadi, G. A. Jullien, W. C. Miller and R. S. Lashkari, "Digital filter design using genetic algorithm," *Proc. of the IEEE Conference on Advances in Digital Filtering and Signal Processing*, pp. 34-38, Victoria, BC, Jun. 1998.

[12] A. Fuller, B. Nowrouzian, and F. Ashrafzadeh, "Optimization of FIR digital filters over the canonical signed-digit coefficient space using genetic algorithms," *Proc. of the IEEE Midwest Symposium on Circuits and Systems*, pp. 456-469, Aug. 1998.

[13] Y. J. Yu, Y. C. Lim, "Genetic algorithm approach for the optimization of multiplierless sub-filters generated by the frequency-response masking technique," the *IEEE 9th International Conference on Electronics Circuits and Systems*, vol. 3, pp.1163-1166, Dubrovnik, Croatia, Sept. 15-18, 2002.

[14] P. Gentili, F. Piazza, and A. Uncini, "Efficient Genetic Algorithm Design for Power-of-Two FIR Filter," *Proc. Of the IEEE International Conference on Acoustic Speech and Signal Processing*, pp. 1268-1271, Detroit, USA, May 7-12, 1995.

[15] Y. J. Yu and Y. C. Lim, "New Natural Selection Process and Chromosome Encoding for the Design of Multiplierless Lattice QMF Using Genetic Algorithm," *Proc. of the IEEE 8th IEEE International Conference on Electronics, Circuits and Systems*, vol.3, pp. 1273 -1276, Malta, Sept. 2-5, 2001.

[16] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, 1975.

[17] Y. C. Lim, R. Yang, and B. Liu, "The Design of Cascaded FIR Filters," *IEEE International Symposium on Circuits and Syst.*,vol. 2, pp. 181-184, Piscataway, NJ, Jun. 1996.

[18] L. M. Smith and M. E. Henderson Jr., "Roundoff noise reduction in cascade realizations of FIR digital filters", *IEEE Trans. Signal Processing*, vol. 48, no. 4, pp.1196-1200, Apr. 2000.

[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671-680, May 1983.

[20] Y. Lian and Y. C. Lim, "Zeros of linear phase FIR filter with piecewise constant frequency response," *Electron. Lett.*, vol. 28, pp. 203–204, Jan.1992.

[21] S. Lin, "Computer solutions of the traveling salesman problem," *Bell syst. Tech. J.*, vol. 44, pp. 2245-2269, Dec. 1965.

[22] W. H. Press, S. A. Teukolsky, W.T.Vettering, and B. P. Flannery, *Numerical Recipes in FORTRAN*, 2nd ed. New York: Cambridge Univ. Press, 1992, ch. 10.

[23] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey," *IEEE Computer*, vol. 27(6), pp.17-26, July 1994.

[24] K. A. Dejong, *An analysis of the behavior of a class of genetic adaptive systems*, Ph.D. dissertation, University of Michigan (1975).

[25] K. A. Dejong, "Adaptive system design: a genetic approach," *IEEE Trans. Syst., Man, and Cybernitic*s, vol. 10, no. 9, pp. 566-574, Sep. 1980.

[26] D. E. Goldberg, *Genetic algorithms in Search, Optimization and Machine Learning, Reading*, MA: Addison Wesley. 1989

[27] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. Syst., Man, and Cybernietic*s, vol. SMC-16, no. 1, pp. 122-128, Jan./Feb. 1986.

[28] J. D. Schaffer et. al., "A study of control parameters affecting online performance of genetic algorithms for function optimization," *Proc. the IEEE Third International Conference on Genetic algorithms*, pp. 51-60, San Mateo CA: Morgan Kaufman, Jun. 1989.

[29] A. E. Eiben, R. Hinterding and Z. Michalewicz, "Parameter Control in Evolutionary Algorithms," *IEEE Trans. Evolutionary Computation*, vol. 3. no. 2, pp 124-140, Jul. 1999.

[30] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," *IEEE Trans. Syst., Man and Cybernetics*, vol. 24, no. 4, Apr., 1994.

[31] J. D. Schaffer and A. Morishma, “An adaptive crossover mechanism for genetic algorithms,” *Proc. the Second International Conference on Genetic Algorithms,* pp. 36-40, 1987.

[32] T. C. Fogarty, ”Varying the probability of mutation in genetic algorithms,” *Proc. the Third International Conference on Genetic Algorithms*, pp. 104-109, 1989.

[33] L. Davis, “Adapting operator probabilities in genetic algorithms,” *Proc. the Third International Conference. of Genetic Algorithms,* pp. 61-69,1989.

[34] K. S. Tang, K. F. Man, S. Kwong and O. He, “Genetic Algorithms and their applications,” *IEEE Signal Processing Magazin*e, pp. 22 - 37, November 1996.

