MINIMIZATION OF HAMMING DISTANCE IN FIR FILTERS BY USING STEEPEST DESCENT METHOD & EVOLUTIONARY PROGRAMMING

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
In
Power Systems & Electric Drives

Thapar University, Patiala

By:
PRINCE JINDAL
(Regn. No. 80741017)

Under the supervision of:
Mr. Parag Nijhawan
Sr. Lecturer (EIED)

JULY 2009

ELECTRICAL & INSTRUMENTATION ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

CERTIFICATE

I hereby certify that the work which is being presented in this thesis entitled "**Minimization Of Hamming Distance Of FIR Filters By Using Steepest Decent Method & Evolutionary Programming",** in partial fulfillment of the requirements for the award of degree of master of engineering in power system & electric drives submitted in electrical & instrumentation engineering department of Thapar University Patiala is an authentic record of my own work carried out under the supervision of Mr. Parag Nijhawan, Sr. lecturer, EIED.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university

Prince Jindal 15/07/09

(PRINCE JINDAL)

This is to certify that the above statement made by the candidate is correct and true to best of my knowledge.

Parag 15/07/09

(PARAG NIJHAWAN)
SR. LECTURER
Electrical & Instrumentation Engg. Department
Thapar University Patiala

Countersigned by:

S. Ghosh

(DR. S. GHOSH)
Professor & Head
Electrical & Instrumentation Engg. Department
Thapar University
Patiala.

RK Ahare 22/7

(DR. R.K SHARMA)
Dean(Academics Affairs)
Thapar University,
Patiala.

Abstract

With the explosive growth of wireless communication system and portable devices, the power reduction has become a major problem. In applications, such as personal communication systems, and portable storage devices, low power dissipation, hence longer battery lifetime is a must. With the rapid growth of internet and information on demand, handheld wireless terminals are becoming increasingly popular. Many of the communication systems today utilize digital signal processors (DSP) to resolve the transmitted information. Finite Impulse Response (FIR) filters have been and continue to be important building blocks in many digital signal processing (DSP) systems.

Signal switching activity is the major component of power dissipation in CMOS circuits. Hamming distance is a measure of switching activity corresponding to the number of energy consuming transitions in multiplier and accumulate (MAC) of FIR filter while implementing on Digital Signal Processor (DSP).The transition densities of multiplier inputs depend on the Hamming distance between successive filter coefficient values. For a multiplier the power is directly dependent on the transition densities and the probabilities of multiplier inputs. The Hamming distance between consecutive coefficient values and the number of signal toggling in opposite directions thus forms the measure of bus power dissipation. In this thesis the Hamming distance in FIR filters is minimized by minimizing the switching activity using *'Steepest Decent'* and *'Evolutionary Programming'* optimization techniques to reduce the power dissipation and to increase the battery life of portable multimedia devices.

ACKNOWLEDGEMENT

First of all I would like to thank the Almighty, who has always guided me to work on the right path of the life.

I would like to express a deep sense of gratitude and thanks to my thesis guide Mr. Parag Nijhawan, who has been a constant source of inspiration to me throughout the period of this thesis. It was his competent guidance, constant encouragement and critical evaluation that helped me to develop a new insight into my thesis. His calm and professionally impeccable style of handling situations not only steered me through every problem, but also helped me to grow as a matured person. I am also thankful to him for trusting my capabilities to develop this thesis under his guidance

I also extend my warm thanks to Dr. S. Ghosh, Prof. & Head, EIED without whom my thesis would not have been possible.

I am also thankful of all my family members and all my friends for their great help and encouragement in carrying the present work.

PRINCE JINDAL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Nomenclature

CMOS	Complementary Metal Oxide Semiconductor
DSP	Digital Signal Processing
Etot	Sum of errors throughout discrete frequency domain.
HD	Hamming Distance
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
fs	Sampling frequency
fc	Cut-off frequency
GA	Genetic Algorithm
$H(e^{jw})$	Magnitude response of filter
LMS	Lest mean square
Hd	Desired magnitude response of filter
MAC	Multiply and accumulate
MOP	Multiobjective optimization problem
N	Number of filter coefficients
Pdb	Passband ripples
Sdb	Stopband attenuation
*x**	Optimal point
α	Switching activity
Vdd	Supply Voltage
ω	Angular frequency
wi	Weighting factor for multiobjective optimization

CHAPTER 1
INTRODUCTION

An important trend that has been a major driver of the electronics industry in recent years is the growing demand for portable computing and communication devices. This demand has been fueled by the quality of life and business productivity improvements that have been provided by these devices. There has been a tremendous interest in laptop computers, personal digital assistants, mobile phones, and pagers. This is only the beginning, however, as there are more sophisticated devices on the horizon that will provide increasingly sophisticated capabilities and features. In future mobile, wireless terminals provide users with ubiquitous and untethered access to multimedia content and computing services available from a high-bandwidth backbone network of computers.

A portable multimedia terminal must provide two fundamental capabilities: the ability to process multimedia information and the ability to communicate that information through various wired and/or wireless communications channels. Speech, audio, video, and graphics are examples of the types of data that are processed by a typical multimedia terminal. The technology that provides the underlying algorithms to process these data types is Digital Signal Processing (DSP). Digital signal processing is also the technology that is applied to process the signals that are used to communicate information over a wired or wireless communication channel. Improvements in computational performance provided by advances in integrated circuit fabrication technology allow the use of more and more sophisticated signal processing techniques that allow greater functionality and performance and richer modes of communication in portable multimedia terminals.

A major problem associated with increases in the processing power and the sophistication of signal processing algorithms is the increasing levels of power dissipation. A mobile terminal is typically powered by batteries, a limited source of energy. For a portable device to be useful, it must have a reasonable amount of run time before the batteries run out and need to be recharged. Another problem with high levels of power dissipation is the cost of packaging and cooling. Low-power integrated circuits can be placed in inexpensive and compact packages. High-power devices, on the other

hand, require expensive and bulky packages and cooling mechanisms. High levels of power dissipation also mean high operating temperatures that adversely affect the reliability of an integrated circuit.

Power dissipation is not a new problem. In the middle of 1980s, designers faced the same problem. The solution then was to switch from NMOS technology, which suffered from static power dissipation, to CMOS technology. CMOS was far more energy-efficient than NMOS and was the most effective and economically viable way to build more and more powerful microprocessors. In fact, CMOS was so energy efficient, that power became an afterthought once again. But in recent years, increasing levels of computing performance have made power an important and challenging problem once again, and this time, there is no magic technological solution to make it disappear.

To provide the required computing power and to increase the energy efficiency of signal processing circuits, designers have developed numerous design techniques that can be applied in custom, application-specific integrated circuits. While this approach has been successful in increasing energy efficiency, it suffers from the drawback that the resulting devices can only provide the limited functionality that they were designed to provide, i.e., they are not very flexible. In reality, however, a variety of multimedia data types and services, modes of communication and associated standards are in use, and it is highly desirable to have devices that can deal with this variety. Therefore, it is highly desirable that flexible, programmable components be used to implement the processing functions required in a modern computing/communication device.

1.1. Power Dissipation – Factors and Measures

The motivation for low power electronics has stemmed from three reasonably distinct classes of requirement

1. The earliest and most demanding of these is for portable battery operated equipment that is sufficiently small in size and weight and long in operating life. The goal is to satisfy the user of communication and portable multimedia devices.
2. The most recent need is for ever increasing packing density in order to further enhance the speed of high performance systems, which imposes severe restrictions on power dissipation density.

Viewed together, these three classes of need appear to encompass a substantial majority of current applications of electronic equipment. Low power electronics has become the mainstream of the effort to achieve gigs scale integration (GSI).

1.1.1 Sources of Power Consumption

In CMOS circuits the total power dissipation can be obtained from the sum of the three components summarize as:

$$P_{avg} = P_{dynamic} + P_{static} + P_{leakage} \tag{1.1}$$

There are two major sources of power dissipation

1.1.2 Dynamic Dissipation

Dynamic dissipation is due to switching transient (short circuit current) and charging and discharging of load capacitance. The dynamic power may further divided into two components

$$P_{dynamic} = P_{switching} + P_{short-circuit} \tag{1.2}$$

The switching component, $P_{switching}$, arises when the capacitive load, C_L, of a CMOS circuit is charged through PMOS transistors to make a voltage transition from 0 to the high voltage level, which is usually the supply, V_{dd}.

For an inverter circuit as shown in figure 3.3, the power dissipated because of a 0 to 1 transition can be determined from the product $V_{dd}*I_C$ where I_C is the transient current drawn from the supply.

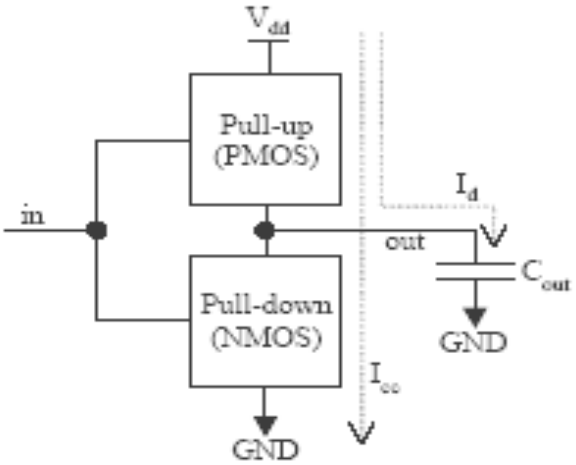

Figure 1.1 CMOS Inverter

1.1.3 Static Dissipation

Static dissipation is due to the leakage current or other current drawn continuously from the power supply. Ideally, CMOS circuits dissipate no static (DC) power since in the steady state there is no direct path from V_{dd} to ground. Of course, this scenario can never be realized in practice since in reality the MOS transistor is not a perfect switch. Static power dissipation, $P_{leakage}$, stems from the leakage current, $I_{leakage}$, which can arise from substrate injection and sub threshold effects and primarily determined by fabrication technology considerations. This current is typically in the *nA* region and contributes little to the overall power consumption. However, in future deep submicron technologies, leakage power will become a problem.

1.2 Low Power Design Methodologies

The design variables that effect the dynamic power consumption of a CMOS circuit have been identified. Now the power minimization problem from various design aspects will be investigated that effect power dissipation: technology, circuit techniques, architectures and algorithms.

1.2.1 Technology

An optimization that could be done at this level is driven by voltage scaling. It is necessary to scale supply voltage for a quadratic improvement in energy per transition. Unfortunately, we pay a speed penalty for a V_{dd} reduction with delays increasing, as V_{dd} approaches the threshold voltage of the devices. The simple first order relationship between V_{dd} and gate delay, t_d for a CMOS gate is given in (1.1),

$$t_d \alpha \frac{1}{(V_{dd} - V_t)^2} \tag{1.3}$$

The objective is to reduce power consumption while keeping the throughput of the overall system fixed. Therefore compensation for these delays at low voltages is required.

1.2.2 Circuit techniques

There are a number of options available in choosing the basic circuit approach and topology for implementing various logic and arithmetic functions. Choices between static vs. dynamic implementations, pass-transistor vs. conventional CMOS logic styles, and synchronous vs. asynchronous timing are just some of the options open to the system designer.

1.2.3 Architecture Optimization

As seen in equation (1.1), gate delays increase drastically, when supply voltage approaches the threshold voltage of the MOS transistor. There are two architectural techniques that can improve the speed of the circuit under reduced supply voltage:

(1) Pipelining: It is a powerful transformation of the datapath to reduce the critical path of the system and improve the speed. It involves the insertion of delay elements, flip flops at specific points of a data flow graph of an algorithm/architecture. The speed gained by this transformation can be traded for low power by voltage scaling.

(2) Parallelism: It is similar to pipelining in that it exploits parallelism in a system, however here this is achieved by duplicating hardware in order to perform a number of similar tasks concurrently.

1.2.4 Algorithm

Choosing the algorithm to implement the application at hand represent the most important decision in meeting the power constraints. From the section (1.2.3), it can be deduced that in order to reap the greatest architectural gains, the ability to parallelize an algorithm will be critical, and the basic computation must be optimized, as the basic theme in low power design is voltage reduction.

Therefore, at the algorithmic level, transformations that can be used to increase speed and allow lower voltages are useful. Often these approaches translate into larger silicon area hence the approach has been termed trading area for power.

At the algorithm level, the size and complexity of a given algorithm i.e. operation counts, word lengths and so on determine the activity. If there are several algorithms for a given task, the one with the least number of operations (arithmetic operation, memory access etc.) is generally preferable.

1.3 Digital Signal Processing

Digital Signal Processing (DSP) is the processing of signals by digital means. Historically the origins of signal processing are in Electrical Engineering, and a signal means an electrical signal carried by a wire or telephone line, or by a radio wave. More generally, a signal is a stream of information representing any thing from stock price to data from a remote sensing-satellite. In many cases, the signal is initially analog electrical voltage or current produced by a microphone or some other type of transducer. In some

situation data is already in digital form, such as output from the readout system of a compact disk. Analog signal must be converted into digital form before DSP techniques are applied. An analog electrical voltage signal, for example, is digitized using an integrated electronic circuit (IC) device called analog-to-digital converter (ADC). This generates digital form in the form of binary numbers whose value represents the electrical voltage input to the device.

DSP technology is nowadays commonplace in such devices as mobile phones, multimedia computers, video recorders, CD players, hard disc drive controllers and modems, and will soon replace analog circuitry in TV sets and telephones. An important application of DSP is in signal compression and decompression. In CD systems, for example, the music recorded on the CD is in a compressed form (to increase storage capacity) and must be decompressed for the recorded signal to be reproduced.

1.4 Analog and Digital Filters

Filters are widely employed in signal processing and communication circuits systems in applications such as channel equalization, noise reduction, radar, audio processing, video processing, biomedical signal processing. In a radio receiver, band pass filter or tuner is used to extract the signals in a radio channel. In an audio graphic equalizer the input signal is filtered into a number of sub-bands signals and the gain for each sub-band can be varied manually with a set of controls to change the perceived audio sensation. The Dolby system used is another example of the application of filters. In this case pre-filtering and post-filtering are used to minimize the effect of noise. In hi-fi audio a compensating filter may be included in a pre amplifier to compensate for the non ideal frequency response characteristics of the speakers. Filters are also used to create perceptual audio/visual effect for entertainment and in broadcast studios. The primary functions of the filters are one of the followings:

1) To confine a signal into prescribed frequency band as in low-pass, high-pass and band-pass filters.
2) To decompose a signal into two or more sub-bands as in filter banks, graphic equalizers, sub-band coders, frequency multiplexers.

3) To modify the frequency spectrum of a signal as in telephone channel equalization and audio graphic equalizers.
4) To model the input-output relation of a system such as telecommunication channels, human vocal tract, and music synthesizers.

Depending on the form of the filter equation and structure of implementation, filters may broadly be classified into the following classes:

1) Linear filters versus non linear filters.
2) Adaptive time varying filters versus non adaptive time invariant filters.
3) Recursive versus non recursive filters
4) Direct form, cascade form, parallel form and lattice structures.

In signal processing, the function of a filter is to remove unwanted parts of the signal, such as random noise, or to extract useful parts of the signal, such as the components lying within a certain frequency range. The block diagram shown in fig.1.2 illustrates the basic idea

Figure 1.2 Basic Filtering Operations

There are two main kinds of filter, analog and digital. They are quite different in their physical makeup and in how they work.

An analog filter uses analog electronic circuits made up from components such as resistors, capacitors and op amps to produce the required filtering effect. Such filter circuits are widely used in such applications as noise reduction, video signal enhancement, graphic equalizers in hi-fi systems, and many other areas. There are well established standard techniques for designing an analog filter circuit for a given requirement.

The analog input signal must first be sampled and digitized using an ADC (analog to digital converter). The resulting binary numbers, representing successive sampled values of the input signal, are transferred to the processor, which carries out numerical calculations on them. These calculations typically involve multiplying the input values by constants and adding the products together. If necessary, the results of these calculations, which now represent sampled values of the filtered signal, are output through a DAC (digital to analog converter) to convert the signal back to analog form.

In a digital filter, the signal is represented by a sequence of numbers, rather than a voltage or current. The main advantages of digital filters over the analog filters are

1) A digital filter is programmable, i.e. its operation is determined by a program stored in the processor's memory. This means the digital filter can easily be changed without affecting the circuitry (hardware). An analog filter can only be changed by redesigning the filter circuit.
2) Digital filters are easily designed, tested and implemented on a general-purpose computer or workstation.
3) Unlike their analog counterparts, digital filters can handle low frequency signals accurately. As the speed of DSP technology continues to increase, digital filters are being applied to high frequency signals in the RF (radio frequency) domain, which in the past was the exclusive preserve of analog technology.
4) The characteristics of analog filter circuits (particularly those containing active components) are subject to drift and are dependent on temperature. Digital filters do not suffer from these problems, and so are extremely *stable* with respect both to time and temperature.
5) Digital filters are very much more *versatile* in their ability to process signals in a variety of ways; this includes the ability of some types of digital filter to adapt to changes in the characteristics of the signal.

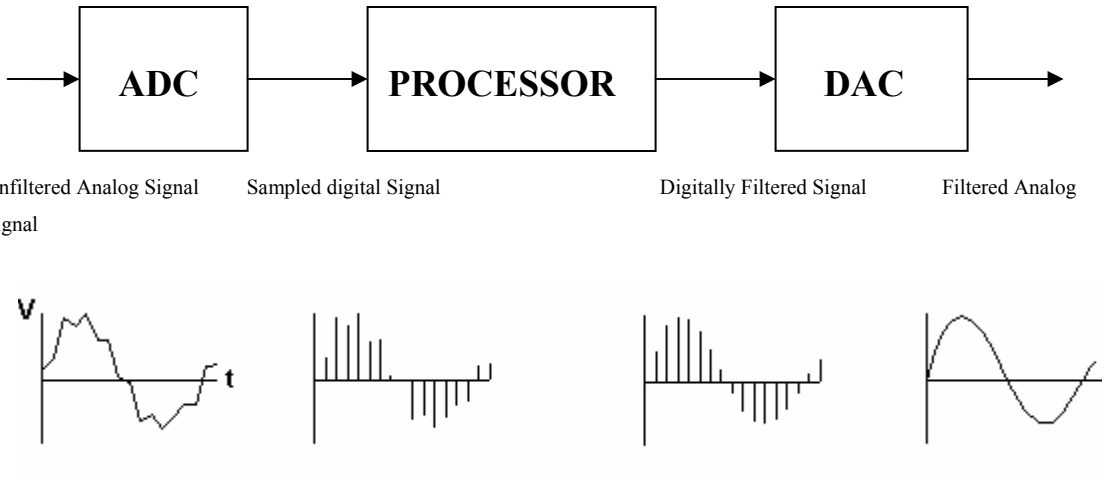

Figure 1.3 Implementation of Filter on Digital Signal Processor.

Fast DSP processors can handle complex combinations of filters in parallel or cascade (series), making the hardware requirements relatively simple and compact in comparison with the equivalent analog circuitry. The block diagram of DSP processor is shown in fig1.3.

1.5 Optimization

Optimization is the act of obtaining the best result under given circumstances. Optimization can be defined as the process of finding the condition that gives the maximum or minimum value of function [48]. If x^* corresponds to the minimum value of function $f(x)$, the same point also corresponds to the maximum value of the function $-f(x)$. Thus optimization can be taken to mean minimization since the maximum of the function can be found by seeking of the negative of the same number. The optimum seeking methods are also known as mathematical programming techniques. These techniques are useful in finding a minimum of a function of a function of several variables under a prescribed set of constraints. Stochastic process techniques can be used to analyze problems described by a set of random variables having known probability distributions and the statistical methods enable one to analyze the experimental data and build empirical models to obtain the most accurate representation of the physical situation.

1.5.1 Statement of an Optimization Problem

An optimization or a mathematical programming problem can be stated as follows

Find X = $[x_1\ x_2\ x_3 \ldots\ldots x_n]^T$

Subjected to the constraints

$g_j(X) \leq 0, \qquad j = 1, 2, \ldots\ldots\ldots\ldots, m$

$h_j(X) = 0 \qquad j = 1, 2, \ldots\ldots\ldots\ldots, p$

Where X is an n-dimensional design vector, $f(X)$ is termed the objective function and $g_j(X)$ and $h_j(X)$ are known as inequality and equality constraints, respectively. The number of design variables n and number of constraints m and/or p need not be related any way. The stated problem is called a constrained optimization problem. Some problems do not involve any constraints and can be stated as

Find $X = [x_1\ x_2\ x_3 \ldots\ldots x_n]^T$ which minimizes $f(X)$ are called unconstrained optimization problems.

1.5.2 Design Vector

Every engineering system or component is defined by a set of quantities some of which are viewed as variables during the design process. Certain quantities are fixed at the outset and these are called preassigned parameters. All the other quantities are treated as variables in the design process and are called design or decision variables x_i, $i = 1, 2, \ldots\ldots, n$. The design variables are collectively represented as a design vector

$X = [x_1\ x_2\ x_3 \ldots\ldots x_n]^T$.

1.5.3 Constraints

In practical design problems, the design variables cannot chosen arbitrary rather, they have to satisfy certain specified functional and other requirements. The restrictions that must be satisfied to produce an acceptable design are collectively called design constraints. Constraints that represent limitations or performance of the system are termed behavior or functional constraints. Constraints that represent physical limitation on design variables are known as geometric constraints or side constraints.

1.5.4 Objective Function

The purpose of the optimization is to choose the best one of many acceptable designs available. Thus a criterion has to be chosen for comparing the different alternative acceptable designs and for selecting the one. The criterion, with respect to which the design is optimized, when expressed as a function of the design variables, is known as objective function. While expressing as a function of the design variables if only one criterion is satisfied the problem is called single objective programming problem. In some situations, there may be more than one criterion to be satisfied simultaneously. An optimization problem involving multiobjective programming problem. With multiple objectives there arises a possibility of conflict, and it is handled by constructing an overall objective function as a linear combination of the conflicting multiple objectives functions. If $f_1(X)$ and $f_2(X)$ denote two objective functions, a new objective function for optimization is constructed as

$f(X) = \alpha_1 f_1(X) + \alpha_2 f_2(X)$

$f(X)$ is a new objective function and α_1 and α_2 are constants whose values indicate the relative importance of one objective function relative to the other.

1.6 Optimization Algorithms

The optimization problems reveal the fact that the formulation of engineering design problems could differ from problem to problem. Certain problems involve linear terms for constraints and objective functions but certain other involve non linear for them. In some problems, the terms are not explicit functions of the design variables. Unfortunately there does not exist a single optimization algorithm which will work in all optimization problems equally efficiently. Some algorithms perform better on one problem but may perform poorly on other problems. That is why the optimization literature contains a large number of algorithms, suitable to solve a particular type of problem. The choice of a suitable algorithm for an optimization problem is, to a large extent, dependent on the user's experience in solving similar problems. The optimization algorithms can be classified into a number of groups, which are briefly discussed as

1.6.1 Single variable optimization algorithms

These algorithms optimize only single objective. These algorithms provide a good understanding of the properties of maximum and minimum points in a function and how optimization algorithms works iteratively to find the optimum point in a problem. These

algorithms are classified into two categories, direct methods and gradient based methods. Direct methods do not use any derivative information of the objective function; only objective function values are used to guide the search process. However gradient based methods use derivative information of first or second order to guide the search process. Optimization algorithms usually contain more than one design variables, single variable algorithms are mainly used as unidirectional search methods in a multivariable optimization algorithms.