[35] Y. C. Lim, R. Yang, D.N. Li and J.J. Song, “Signed power-of-two term allocation scheme for the design of digital filters,” *IEEE Trans. Circuits and Syst., II*, vol. 46, pp. 577-84, May, 1999.

[36] Y. C. Lim, “Frequency-response masking approach for the synthesis of sharp linear phase digital filters,” *IEEE Trans. Circuits and Syst.,* vol. CAS-33, pp. 357-364, Apr. 1986.

[37] Y. C. Lim and Y. Lian, “The optimum design of 1-D and 2-D FIR filters using the frequency response masking technique,” *IEEE Trans. Circuits and Syst., II*, vol. CAS - 40, pp. 88-95, Feb. 1993.

[38] T. Saramaki and Y. C. Lim, “Use of the remez algorithm for designing FRM based FIR filters,” *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.77-97, Apr. 2003.

[39] Y. C. Lim and Y. Lian, "Frequency response masking approach for digital filter design: complexity reduction via masking filter factorization," *IEEE Trans. Circuits Syst., II*, vol. 41, pp. 518-525, Aug. 1994.

[40] Y. Lian and Y. C. Lim, "Reducing the complexity of frequency-response masking filters using half band filters," *Journal of Signal Processing*, vol. 42, no. 3, pp. 227-230, March 1995.

[41] Y. Lian, L. Zhang and C. C. Ko, "An improved frequency response masking approach designing sharp FIR filters," *Journal of Signal Processing*, vol. 81, pp. 2573-2581, Dec. 2001.

[42] Y. Lian, "Complexity reduction for frequency response masking based FIR filters using prefilter-equalizer technique," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.137-155, Apr. 2003.

[43] Y. Lian and C. Z. Yang, "Complexity reduction by decoupling the masking filters from bandedge shaping filter in frequency response masking technique," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.115-135, Apr. 2003.

[44] Luiz C. R. de Barcellors, Sergio L. Netto, and Paulo S. R. Diniz, "Optimization of FRM filters using the WLS-Chebyshev approach," *Journal of Circuits, Systems and Signal Processin*g, vol. 22, no.2, pp.99-113, Apr. 2003.

[45] H. Johansson and T. Saramaki, "Two-channel FIR filter bank utilizing the FRM approach," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.157-191, Apr. 2003.

[46] Miguel B. Furtado Jr., Paulo S. R. Diniz, and Sergio L. netto, "Optimized prototype filter based on the FRM approach for cosine-modulated filter banks," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.193-209, Apr. 2003.

[47] Y. C. Lim, Y. J. Yu, H. Q. Zheng, and S. W. Foo, "FPGA Implementation of digital filters synthesized using the FRM technique," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.211-217, Apr. 2003.

[48] O. Gustafsson, H. Johansson, and L. Wanhammar, "Single filter frequency masking high-speed recursive digital filters," *Journal of Circuits, Systems and Signal Processing*, vol. 22, no.2, pp.219-238, Apr. 2003.

[49] T. Saramaki, and H. Johansson, "Optimization of FIR filters using the frequency response masking approach," *Proc. of the IEEEInternational Conference on Circuits and Systems*, vol. 2, pp.177-180, May 2001.

[50] Y. J. Yu, Y. C. Lim, "FRM based FIR filter design-the WLS approach," *Proc. of the IEEEIntern. Symposium on Circuits and Systems*, vol. 3, pp. 221-224, Arizona, USA, May 26-29, 2002.

[51] W. S. Lu, and T. Hinamoto, "Optimal design of frequency-response masking filters using semidefinite programming," *IEEE Trans. Circuits Syst., I*, vol. 50, pp.557-568, April 2003.

[52] W. S. Lu and T. Hinamoto, "Optimal design of IIR frequency-response-masking filters using second-order cone programming," *IEEE Trans. Circuits Syst., I*, vol. 50, pp. 1401-1412, Nov. 2003.

[53] W. S. Lu and T. Hinamoto, "Improved design of frequency-response-masking filters using enhanced sequential quadratic programming," *Proc. of the IEEE International Symposium on Circuits and Systems*, vol. 5, pp.V-528 - V-531, Vancouver, Canada, May 23-26, 2004.

[54] O. Gustafsson, H. Johansson, and L. Wanhammar, "MILP design of frequency-response masking FIR filters with few SPT terms," *Proc.of theIEEE International Symposium On Control, Communications, Signal Processing*, Hammamet, Tunisia, March 21-24, 2004.

[55] Y. Lian, "Design of discrete valued coefficient FIR filters using frequency-response masking technique," *Proc. of IEEE the 6th Conference on Electronics, Circuits & Systems, Paphos*, Cyprus, Sept., 1999.

[56] Y. Lian, "FPGA implementation of high speed multiplierless frequency response masking FIR filters," *Proc. Of the IEEE Conference on Signal Processing Systems, Design and Implementation*, pp. 317 -325, USA, Oct. 11-13, 2000.