1.6.2 Multi Variable optimization algorithms

These algorithms optimize more than one objective. These algorithms demonstrate how the search for the optimum point progresses in multiple dimensions. Depending on whether the gradient information is used or not used, these algorithms are also classified into direct and gradient based techniques.

1.6.3 Constrained Optimization Algorithms

These algorithms use the single variable and multi variable optimization algorithms repeatedly and simultaneously maintain the search effort inside the feasible search region. These algorithms are mostly used in engineering optimization problems

1.6.4 Specialized Optimization Algorithms

There exist a number of structured algorithms, which are ideal for only a certain class of optimization problems. Two of these algorithms integer programming and geometric programming are often used in engineering design. Integer programming methods can solve optimization problems with integer design variables. Geometric programming methods solve optimization problems with objective functions and constraints written in a special form.

1.7 Optimality Criteria

The purpose of optimization algorithms is find the optimal value of a variable or optimal point in the entire search space, for a point to be optimal there are three different types of criteria

1.7.1 Local optimal point

A point or solution x^* is said to be a local optimal point if there exist no point in the neighborhood of x^*. In the parlance of minimization problems a point x^* is a locally minimal point if no point in the neighborhood as a functional value smaller than $f(x^*)$.

1.7.2 Global optimal point

A point or solution x^* is said to be Global Optimal point, if there exists no point in the entire search space which is better than the point x^*, similarly a point x^* is a global minimal point if no point in the entire search space has a functional value smaller than $f(x^*)$.

1.7.3 Inflection Point

A point or solution x^* is said to be inflection point if the function value increases locally as x^* increases and decreases locally as x^* reduces or if the functional value decreases locally as x^* increases and increases locally x^* decreases.

1.8 Objective of the Thesis

Finite Impulse Response (FIR) filter is implemented as a series of multiply and accumulate operations on a programmable Digital Signal Processor (DSP).The multiply and accumulate (MAC) unit of a digital signal processor experiences high switching activity due to signal transitions which results in higher power dissipation. Hamming Distance forms a measure of the switching activity during implementation of the filter. The Objective of the thesis is to minimize the Hamming distance and reduce the signal toggle by using optimization techniques, '*Steepest Descent*' and '*Evolutionary Programming*' so that its power dissipation is reduced whiles its implementation on a Digital Signal Processor.

CHAPTER 2
LITERATURE REVIEW

Digital filters are now widely used throughout the electronics industry, with one of the most common types being the Finite Impulse Response (FIR) filter. FIR filters can have exactly linear phase response and suffer less from the effects of finite word length than IIR filters. However, the drawback is the hardware complexity. A typical FIR filter may requires many stages in order to provide a suitable response and therefore, rather than having a respective multiplier for each filter stage, a more economical means of realizing such a device is to use a CMOS based digital signal processing device wherein all multiplication operations are multiplexed through a single multiplier is of increasing concern to engineers however, is the power consumption of such CMOS based devices, due in particular to their widespread use in a variety of portable, battery powered electronic devices such as mobile phones. It has been shown that the main source of power dissipation in a typical CMOS logic gate is due to switching power, $P_{dynamic}$. Finite Impulse Response (FIR) digital filters are an essential component of digital signal processors, optimizing the algorithm employed by these filters, has a significant effect on the power consumption of the system.

The advancement in VLSI technology results the systems with highly complex functionalities during the last decade. Major research is carried out in the area of digital signal processing (DSP) where high throughput is most desirable. Now in these days DSP systems are capable of handling and performing complex and computationally intensive signal processing tasks at speeds suitable for real-time operation. Example applications include speech recognition and video signal processing systems.

One of the fastest growing areas in the computing industry is the provision of high throughput DSP systems in a portable form. A major limitation of these systems is the operating time provided by the battery, which is dependent on the characteristics of the battery and the power requirement of the system. Battery technology has peaked in recent years, whereas the number of applications requiring portability is increasing, placing an increasing demand on the power budget. A major concern in both portable and non

portable systems is heat dissipation. Overheating can reduce throughput and operation life. Cooling fans and heat sinks significantly add to the overall cost of a system.

Due to the reasons mentioned above, low power design techniques and methodologies are required to be adapted by VLSI system designers, especially those for portable DSP applications.

Henry Samueli [1] has been presented an improved FIR filter coefficient optimization algorithm which allocates additional nonzero digits in the CSD code to compensate for the nonuniform distribution of the CSD coefficient set. The two-stage algorithm consists of search for an optimum scale factor followed by a bivariate local search in the neighborhood of the scaled and rounded CSD coefficients. The increase in filter complexity resulting from the additional CSD digits is minimal, however a significant improvement in the frequency response is obtained. This technique can also be used in conjunction with other techniques for reducing FIR filter complexity such as the prefilter structures or the interpolated FIR filters. These techniques essentially decompose the filter into a cascade of lower complexity structures. Thus the scaling and local search approach presented in this paper can be applied to the individual sections of the cascade to make the entire Structure multiplierless.

Dusan Kodek and Kenneth Steiglitz [2] have presented a comparison between an optimal (branch and bound) algorithm and a suboptimal (local search) algorithm for the design of finite wordlength finite impulse response (FlR) digital filers and showed experimental results for filters of coefficient length from15 to 35. Finally conclude that when computer resources are not available for the optimal method, it is still worth applying the local search method to the filter with rounded coefficients

Farid N. Najam [3] has observe that a common thread that runs through most causes of runtime failure is the extent of circuit activity, the rate at which its nodes are switching and propose a new measure of activity, called the transition density, which may be defined as the “average switching rate” at a circuit node. Based on a stochastic model of logic signals, also presented an algorithm to propagate density values from the primary inputs to internal and output nodes. To illustrate the practical significance of this work, demonstrated how the density values at internal nodes can be used to study circuit reliability by estimating:

1) The average power and ground currents.

2) The average power dissipation.

3) The susceptibility to electromigration failures.

The density propagation algorithm has been implemented in a prototype density simulator. Using this, presented experimental results to assess the validity and feasibility of the approach. However, the algorithm for computing the density in this paper is very basic and does not take into account the effect of inertial delays of logic gates.

Anantha P. Chandrakasan, Miodrag Potkonjak, Renu Mehra, Jan Rabaey, and Robert W. Brodersen [4] have presented an automated high level synthesis system, HYPER-LP, for optimizing power consumption in algorithm specific datapath intensive circuits using a variety of architectural and computational transformations. The synthesis approach consisted of applying transformation primitives in a well defined manner in conjunction with efficient high level estimation of power consumption. The experimental results indicated that more than an order of magnitude reduction in power is possible over current day design methodologies while maintaining the system throughput. It was found that the optimal supply voltage for minimizing power was much lower than existing standards and was around 1.5 V for most of the examples investigated. This work has addressed some key problems in the automated design of low-power systems, and provides a good starting point for addressing other research problems like detailed power estimation, module selection, partitioning, and scheduling for power optimization.

Chetana Nagendra, Robert Michael, Owens Mary and Jane Irwin [5] have shown that the gate pipelined MAC3 circuits, on account of the simplicity of each pipeline stage, can be clocked at very high speeds. However, this advantage is overshadowed by the high power dissipation in the clock drivers, especially for large circuits such as FIR filters and multipliers. In comparison, the half-bit pipelined MAC2 designs have comparable performance (only slightly slower), but have lesser area, power dissipation and present about half the clock load of the MAC3 designs.

A.T. Erdogan and T. Arslan [6] have proposed a new multiplication scheme, for application to single multiplier CMOS based DSP processors, for the implementation of low power digital FIR filters through the reduction of switching activity within the multiplier section of the filter. The scheme operates in conjunction with a transpose direct

form FIR filter structure and a modified DSP processor architecture, through a significant reduction in power can be obtained by using algorithms to order the filter coefficients. This reduction has demonstrated using two basic examples, with different wordlengths and filter orders. Achieving up to 63% reduction in switching activity.

Darren N. Pearson, Keshab K. Parhi [7] have presented a novel approach for low power implementations of finite impulse response (FIR) filters with less hardware overhead than traditional FIR implementations. Parallel or block processing, with duplication of hardware can reduce power consumption parallel processing by block size L requires the critical path to be charged in L times longer time as compared with the sequential implementation and the identical critical path can be charged in longer time with lower supply voltage which leads to lower power consumption. The hardware cost of this approach increases linearly with the block size L and proposed a general technique for block implementation of FIR filters which requires fewer multipliers than the straightforward block implementation. The use of this approach can lead to a further reduction in power consumption and hardware cost.

R. Hezar and V. K. Madisetti [8] have proposed in this paper an efficient design procedure for digital FIR filters whose coefficients are restricted to the ternary set {-1, 0, +1 }, cascaded by a multiplication-free architecture. A dynamic programming algorithm, minimizing the instantaneous error, also proposed to assist in the search for the optimal ternary filter coefficient set. Good power reductions in VLSI implementations appeared feasible when compared to standard implementations. Compared to the other dynamic programming approaches and obtained around 10 dB more stop-band attenuation in our low pass filter designs. Current efforts seek to implement these designs in VLSI.

Mahesh Mehendale, S.D. Sherlekar and G. Venkatesh [9] have presented low power realization of FIR filters using multirate architectures in this paper. They have analyzed the computational complexity of these architectures and shown it to be lower than the both the direct form and block FIR implementations. The architectures enable reducing the frequency and the supply voltage, thus significantly reducing the power dissipation. The results show that with multirate architectures, the power can be reduced by as much as 73% without resulting in any datapath area overhead. Have also showed how the multirate architecture can be implemented using the ’C2x/’C5x DSPs and

presented results to show up to 43% power reduction. Multirate architectures are thus the most power efficient architectures for both the dedicated ASIC and the programmable DSP based implementations.

Mahesh Mehendale, S.D. Sherlekar and G.Venkatesh [10] have presented algorithmic and architectural transforms for low power realization of Finite Impulse Response (FIR) filters implemented both in software on programmable DSPs and as hardwired macros. For the programmable DSP based implementation, these transform address power reduction in the program memory address and data busses and also the multiplier. Also propose architectural extensions to support some of these transformations. The transforms for hard-wired FIR filters aim at reducing the supply voltage while maintaining the throughput and present transforms that reduce the computational complexity of the FIR filter computation and thus achieve power reduction.

P. K. Merakos, K. Masselos, O. Koufopavlou, S. Nikolaidis and C. E. Goutis [11] have presented a novel high level transformation for low power implementation of FIR filters in this paper. The new idea is the reordering of the filter coefficients aiming at the minimization of the switching activity. As a measure of the switching activity the Hamming Distance (HD) between successive coefficients, stored in a memory, is used. The transformation can be incorporated both in application specific architectures and in general purpose programmable architectures. The reordering of the N coefficients, for the HD optimization of their sequence, can be modeled by a Traveling Salesman Problem (TSP), which is a well known NP complete problem. A novel heuristic algorithm for a fast and accurate solution to this problem is proposed. Experimental results show that the proposed technique leads to significant power savings in terms of switching activity reduction.

S.D. Sherlekar, G.Venkatesh and Mahesh Mehendale [12] have addressesed the problem of reducing power dissipation of finite impulse response (FIR) filters implemented on programmable digital signal processors (DSP's). It has described a generic DSP architecture and identifies the main sources of power dissipation during FIR filtering and present seven transformations to reduce power dissipated in one or more of these sources. These transformations complement each other and together operate at

algorithmic, architectural, logic and layout levels of design abstraction. Each of the transformations is discussed in details and the results are presented to highlight its effectiveness. They have showed that the power dissipation can be reduced by more than 40% using these transforms. The transformations have been encapsulated in a framework that provides a comprehensive solution to low-power realization of FIR filters on programmable DSP's.

Vijay Sundararajan and Keshub K. Parhi [13] have presented novel low power synthesis technique for the design of folded or time multiplexed programmable coefficient FIR filters where storage area is trade off for lowering power consumption. A systematic technique is developed for low power mapping of FIR filters to architectures with arbitrary number of multipliers and adders. Power consumed in multipliers is reduced by, reducing switching activity at both the data input as well as the coefficient input. Common input operands can be exposed by unfolding, which however leads to a memo increase. Simulation results are obtained for folding 65 and 129 tap band pass FIR filters. The average power consumed in a multiplier for a fixed number of hardware multipliers with varying unfolding factors compared. It has observed that most of the gains due to unfolding are obtained for relatively small unfolding factors and therefore for relatively small memory area overhead. Depending on the unfolding factor employed the average power consumed in a multiplier seen to reduce anywhere from 54.75% to 81.73% when transpose FIR filters are synthesized as opposed to synthesizing direct-form F!R filters with no unfolding.

Zhan Yu, Meng Lin Yu, Kamran Azadet and Alan N. Willson [14] have exploited the sign-extension property of a 2's complement number in this work. A reduced representation for 2's complement numbers has been proposed to avoid sign extension and the switching of the sign extension bits. The maximum magnitude of a 2's complement number is detected and its reduced representation is dynamically generated to represent the signal. A constant error introduced by the reduced representation and this error is also dynamically compensated. The proposed signal representation is particularly useful in digital filters where the coefficients are slowly varying and have small magnitudes. The proposed technique has been implemented in an adaptive filter and shown to reduced power dissipation by over 38%.

Chang Young Han, Hyoung Joon Park, and Lee Sup Kim [15] have focused on power reduction during multiplication and exploited the spatial redundancy of images. Multipliers were separated into two parts, the higher and lower parts of multiplication. Also optimized the architecture by separating the grayscale pixel data into the higher 4 bits and the lower 4-bits for multiplication and removed unnecessary multiplications by accessing a cache which stores the higher 4-bits multiplication result. A replacement strategy, employing an incrementor, has proposed for adequate cache architecture for image data processing. The total power was reduced by 14% in the multiplier and 10% in the 1-D 4-tap FIR filter. The speed increase was about 17% due to the use of the smaller multipliers. Therefore, dropping the supply voltage with the same clock frequency and constant throughput can save more power.

Harry J. M. Veendrick [16] has derived a simple formula for quick calculations of the maximum short-circuit dissipation of static CMOS circuits. A detailed discussion of this short-circuit dissipation is given based upon the behavior of the inverter when loaded with different capacitances. It was found that if each inverter of a string is designed in such a way that the input and output rise and fall times are equal, the short-circuit dissipation will be much less than the dynamic dissipation <20 percent. This result has been applied to a practical design of a CMOS driving circuit (buffer), which is commonly built up of a string of inverters.

Keshab K. Parhi [17] has reviewed several approaches for low power implementations of building blocks for digital subscriber line (DSL) systems. Low power implementations of fast Fourier transforms (FFTs), FIR filters, and equalizers, and reduction of power consumption by use of dual supply voltages have addressed. It has shown that use of separate Galois Field functional units for multiply accumulates and degree reduction can reduce the energy consumption of RS coders dramatically. A hybrid feed forward and feedback commutator scheme based FFT has shown to require less area and full hardware utilization efficiency. Reduction of switching activity at one or both inputs of the multipliers is a key to reduction of power consumption in FIR filters and equalizers. The switching activity reduced by use of transpose structure and by time-multiplexing of an unfolded filter. A well established retiming approach can be

generalized to find those non critical gates which can be operated with lower supply voltages to reduce the overall system power consumption.

Youngbeom Jang and Sejung Yang [18] have presented a high speed, low power canonic signed digit (CSD) linear phase finite impulse response (FIR) filter structure using vertical common sub expression. In the conventional linear phase CSD filter, the horizontal common sub expression method has been widely utilized due to the inherent symmetrical filter coefficients. In this, use has been made of the Fact that the most significant bits of adjacent filter coefficients in the linear phase filter are also equal since they have mostly similar values. It has shown that the proposed structure is more efficient in the case where bit precision of implementation is lower.

Alberto Garcia, Lukusa D. Kabulepa and Manfred Glasner [19] have proposed an efficient technique for estimating the node transition activity in MAC architectures implementing FIR filters. The emphasis has been on the estimation of the activity at the output of the MAC multiplier. It is based on divide and conquers approach, an accurate yet efficient method yet efficient procedure is developed. The technique has been evaluated for different synthetic and real data sets. The results depict only very slight discrepancies with respect to precise bit level simulation.

Kyung Saeng Kim and Kwyro Lee [20] have designed a 32-tap FIR filter with two 16-tap macros according to the derived condition on the basis of the one chip criterion such as the product of the occupied area and power consumption, which is especially applicable to an FIR filter with multiple taps.

The chip is implemented in 0.6μm CMOS technology with three levels of metal and occupies 2.3×2.5 mm^2 of silicon area having operating frequency of 20 MHz and consumes 75 mW at V_{dd} = 3.3 V.

A.T. Erdogan, M. Hasan and T. Arslan [21] have presented a number of novel architectures for the implementation of low power FIR filtering cores. These architectures have directly translated from flexible algorithms which exploit data and coefficient correlation in order to minimize the effective switched capacitance on the multiplier, and data/coefficient buses. Another characteristic of these algorithms is that they can be combined to form more power efficient algorithms which in turn could he mapped to more effective architectures. It has described the FIR filtering architectures, the

arithmetic processing cores which characterize individual architectures, and provides results which demonstrate up to 39% reduction in power.

Jongsun Park, Woopyo Jeong, Hamid Mahmoodi Meimand, Yongtao Wang, Hunsoo Choo and Kaushik Roy [22] have presents programmable digital finite impulse responses (FIR) filter for high performance and low power applications. The architecture is based on a computation sharing multiplier (CSHM) which specifically targets computation reuse in vector scalar products and can be effectively used in the low complexity programmable FIR filter design. Efficient circuit level techniques, namely a new carry select adder and conditional capture flip-flop (CCFF), are also used to further improve power and performance.

Sathyanarayan **S.** Rao and Arun Ramasubrahmanyan [23] have described that A new algorithm for the design of discrete coefficient FIR filters in the powers-of-two space has been proposed that has advantages of less computation time and simultaneous optimization of filter coefficients. From the example results, it is clear that the proposed method leads to closer approximation to the infinite precision coefficient filter and to the desired frequency response and also the computer time required does not increase drastically with filter length In this letter, we have restricted to designing filters with fixed number of powers-of-two terms and uniform frequency samples of desired frequency response. Also there were no constraints other than the discreteness of coefficients.

Xiaoping Lai have presented that [24] the constrained Chebyshev design of linear phase FIR filters with inequality constraints and equation constraints in the frequency domain has been studied in this paper. The design problem is converted into a series of Chebyshev design problems with frequency equation constraints, which can be solved by the MRemez algorithm proposed in this paper. Convergence of the solutions to this series of problems has been established, and an iterative MRemez algorithm has been proposed. This algorithm uses the MRemez algorithm as the computational core of the iterations. Matlab programs have been designed for the MRemez algorithm and the iterative MRemez algorithm. Design examples demonstrate the effectiveness and the fast convergence of the proposed algorithms. In many filter-design problems, additional constraints are often imposed on the optimal filter in the sense of, say, and minimal

Chebyshev error norm. Based on the characteristic properties of the optimal filter for the Chebyshev design with frequency equation constraints, a modified Remez (MRemez) algorithm is proposed in this paper.

Hong Yang and Yiding Wang have presented about hamming distance that [25] Hamming distance is widely used in channel coding for the purpose of minimizing errors caused in code transmission and the error probability in decoding. A good channel code is designed so that, if a few errors occur in transmission, the output can still be identified with the correct input. This is possible because although incorrect, the output is sufficiently similar to the input to be recognizable. The idea of similarity is made more firmly by the definition of a Hamming distance. Let x and y be two binary sequences of the same length.

$$D(x, y) = \sum_i |(x_i - y_i)|$$

The Hamming distance between these two codes is the number of symbols that disagree. Supposing the code x is transmitted over the channel due to errors, y is received. The decoder will assign to y the code x that minimizes the Hamming distance between them. For example, if the transmitter sends2 (10000) but there is a single bit error and the receiver gets 2 (10001), it can be seen that the "nearest" codeword is in fact 2 (10000) and so the correct codeword is found.

2.1 Multi objective Optimization

Real world engineering design problems are usually characterized by the presence of many conflicting objectives, so the engineering design problems are treated as multiobjective optimization problems. In a multiobjective optimization problem (MOP) there may not exist one solution that is best with respect to all objectives. Usually the aim is to determine the trade off surface, which is a set of non dominated solution points, known as Pareto optimal (PO) or non inferior solutions. In view of the fact that one of the solutions in the non dominated set is absolutely better than any other, any one of them is an acceptable solution. The choice of one solution over the other requires problem knowledge and a number of problem related factors.

Dragan Cvetkovic and Ian C. Parmee [26] have described a new preference method and its use in multiobjective optimization. These preferences are developed with a goal to reduce the cognitive overload associated with the relative importance of a

certain criterion within a multiobjective design environment involving large numbers of objectives.

Christakis Charalambous [27] have discussed the problem of decision making where there are many conflicting objectives is known as the multicriterion optimization (MCO) problem, and this forms the basis for most engineering designs. The purpose of this paper is to present a new method for generating efficient points for the MCO problem. A new function, the shifted minimax function has defined that has the property that any efficient point of the MCO problem can be generated by minimizing the new function with appropriately chosen shifts. Using the new idea for generating efficient points the optimality conditions for the MCO problem can be easily derived. The new method has also applied to the simultaneous amplitude and group delay approximation problem of 1-D digital filters.

Michel. R. Lightner and Stephen. W. Director [28] has examined multicriterion optimization (MCO) one aspect of multicriterion decision making (MCDM) problem. The ideas and method of MCO problem are reviewed. Then developed a weighted ∞-norm method for generating noninferior soutions to the MCO problem. User oriented weight selection heuristics have developed and applied this technique to the MCO design of an MOSFET NAND gate.

As most optimization problems are multiobjective in nature there are many methods available to tackle this kind of problem. Generally the multiobjective optimization problem (MOOP) is handled in four different ways depending upon when the decision maker articulates its preference concerning the different objectives: never, before, during or after the actual optimization procedure. In the first two approaches, the different two objectives are aggregated to one overall objective function. Optimization is then conducted with one optimal design as a result. The result is then strongly dependent on how the objectives were aggregated. Different methods have been developed in the literature to support the decision maker in aggregating the objectives. The third approach is an iterative process where the decision maker progressively articulates its preferences on the different objectives. The underlying assumptions are that once the search for an optimal solution has started and the decision maker has been presented with some alternatives, it will be better equipped to value the objectives. In the fourth and final

approach, optimization is conducted without the decision maker articulating any preferences among the objectives. The outcome of this optimization is a set of Pareto optimal solutions, which elucidate the trade off between the objectives. The decision maker then has to trade the objectives against each other in order to select the final design. Thus optimization is conducted before the decision maker articulates his preferences.