[57] L. Cen and Y. Lian, "Low-Power implementation of frequency response masking based FIR filters," *Proc. of the IEEE Fourth International Conference on Information, Communications & Signal Processing and Fourth Pacific-Rim Conference on Multimedia*, vol. 3 , 15-18, pp. 1898-1902, Singapore, Dec. 15-18, 2003.

[58] L. Cen and Y. Lian, "High speed frequency response masking filter design using genetic algorithm," *Proc. of the IEEE International Conference on Neural Networks & Signal Processing*, vol. 1, 14-17, pp. 735 – 739, Nanjing, China, Dec. 14-17, 2003.

[59] C. C. Lo and W. H. Chang, "A multiobjective hybrid genetic algorithm for the capacitated multipoint network design problem", *IEEE Trans. on Systems, Man and Cybernetics*, *II*, vol. 30, no. 3, pp. 461 – 470, Jun. 2000.

[60] S. A. Kazarlis, S. E. Papadakis, J. B. Theocharis, and V. Petridis, "Microgenetic Algorithms as Generalized Hill-Climbing Operators for GA Optimization," *IEEE Trans. Evolutionary Computation*, vol. 5, no. 3, Jun. 2001.

[61] A.M. Sabatini, "A hybrid genetic algorithm for estimating the optimal time scale of linear systems approximations using Laguerre models", *IEEE Trans. on Automatic Control,* pp. 1007 – 1011, vol. 45, no. 5, May 2000

[62] D. W. Gong, Y. Zhou, X. J. Guo, X. P.Ma and M. Li, "Study on An Adaptive Tabu Search Genetic Algorithm," *Proc. of the IEEE 4th World Congress on Intelligent Control and Automation*, pp. 3063-3065, Shanghai, P.R. China, Jun. 2002.

[63] K. Krishnakumar, "Micro-genetic algorithms for stationary and non-stationary function optimization," *SPIE: Intelligent Control and Adaptive Systems*, vol. 1196, pp. 289-296, 1989.

[64] S. K. Mitra and J.F. Kaiser, *Handbook for Digital Signal Processing,* John Wiley & Sons, 1993, ch. 4, 8.

[65] A. V. Oppenheim and R.W. Schafer, *Discrete-Time Signal Processing,* Engle-wood Cliffs, NJ: Prentice-Hall, 1989, ch. 7.

[66] L.R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall, 1975, ch. 3.

[67] T. Saram*ä*aki, “A systematic technique for designing highly selective multiplier-free FIR filters," *Proc. The IEEE International Conference Circuits Syst.,* Singapore, pp. 484-487, 1991.

[68] Y. C. Lim and A. G. Constantinides, “Linear phase FIR digital filter without multipliers," *Proc. IEEE International. Symposium Circuits and Syst.,* pp. 185-189, 1979

[69] O. Gustafsson, H. Johansson and L. Wanhammar, “An MILP approach for the design of linear-phase FIR filters with minimum number of signed-power- of-two terms," *Proc. European Conf. Circuit Theory Design,* vol. 2, pp. 217-220, Espoo, Finland, Aug. 2001.

[70] J. H. Lee, C. K. Chen and Y. C. Lim, “Design of discrete coefficient FIR digital filters with arbitary amplitude and phase responses," *IEEE Trans. Circuits Syst.,* vol. 40, pp. 444-448, July, 1993.

[71] -------,“Discrete coefficient FIR digital filter design based upon an LMS criteria", *IEEE Trans. Circuits and Systems,* vol. CAS-30, pp.723-739, Oct. 1983.

[72] D. Ait-Boudaoud and R. Cemes, “Modified sensitivity criterion for the design of powers-of-two FIR filters," *Electron. Lett.,* vol. 29, pp. 1467-1469, Aug. 1993.

[73] H. Shaffeu, M. M. Jones, H.D. Griffiths and J.T. Taylor, “Improved design procedure for multiplierless FIR digital filters," *Electron. Lett.,* vol. 27, pp. 1142-1144, June 20, 1991.

[74] C. L. Chen, A. N. Willson Jr., “A Trellis search algorithm for the design of FIR fiters with signed-powers-of-two coefficients," *IEEE Trans. circuits syst. II,* vol. 46, pp. 29-39, Jan. 1999.

[75] J. W. Tukey, *Exploratory data analysis, Reading*, MA: Addison-Wesley, 1977.

[76] J. F. Kaiser, and R. W. Hamming, “Sharpening the response of a symmetrical nonrecursive filter by multiple use of the same filter,” *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-25, no.5, pp. 415-422, Oct. 1977.

[77] C. K Chen, and J. H. Lee, “Design of sharp-cutoff FIR digital filters with prescribed constant group delay,” *IEEE Trans. Circuits and Systems, II: Analog and digital signal processing*, vol. 43, no. 1, Jan. 1996.