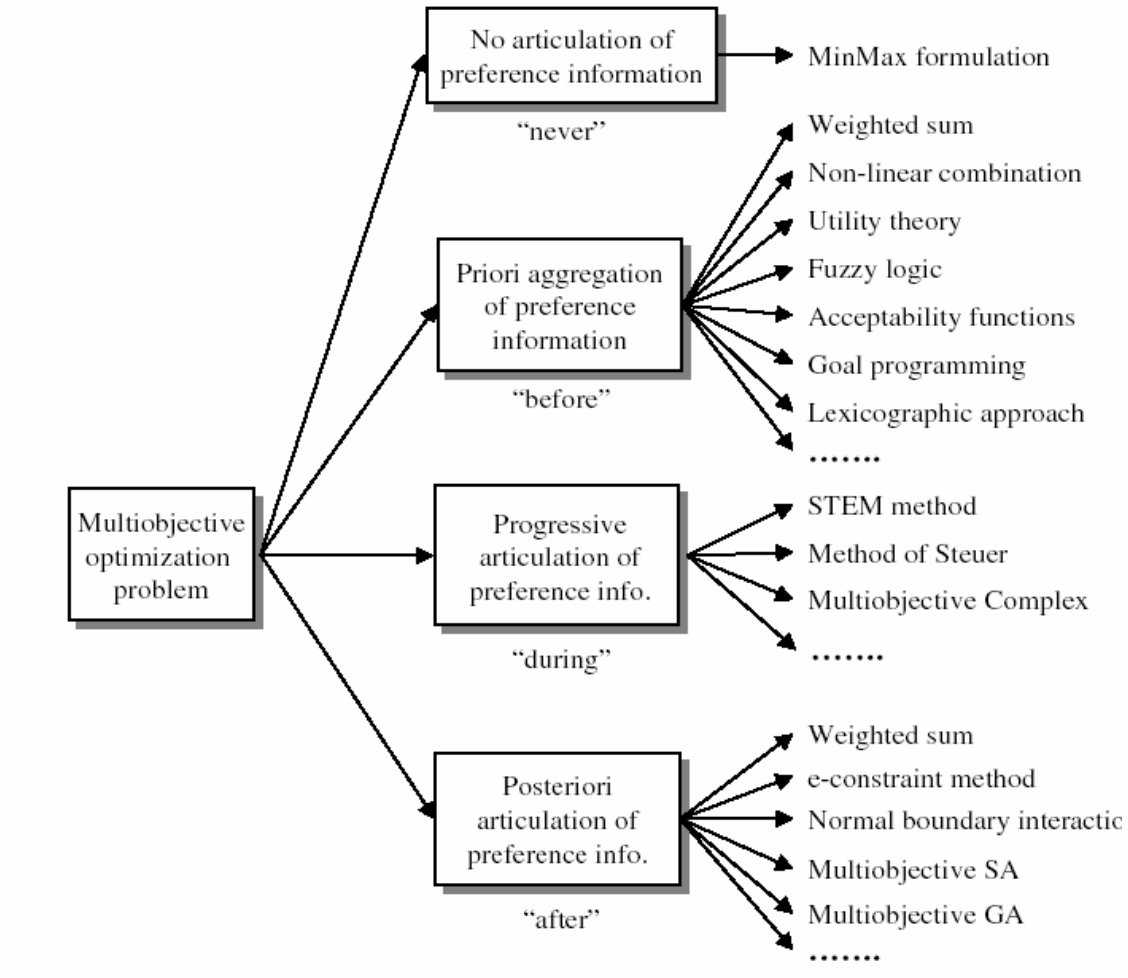

Figure 2.1 Classification of methods of Multiobjective Optimization

2.2 Solution Methodologies

Combining the multiple objectives into one scalar objective whose solution is Pareto optimal point for the original MOP almost always solves the multiobjective problem. Most algorithms have been developed in the linear frame work (linear frame

work and linear constraints), some important techniques described below are used to solve multiobjective optimization.

2.2.1 Weighed Sum Strategy

The weighted sum strategy converts the multiobjective problem of minimizing the vector *F (X)* into a scalar problem by constructing a weighted sum of all the objectives

$$\min_{x \in \Omega} \quad f(x = \sum_{i=1}^{m} w_i F_i(x) \qquad (2.1)$$

$$\text{Subjected to} \quad \sum_{i=1}^{m} w_i = 1 \text{ and } w_i \geq 0 \qquad (2.2)$$

The problem is then optimized using a standard unconstrained optimization algorithm. The problem here is in attaching weighting coefficients to each of the objective.

2.2.2 Є-Constraint Method

The method optimizes one of the objective functions while the others are required to have specified upper bounds. In other words, it minimizes one objective function and simultaneously

Maintains the maximum acceptable levels for the other objective functions. The formulation adopted in this work is as follows:

Minimize $F_i(X), \quad i = 1, 2, \ldots\ldots\ldots r \qquad (2.3)$

Subjected to $g_j(X) \leq 0, \quad j = 1, 2, \ldots\ldots\ldots m \qquad (2.4)$

$h_j(X) = 0, \quad j = 1, 2, \ldots\ldots\ldots p \qquad (2.5)$

$F_i(X) \leq \epsilon_j, \quad j = 1, 2, \ldots\ldots\ldots \text{ and } i \neq j \qquad (2.6)$

The selections of $F_i(X)$ and of this method are not ϵ_j straightforward and depend on the particular problem under consideration. As shown in the above formulation, the optimization problem can be solved for all $F_i(X)$'s $(i = 1, 2, \ldots\ldots\ldots r)$ and the optimum solution that is best suited to the problem can be chosen among the *r* solutions. But this involves much computational effort. The choice of ϵ_j is arbitrary and any choice may also be used depending upon the problem. But the basic philosophy of this method does not alter with different selections of ϵ_j. In general, the higher values of ϵ_j 's mean a wider feasible region for the single objective optimization problem and this may in turn give a more improved solution for $F_i(X)$ at the expense of the other objective functions.

CHAPTER 3

STEEPEST DESCENT BASED FILTER DESIGN

3.1 Filter algorithms

The term filter is often used to describe a device in the form of physical hardware or software that is applied to a noisy set of data in order to extract information about a prescribed quantity of interest.

In general linear time-invariant discrete-time filters are characterized by the general linear constant coefficient difference equation given in (3.1)

$$y(n) = -\sum_{k=1}^{N} a_k .(n-k) + \sum_{k=0}^{M} b_k .x(n-k) \qquad (3.1)$$

Filters are divided into two categories according to their impulse response, those that have a finite duration impulse response (FIR) and those that have an infinite duration impulse response (IIR).

3.2 Finite Impulse Response (FIR) filters

In general an FIR filter is described by the difference equation given in (3.2).

$$y(n) = \sum_{k=0}^{M} b_k .x(n-k) \qquad (3.2)$$

Such a difference equation can be implemented using a transversal filter as shown in figure 3.1. The transversal filter, which is also referred to as a tapped delay line filter, consists of three basic elements: (1) unit-delay element, (2) multiplier, and (3) adder.

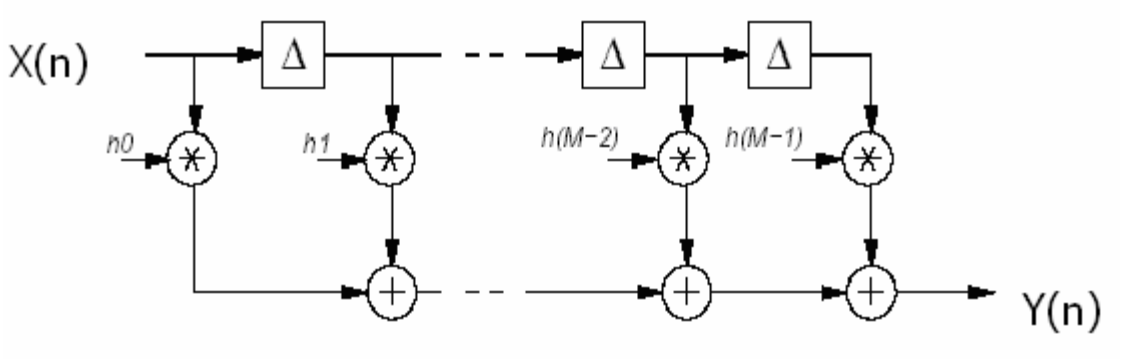

Figure 3.1: Transversal FIR Filter

The number of delay elements used in the filter determines the finite duration of its impulse response. The number of delay elements, shown as *M* in figure 3.1 is commonly referred as the filter order. The role of each multiplier in the filter is to multiply the tap input by a filter coefficient called tap weight.

Because of its linear phase response, FIR filters are extensively used in existing DSP applications. For an FIR filter to have linear phase, its impulse response should satisfy the following symmetry (+), asymmetry (-) condition.

$$h(n) = \pm h(M-1-n) \qquad n = 0, 1 \ldots\ldots M\text{-}1 \tag{3.3}$$

Here *M* is the order of the FIR filter.

FIR filters are particularly useful for applications where exact linear phase response is required. The FIR filter is generally implemented in a non-recursive way which guarantees a stable filter. FIR filter design essentially consists of two parts

(i) Approximation problem

(ii) Realization problem

The approximation stage takes the specification and gives a transfer function through four steps. They are as follows:

(i) A desired or ideal response is chosen, usually in the frequency domain.

(ii) An allowed class of filters is chosen (e.g. the length N for a FIR filters).

(iii) A measure of the quality of approximation is chosen.

(iv) A method or algorithm is selected to find the best filter transfer function.

The realization part deals with choosing the structure to implement the transfer function which may be in the form of circuit diagram or in the form of a program.

There are essentially three well known methods for FIR filter design namely:

(1) The window method

(2) The frequency sampling technique

(3) Optimal filter design methods

3.2.1 The Window Method

In this method, the desired frequency response specification $H_d(w)$, corresponding unit sample response $h_d(n)$ is determined using the following relation

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(w)^{ejwn}\, dw \tag{3.4}$$

$$H_d(w) = \sum_{n=-\infty}^{\infty} h_d(n) e^{-jwn} \tag{3.5}$$

$$W(w) = \sum_{n=0}^{M-1} w(n) e^{-jwn} \tag{3.6}$$

Thus the convolution of $H_d(w)$ with $W(w)$ yields the frequency response of the truncated FIR filter

$$H(w)\frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(v) W(w-v) dw \tag{3.7}$$

The frequency response can also be obtained using the following relation

$$H(w) = \sum_{n=0}^{M-1} h(n) e^{-jwn} \tag{3.8}$$

But direct truncation of $h_d(n)$ to M terms to obtain $h(n)$ leads to the Gibbs phenomenon effect which manifests itself as a fixed percentage overshoot and ripple before and after an approximated discontinuity in the frequency response due to the non-uniform convergence of the Fourier series at a discontinuity. Thus the frequency response obtained by using (3.10) contains ripples in the frequency domain. In order to reduce the ripples, instead of multiplying $h_d(n)$ with a rectangular window $w(n)$, $h_d(n)$ is multiplied with a window function that contains a taper and decays toward zero gradually, instead of abruptly as it occurs in a rectangular window. As multiplication of sequences $h_d(n)$ and w(n) in time domain is equivalent to convolution of $H_d(w)$ and $W(w)$ in the frequency domain, it has the effect of smoothing $H_d(w)$.

The several effects of windowing the Fourier coefficients of the filter on the result of the frequency response of the filter is as follows:

(i) A major effect is that discontinuities in *H(w)* become transition bands between values on either side of the discontinuity.

(ii) The width of the transition bands depends on the width of the main lobe of the frequency response of the window function, *w(n)* i.e. *W(w*).

(iii) Since the filter frequency response is obtained via a convolution relation, it is clear that the resulting filters are never optimal in any sense.

(iv) As *M* (the length of the window function) increases, the mainlobe width of *W(w)* is reduced which reduces the width of the transition band, but this also introduces more ripple in the frequency response.

(v) The window function eliminates the ringing effects at the band edges and does result in lower sidelobes at the expense of an increase in the width of the transition band of the filter.

Some of the windows commonly used are as follows

3.2.1.1 Bartlett triangular window

$$W(n) = \frac{2(n+1)}{N+1} \qquad n = 0, 1, 2, \ldots\ldots, (N\text{-}1)/2 \qquad (3.9)$$

$$= 2 - \frac{2(n+1)}{N+1} \qquad n = (N\text{-}1)/2, \ldots\ldots, N\text{-}1$$

$$= 0 \qquad \text{otherwise}$$

3.2.1.2 Generalized cosine windows

(i) Rectangular

(ii) Hanning,

(iii) Hamming

(iv) Blackman

The general expression for the above four windows may be given as follows

$$W(n) = a - b\cos(2p(n+1)/(N+1)) + c\cos(4p(n+1)/(N+1)) \quad n = 0,1\ldots.N-1 \qquad (3.10)$$
$$= 0 \qquad \text{otherwise}$$

3.2.2 The Frequency Sampling Technique

In this method, the desired frequency response is provided as in the previous method. Now the given frequency response is sampled at a set of equally spaced frequencies to obtain *N* samples. Thus, sampling the continuous frequency response $H_d(w)$ at *N* points essentially gives us the *N* point DFT of $H_d(2pnk/N)$. Thus by using the IDFT formula, the filter coefficients can be calculated using the following formula

$$H(n) = \frac{1}{N}\sum_{n=o}^{N-1} H(k)e^{j(2\pi n/N)k} \qquad (3.11)$$

Now using the above N-point filter response, the continuous frequency response is calculated as an interpolation of the sampled frequency response. The approximation error would then be exactly zero at the sampling frequencies and would be finite in frequencies between them. The smoother the frequency response being approximated, the smaller will be the error of interpolation between the sample points.

There are two different set of frequencies that can be used for taking the samples. One set of frequency samples are at $f_k = k/N$ where $k = 0,1,....N-1$. The other set of uniformly spaced frequency samples can be taken at $f_k = (k + ½)/N$ *for* $k = 0,1,....N-1$. The steps involved in this method are as follows

(i) The desired magnitude response is provided along with the number of samples, N. Given N, the designer determines how fine an interpolation will be used. It was found by that for designs they investigated, where *N* varied from 15 to 256, 16*N* samples of *H(w)* lead to reliable computations, so 16 to 1 interpolation was used.

(ii) Given N values of H_k, the unit sample response of filter to be designed, *h(n)* is calculated using the inverse FFT algorithm.

(iii) In order to obtain values of the interpolated frequency response two procedures were suggested are:

(a) *h(n)* is rotated by *N/2* samples(*N* even) or *(N-1)/2* samples for *N* odd to remove the sharp edges of impulse response, and then 15N zero valued samples are symmetrically placed around the impulse response.

(b) *h(n)* is split around the *N/2nd* sample, and 15N zero-valued samples are placed between the two pieces of the impulse response.

(iv) The zero augmented sequences are transformed using the FFT algorithm to give the interpolated frequency responses.

3.2.2.1 Merits and demerits of frequency sampling technique

(i) Unlike the window method, this technique can be used for any given magnitude response.

(ii) This method is useful for the design of non-prototype filters where the desired magnitude response can take any irregular shape.

(iii)There are some disadvantages with this method i.e. the frequency response obtained by interpolation is equal to the desired frequency response only at the sampled points. At the other points, there will be a finite error present.

3.3. Optimal Filter Design Methods

Many methods are present under this category. The basic idea in each method is to design the filter coefficients again and again until a particular error is minimized. The various methods are as follows:

(i) Least squared error frequency domain design

(ii) Weighted Chebyshev approximation

3.3.1 Least squared error frequency domain design

As seen in the previous method of frequency sampling technique there is no constraint on the response between the sample points, and poor results may be obtained. The frequency sampling technique is more of an interpolation method rather than an approximation method. This method controls the response between the sample points by considering a number of sample points larger than the order of the filter. The purpose of most filters is to separate desired signals from undesired signals or noise. As the energy of the signal is related to the square of the signal, a squared error approximation criterion is appropriate to optimize the design of the FIR filters.

The frequency response of the FIR filter is given by (8) for a *N*-point FIR filter. An error function is defined as follows

$$Error(E) = \sum \left| H(w_k) - H_d(w_k) \right|^2 \tag{3.12}$$

Where $w_k = (2\pi k)/L$ and $H_d(wk)$ are L samples of the desired response, which is the error measure as a sum of the squared differences between the actual and desired frequency response over a set of L frequency samples. The method consists of the following steps

(i) First *'L'* samples from the continuous frequency response are taken, where $L>N$ (length of the impulse response of filter to be designed)

(ii) Then using the following formula

$$h(n) = \frac{1}{N} \sum_{n=0}^{L-1} H(k) e^{j(2\pi n/N)k} \tag{3.13}$$

The L point filter impulse response is calculated.

(iii) Then the obtained filter impulse response is symmetrically truncated to desired length N.

(iv) Then the frequency response is calculated using the following relation

$$H(w) = \sum_{n=0}^{N-1} h(n) e^{-jwn} \tag{3.14}$$

(v) The magnitude of the frequency response at these frequency points for $w_k = (2\pi\ k)/L$ will not be equal to the desired ones, but the overall least square error will be reduced effectively this will reduce the ripple in the filter response.

To further reduce the ripple and overshoot near the band edges, a transition region will be defined with a linear transfer function. Then the L frequency samples are taken at $w_k = (2\pi\ k)/L$ using which the first N samples of the filter are calculated using the above method. Using this method reduces the ripple in the interpolated frequency response.

3.3.2 Weighted Chebyshev Approximation

In this method, following terms are defined

$H_d(w)$ = the desired (real) frequency response of the filter

$H(w)$ = the frequency response of the designed filter

$W(w)$ = the frequency response of the weighting function

The weighting function enables the designer to choose the relative size of the error in different frequency bands. The frequency response of linear phase filters for four different types can be written as follows: $H(w) = e^{-jw(N-1)/2} e^{j(p/2)L} H^*(w)$

$$E(w) = W(w)[H_d(w) - H^*(w)] \quad (3.15)$$

$$E(w) = W(w)[H_d(w) - P(w)\,Q(w)] \quad (3.16)$$

As *Q(w)* is a fixed function of frequency, we can factor out *Q(w)* is a fixed function of frequency, we can factor out As *Q(w)* is a fixed function of frequency, we can factor out *Q(w)*

$$E(w) = W(w)Q(w)\,[H_d(w)/Q(w) - P(w)] \quad (3.17)$$

Then two more terms are defined as:

$$\tilde{W}(w) = W(w)Q(w) \quad (3.18)$$

$$H_d^*(w) = H_d(w)/Q(w) \quad (3.19)$$

The error function can now be written as

$$E(w) = \hat{W}(w)[H_d^* - P(w)Q(w)] \quad (3.20)$$

Thus the Chebyshev approximation consists of finding the set of coefficients $\tilde{a}$ to $\tilde{d}$ so as to minimize the maximum absolute value of *E(w)* over the frequency bands in which the approximation is being performed. The Chebyshev approximation problem may be stated mathematically as

$$|E(w)| = \min\,[\max\,|E(w)|] \quad (3.21)$$

The solution to this problem is given by Parks and McClellan who applied a theorem in theory of Chebyshev approximation called the alternation theorem.

3.4. Implementation of an FIR Filter on DSPs

A low power programmable processor core that will handle the FIR filters described above should have a multiply-accumulate (MAC) unit as one big combinational unit coupled with accumulators register. Having an MAC unit provides several advantages in terms of reducing power consumption. The entire operation sequence $add \rightarrow multiply \rightarrow add$ is done in a single step and controlled by a single instruction. This reduces a significant amount of the effort required in instruction fetching, decoding, and controlling the datapath, if the processor core were to implement this operation sequence with separate instructions, such as add, multiply and add. The MAC unit also avoids temporary values to be write/read to/from the register file, thus preventing excessive data movement. These temporary variables are mapped to the wires connecting the adder, multiplier and the accumulator of the MAC unit.

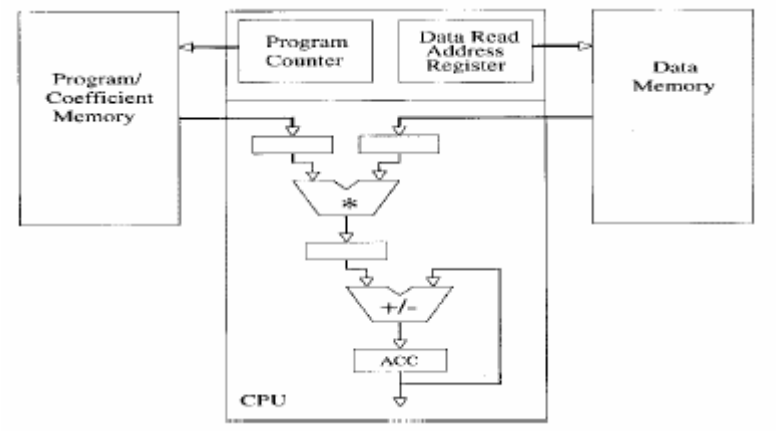

Figure 3.2 DSPs architecture

In order to perform a multiply and accumulate operation in MAC unit, we need to feed this unit with data values and a coefficient simultaneously. This implies a dual port data memory, which holds delay line elements of the FIR filter, and a coefficient memory.

It should have a data address generation unit that should support circular buffering technique. This technique is used to implement address pointer wraparound and therefore allows shifting the delay line of an FIR filter in a power efficient way. For instance when a new input is shifted into the delay line, it is replaced with the “oldest” delay element and the address pointer is incremented by one, now pointing to the new “oldest” delay

element. So instead of shifting all the delay elements, it modifies the pointer to get the same effect. When the address pointer reaches the end of the delay line buffer, it is automatically wrapped around to the beginning of the same buffer. However, each modification to the address pointer should be checked if the address pointer is still within the bounds that specify the start and end of the delay line buffer. An alternative way to implement FIR delay line would be to cascade registers and form a shift register block with M registers, where M is the filter order. But this implementation will suffer from excessive switching activity while shifting all the delay elements. This method may only make sense if the order of the filter or equivalently the number of registers to be cascaded is relatively small.

The FIR algorithm can then be mapped into this architecture as a series of MAC instructions. It could be noted that during the execution of the FIR algorithm, the coefficient values directly impact the signal switching activity especially in the coefficient memory data bus and the multiplier. The coefficients can be optimized so as to reduce this signal switching activity and thus reduce power dissipation.

3.5 Switching Activity

A chip can contain a huge amount of physical capacitance, but if it does not switch then no dynamic power will be consumed. The switching activity determines how often this switching occurs. As given in there are two components to switching activity. The first is the data rate, f_{clk}, which reflects how often on average, new data arrives at each node. This data might or might not be different from the previous data value. In this sense, the data rate f_{clk} describes how often on average, switching could occur. For example, in synchronous systems f_{clk} might correspond to the clock frequency.

The second component of activity is the data activity, $\alpha_{0\to1}$ corresponding to the expected number of energy consuming transitions that will be triggered by the arrival of each new piece of data. So while fclk determines the average periodicity of data arrivals, $\alpha_{0\to1}$ determines how many transitions each arrival will spark. For circuits that do not experience glitching $\alpha_{0\to1}$ can be interpreted as the probability that an energy consuming (zero to one) transition will occur during a single clock period.

The data activity $\alpha_{0\to1}$ can be combined with the physical capacitance C_L to obtain an effective capacitance, $C_{eff} = \alpha_{0\to1} * C_L$ which describes the average capacitance charged during each $1/f_{clk}$ period. This reflects the fact that neither the physical capacitance nor the activity alone determines dynamic power consumption. Evaluating the effective capacitance of a design is non-trivial, as it requires knowledge of both the physical aspects of the design (such as technology parameters, circuit structure, and delay model) as well as the signal statistics (data activity and correlations).

3.5.1 Physical Capacitance

Dynamic power consumption depends linearly on the physical capacitance being switched. The physical capacitance in CMOS circuits stems from two primary sources: devices and interconnect. Capacitances can be kept at a minimum by using less logic, smaller devices, and fewer and shorter wires. Some techniques reducing the active area include resource sharing, logic minimization and gate sizing. Techniques for reducing the interconnect include register sharing, common sub-function extraction, placement and routing. However we are not free to optimize capacitance independently. For example reducing device sizes reduces physical capacitance, but it also reduces the current drive ability of the transistors making the circuit operate more slowly. This loss in performance might prevent us from lowering V_{dd} as much as it might otherwise be able to do. If the designer is free to scale voltage it does not make sense to minimize physical capacitance without considering the side effects. Likewise, if voltage and/or activity can be significantly reduced by allowing some increase in interconnect capacitance, this may result in a net decrease in power.

3.6 Power Dissipation in FIR Filters

Each step in FIR filter algorithm involves getting the appropriate coefficient and data value and performing multiply-accumulate computation. Thus address and data buses of both the memories and multiplier see experiences the most signal switching activity during FIR filtering. Hence these form the main sources of power dissipation.

3.6.1 Measures of Power Dissipation in Buses

Signal switching activity is the major component of power dissipation in CMOS circuits. The power dissipation depends both on the frequency of switching and capacitive loading of the signal. For a typical embedded processor, address and data

buses are networks with a large capacitive loading. Hence signal in these networks has a significant impact on power consumption. In addition to the capacitance of each signal, inter signal capacitance also contributes to bus power dissipation. The power dissipation due to inter signal capacitance varies depending upon the adjacent signal values. The current required for signals to switch between 5's (0101) and A's (1010) is about 25% more than the current required for the signals to switch between 0's (0000) and F's.

3.6.2 Measures of Power Dissipation in Multiplier

Due to high speed requirements, parallel array architecture are used for implementing dedicated multiplier in programmable DSPs. The power dissipation of the multiplier is directly proportional to the number of switchings at all the internal nodes of the multiplier. The number of internal node switchings depends on the multiplier input values. This dependence can be analyzed using the measure of circuit activity the transition density. Transition density of a signal is the average number of transitions of the signal per unit time. For an array multiplier it can be shown that power dissipation depends on the transition densities and the probabilities of the multiplier inputs. The transition density of the multiplier inputs depends upon the Hamming distance between successive input values. The input signal probability depends upon the number of 1s in the input values of the multiplier. These two thus forms the measure of multipliers power dissipation. It can also be shown that the transitions in least significant (LSBs) of the multiplier input contribute more to the power dissipation than the most significant bits (MSBs). Thus while minimizing transition densities of all the input is important, larger gains are achieved by focusing on lower order bits of the input signal.

3.6.3 Power Reduction in Data Buses and Multipliers

During FIR filtering, the coefficient and data memory data buses provide successive coefficients and data values for the weighted sum computation. The power dissipation in the coefficient memory data bus hence depends on the successive coefficient values and the power dissipation in the data memory data bus depends upon the successive data values. Since the data memory value forms the input to the FIR filter, the data memory values forms the inputs to the FIR filter, the data memory data bus power dissipation cannot be controlled during FIR filter synthesis. The coefficient memory data bus power however can very much be minimized by optimizing the filter

coefficient so as to reduce the Hamming distance between the successive coefficients values and also by reducing the total number of signal toggling in opposite directions between successive coefficients. The coefficients and the input data samples form the inputs to the multiplier during FIR filtering. The multiplier power dissipation thus depends upon the number of toggles and also on the number of 1s in these inputs. The coefficient optimization for reducing the coefficient memory data bus power thus also reduces the multiplier power. Higher power reduction can be achieved by focusing on the lower significant bits of the coefficients during minimization.

3.7. Hamming Distance Minimization Algorithm

3.7.1 Problem definition

The Hamming distance minimization problem using Steepest Decent approach stated as follows:

For a Given N-tap FIR filter with coefficients A_i, $i = 0, N-1$ that satisfy the filter response in terms of passband ripples, stopband attenuation and linear phase, find a new set of coefficients A_i, $i = 0, N-1$ such that the total Hamming distance between successive coefficients is minimized while still satisfying the desired filter characteristics in terms of passband ripple and stopband attenuation. Also retain the linear phase characteristics if such this constraint is specified.

3.7.2 Problem Formulation

The Hamming Distance minimization problem is formulated as a local search problem, where the optimum coefficient values are searched in their neighborhood. This is done by using an iterative improvement process. During each iteration one or more coefficients are suitably modified so as to reduce the total Hamming distance while still satisfying the desired filter characteristics. The optimization process continues till no further reduction is possible.

The coefficient optimization is done in two phases. In the first phase, all the coefficients are scaled uniformly. The advantage of such an approach is that it does not affect the filter characteristics in terms of passband ripples and stopband attenuation and phase response. The scaling results in the same gain/attenuation ratio. In the second phase of optimization one coefficient is perturbed in each iteration. In case of requirement to retain the linear phase characteristics, the coefficients are perturbed in pairs (A_i and A_{N-1-}

$_i$) so as to preserve coefficient symmetry. The selection of coefficient for perturbation and the amount of perturbation has the direct impact on overall optimization quality. Various strategies can be adopted for coefficients perturbation. The strategies adopted here include *'Steepest Decent'* and *'Evolutionary programming'*. In the Steepest Decent approach the best coefficient perturbation is selected at every stage. The new value in the neighborhood of the coefficient value is searched for which the Hamming distance is minimum from the previous value.

3.7.3 Hamming Distance Minimization Algorithm – Components

3.7.3.1 Coefficient Scaling

The first phase of the algorithm involves uniformly scaling the coefficient so as to reduce the total Hamming distance between successive coefficients. For N-tap filter with N coefficients *(A_i, i = 0, N-1)*, the output *Y (n)* is given by equation (3.2). Scaling the output preserves the filter characteristics in terms of passband ripples and stopband attenuation, but results in overall magnitude gain equal to the scale factor. For a scale factor K, from equation (3.2)

$$K * Y(n) = K * (\sum_{i=o}^{N-1} (A_i * X_{n-1})) = \sum_{i=0}^{N-1} ((K * A_i) * X_{n-1}) \qquad (3.22)$$

Thus the coefficient of a scaled filter are given by *($K*A_i$)*. The scaled coefficients have different bit patterns in their binary representation than the unscaled coefficients. Given the allowable range of scaling ($\pm$ 3db), an optimal scaling factor can be found such that the total Hamming distance between successive coefficients ($K*A_i$) is least. The scaled coefficients from the initial solution for the phase second of the algorithm.

3.7.3.2 Computing Filter Response

In phase second of the algorithm, the filter characteristics are computed for every perturbation. The overall computational efficiency of the algorithm thus depends on how fast the filter characteristics (passband ripple and stopband attenuation) are derived. The frequency response of the filter is can be computed by taking Fourier Transform of its unit impulse response. The frequency response of an FIR filter can be computed by taking Fourier Transform of a sequence consisting of a filter coefficients padded with 0s. The radix-2 FFT (Fast Fourier Transform) technique is used to compute the filter frequency response. The numbers of zeros to be padded with are decided based on the

desired resolution (number of frequency points) and also make the total number of points a power of 2. Multiply the number of coefficients with 8 and pick the nearest highest power of 2 to decide the total number of points for FFT computation. The passband and stopband frequency ranges, the passband and stopband attenuation are be computed by analyzing the frequency response.

3.7.3.3 Coefficient Perturbation

Perturbing a coefficient, the selected coefficient A_i is modified in such a way that the Hamming distance between the new coefficient value A_i and its adjacent coefficient values $(A_i\text{-}1, A_i\text{-}1)$ is less than the Hamming distance between the current coefficient values A_i and the adjacent coefficient values. The change in coefficient value needs to be as small as possible, so as to minimally impact the filter characteristics. The neighborhood of the selected coefficient (A_i) is searched so as to find the nearest highest and near lower coefficient so as to reduce the Hamming distance. The maximum allowable difference between the perturbed coefficients and the original coefficients can be controlled so as to focus on the lower significant bits during the optimization.

3.7.3.4 Coefficient Selection

Using Steepest Decent strategy, for every coefficient its nearest higher and nearest lower coefficient values are identified. A new set of coefficient is formed by replacing one of the coefficient with its nearest higher or nearest lower value. This approach is used to generate 2N sets of coefficients for an N tap filter, during each iteration of the optimization process. From the 2N sets of coefficients for the next iteration. The gain function γ is computed as follows

$$\gamma = (P_{db\_req} - P_{db}) / P_{db\_req} + (S_{db\_req} - S_{db}) / S_{db\_req} * HD_{\_red} \qquad (3.23)$$

Where $HD_{\_red}$ is the reduction in the total Hamming distance for the new set of coefficients compared to the total Hamming distance for the current set of coefficients, P_{db_req} is the desired passband ripple, S_{db_req} is the desired stopband attenuation, P_{db} is the passband ripple of new set of coefficients and S_{db} is the stopband attenuation for the new set of coefficients.

3.7.4 Algorithm for Hamming Distance minimization of FIR filters using Steepest Descent technique.

Step 1 :- For a given FIR filter coefficients A[i] (i =1,N-1) and given pass band ripples (P_{db_req}) and stop band attenuation(S_{db_req}). Calculate the Hamming Distance between A[i], A[i-1] and A[i], A[i+1]

Step 2 :- Now perturb each coefficient (increase the value of each coefficient one by one by 1) and calculate new hamming distance between the coefficients.

A[i+], A[i-1] and A[i+], A[i+1]

Such that

$HD(A[i], A[i-1]) + HD(A[i], A[i+1]) > HD(A[i+], A[i-1]) + HD(A[i+], A[i+1])$

And

Euclidian distance $(A[i+] - A[i])$ is minimum

Step 3 :- Replace *A[i]* with *A[i+]* to get a new set of coefficients.

Step 4 :- Compute pass band ripples(P_{dbi+}) and stop band attenuation(S_{dbi+}) from a new set of coefficients *A[i+]*

Step 5 :- If pass band ripples $P_{dbi+} < P_{db\_req}$ and stop band attenuation $S_{dbi+} > S_{db\_req}$ calculate tolerance

$T_{oli+} = (P_{db\_req} - P_{dbi+})/P_{db\_req} + (S_{dbi+} - S_{db\_req})/S_{db\_req}$

Else

$Tol_{i+} = 0$

Step 6 :- Now again perturb each coefficient (decrease the value of each coefficient one by one by 1) and calculate new hamming distance between the coefficients.

A[i-], A[i-1] and A[i-], A[i+1]

Such that

$HD(A[i], A[i-1]) + HD(A[i], A[i+1]) > HD(A[i-], A[i-1]) + HD(A[i-], A[i+1])$

And

Euclidian distance $(A[i-] - A[i])$ is minimum

Step 7 :- :- Replace *A[i]* with *A[i-]* to get a new set of coefficients.

Step 8 :- Compute pass band ripples (P_{dbi-}) and stop band attenuation (S_{dbi-}) from a new set of coefficients *A[i-]*.

Step 9 :- If pass band ripples $P_{dbi-} < P_{db\_req}$ and stop band attenuation $S_{dbi-} > S_{db\_req}$ calculate tolerance

$T_{oli-} = (P_{dbreq} - P_{dbi-})/P_{dbreq} + (S_{dbi-} - S_{dbreq})/S_{dbreq}$

Else

$Tol_{i-} = 0$

Step 10 :- Calculate gain function **γ** for new coefficient values $A[i+]$ and $A[i-]$

$\gamma = (Tolerance * HD\ reduction)$ is maximum

for $\boldsymbol{\gamma} > 0$

Replace original coefficients with new value

CHAPTER 4

EP BASED FILTER DESIGN

Evolutionary Programming is search algorithms based on the mechanics of natural selection and natural genetics. Evolutionary Programming, originally conceived by Lawrence J. Fogel in 1960, is a stochastic optimization strategy similar to Genetic Algorithms, but instead places emphasis on the behavioral linkage between parents and their off springs, rather than seeking to emulate specific Genetic operators as observed in nature. Evolutionary programming is similar to EVOLUTION STRATEGIEs, although the two approaches developed independently. Like both ES and GAs, EP is a useful method of optimization when other techniques such as gradient descent or direct, analytical discovery are not possible. Combinatory and real-valued FUNCTION OPTIMIZATION in which the optimization surface or FITNESS landscape is “rugged", possessing many locally optimal solutions, are well suited for evolutionary programming. Before discussing Evolutionary based OPF, it is advisable to discuss Evolutionary programming.

4.1 Overview of Evolutionary Programming

L. J. Fogel presents evolutionary Programming (EP). He initially studied this method to develop the artificial intelligence and succeeded in evolving a mathematical automaton that predicts a binary time series. Later, in the middle of 80’s, his son David Fogel further developed it to solve more general tasks including prediction problems, optimization, and machine learning. Since this approach modeled organic evolution at the level of evolving species, the original EP does not rely on any kind of recombination. Evolutionary programming (EP) is a stochastic optimization strategy, which places emphasis on the behavioral linkage between parents and their offspring. It is a powerful and general optimization method, which does not depend on the first and second derivatives of the objective function and the constraints of the problem. Evolutionary programming is a probabilistic search technique, which generates the initial parent vectors distributed uniformly in intervals within the limits and obtains global optimum solution over number of iterations. The main stages of this technique are initialization,

creation of off – spring vectors by mutation and competition and selection of best vectors to evaluate best fitness solution. As the history of the field suggests there are many different variants of Evolutionary Algorithms. The common underlying idea behind all these techniques is the same: given a population of individuals the environmental pressure causes natural selection (survival of the fittest) and this causes a rise in the fitness of the population. Given a quality function to be maximized we can randomly create a set of candidate solutions, i.e., elements of the function's domain, and apply the quality function as an abstract fitness measure – the higher the better. Based on this fitness, some of the better candidates are chosen to seed the next generation by applying recombination and/or mutation to them. Recombination is an operator applied to two or more selected candidates (the so-called parents) and results one or more new candidates (the children). Mutation is applied to one candidate and results in one new candidate. Executing recombination and mutation leads to a set of new candidates (the offspring) that compete based on their fitness (and possibly age) – with the old ones for a place in the next generation. This process can be iterated until a candidate with sufficient quality (a solution) is found or a previously set computational limit is reached. In this process there are two fundamental forces that form the basis of evolutionary systems.

• Variation operator (mutation) creates the necessary diversity and thereby facilitates Novelty.

• Selection acts as a force pushing quality.

The combined application of variation and selection generally leads to improving fitness values in consecutive populations. It is easy (although somewhat misleading) to see such a process as if the evolution is optimizing, or at least "approximating", by approaching optimal values closer and closer over its course. Alternatively, evolution it is often seen as a process of adaptation. From this perspective, the fitness is not seen as an objective function to be optimized, but as an expression of environmental requirements. Matching these requirements more closely implies an increased viability, reflected in a higher number of offspring. The evolutionary process makes the population adapts to the population adapt to the environment better and better. Let us note that many components of such an evolutionary process are stochastic. During selection fitter individuals have a higher chance to be selected than less fit ones, but typically even the weak individuals

have a chance to become a parent or to survive. For mutation, the pieces that will be mutated with in a candidate solution and the new pieces replacing them are chosen randomly. The evaluation (fitness) function represents a heuristic estimation of solution quality and the variation and the selection operators drive the search process. The most important advantage of EP is that it uses only the objective function information and hence independent of the nature of the search space such as smoothness, convexity or uni-modality, etc.

4.2 EP vs. Classical methods

The characteristics which make EP differ from other classical methods are as follows:

1) EP requires only the evaluation of the fitness function, which is formed from objective function so that every solution could be given a quality value.
2) EP operates on the encoded string of the problem parameters rather than the actual parameters of the problem.
3) EP has an advantage over quadratic programming, non-linear programming in the sense that these methods suffer from the difficulty in handling inequality constraints.
4) EP uses a population of points rather than a single point in their search.
5) EP also results in less loss of accuracy
6) Classical methods are not guaranteed to converge to the global optimum of the general non-convex optimum problem where as EP has this advantage of converging into a global optimum.

4.3 Components of Evolutionary Programming

Application or definition of Evolutionary Programming require certain steps, components, procedures or operators that must be specified in order to define a particular EA. The most important components are:

• Representation (definition of individuals)
• Evaluation function (or fitness function)
• Population
• Parent selection mechanism
• Variation operators like mutation
• Survivor selection mechanism (replacement)

Each of these components must be specified in order to define a particular EA. Furthermore, to obtain a running algorithm the initialization procedure and a termination condition must be also defined. This is shown in the block diagram given below.

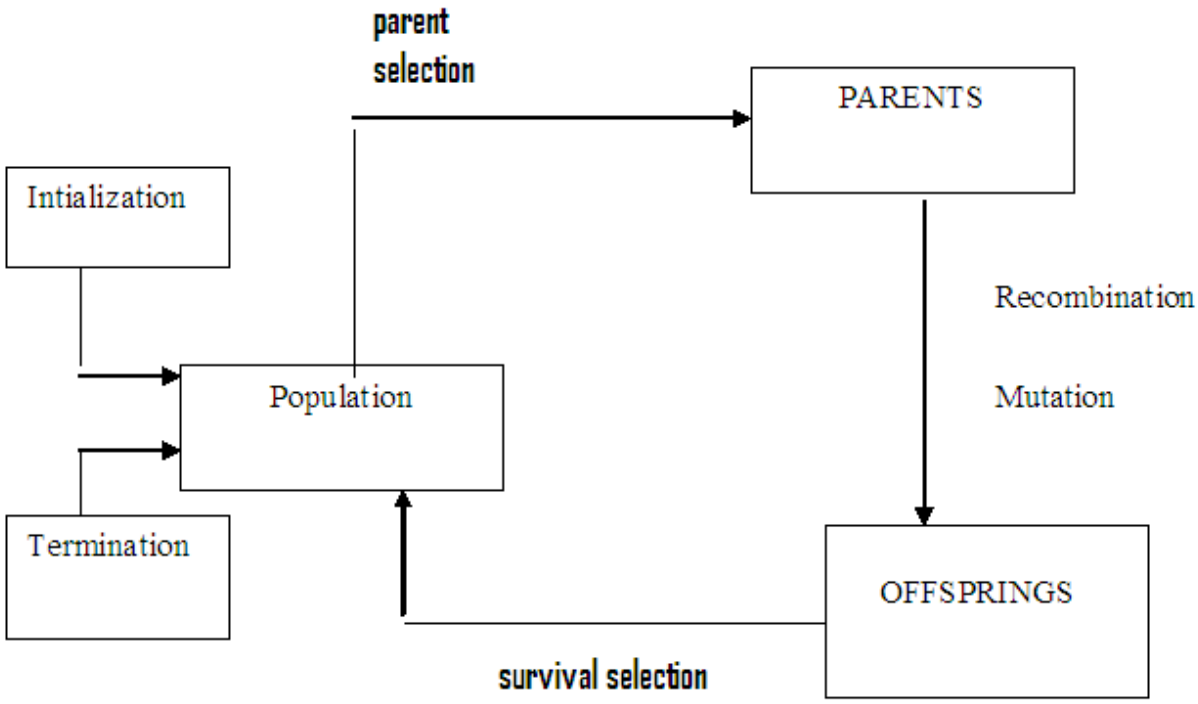

Fig: 4.1 The general scheme of Evolutionary Programming

4.3.1 Representation (Definition of Individuals)

The first design step is commonly called representation, as it amounts to specifying a mapping from the phenotypes onto a set of genotypes that are said to represent these phenotypes which means to link the" real world" to the "EA world", that is to set up a bridge between the original problem context and the problem solving space where evolution will take place. Objects forming possible solutions within the original problem context are referred to as phenotypes, their encoding, the individuals within the EA, are called genotypes. For instance, given an optimization problem on integers, the given set of integers would form the set of phenotypes. Then one could decide to represent them by their binary code, hence 18 would be seen as a phenotype and 10010 as a genotype representing it. It is necessary to understand that the phenotype space can be very different from the genotype space, and that the whole evolutionary search takes

place in the genotype space. A solution – a good phenotype – is obtained by decoding the best genotype after termination. To this end, it should hold that the (optimal) solution to the problem at hand – a phenotype – is represented in the given genotype space. The common EA terminology uses many synonyms for naming the elements of these two spaces. From the side of the original problem context, candidate solution, phenotype, and individual are used to denote points of the space of possible solutions. This space itself is commonly called the phenotype space.

In this sense it is synonymous with encoding, e.g., one could mention binary representation or binary encoding of candidate solutions. The inverse or reverse mapping from genotypes to phenotypes is usually called decoding and it is required that the representation be invertible: to each genotype there has to be at most one corresponding phenotype. The word representation can also be used in a slightly deferent sense, where the emphasis is not on the mapping itself, but on the "data structure" of the genotype space.

4.3.2 Evaluation Function (Fitness Function)

Evaluation function represents the task to solve in the evolutionary context. Technically, it is a function or procedure that assigns a quality measure to genotypes. More accurately, it defines what improvement means. The role of the evaluation function is to represent the requirements to adapt to. It forms the basis for selection, and thereby it facilitates improvements. Typically, this function is composed from a quality measure in the phenotype space and the inverse representation.

4.3.3 Population

Population means how many individuals have in it i.e. setting the population size. As opposed to variation operator, that act on one or two parent individuals, the selection operators (the parent selection and survival selection) work at population level. In general, the take the whole population into account and choices are always made relative to what we have. For instance, the best individual of the given population is chosen to be replaced by a new one. In almost all EPs applications, the population size is constant, not changing during the evolutionary search. The objective of population is to hold (the representation of) possible solutions. A population is a multi set of genotypes. The population forms the unit of evolution. Individuals are static objects not changing or

adapting. It is the population that does. The diversity of a population is a measure of the number of different solutions present. No single measure for diversity exists; typically people might refer to the number of different fitness values present, the number of phenotypes present, or the number of different genotypes.

4.3.4 Parent Selection Mechanism

Chromosomes are selected from the population to be parents. The problem is how to select these chromosomes. According to Darwin's evolution theory the best ones should survive and create new offspring. The objective of parent selection or mating selection is to compare among the individuals based on their quality, and to allow the better individuals to become parents of the next generation. An individual is a parent if it has been selected to undergo variation in order to create offspring. Together with the survivor selection mechanism, parent selection is responsible for pushing quality improvements. Thus, high quality individuals get a higher chance to become parents than those with low quality. However, low quality individuals are often given a small, but positive chance; otherwise the whole search would become too greedy and get stuck in a local optimum.

There are many methods how to select the best chromosomes, for example roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection and some others which are explain below:

4.3.4.1 Roulette Wheel Selection

Parents are selected according to their fitness. The better the chromosomes are, the more chances to be selected they have. Imagine a roulette wheel where are placed all chromosomes in the population, every one has its place big accordingly to its fitness function. Then a marble is thrown there and selects the chromosome. Chromosome with bigger fitness will be selected more times.

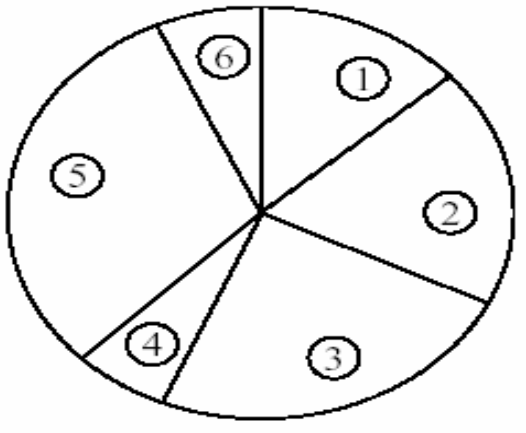

Figure 4.2 Roulette wheel selections

For example, in Fig. 4 the circumference of the roulette wheel is the sum of all six individual's fitness values. Individual 5 is the fit individual and occupies the largest interval, whereas individuals 6 and 4 are the least fit and have correspondingly smaller intervals within the roulette wheel. To select an individual, a random number is generated in the interval [0, Sum] and the individual whose segment spans the random number is selected. This process is repeated until the desired number of individuals has been selected.

4.3.4.2 Rank Selection

In the roulette wheel selection, the problem is when the fitnesses differ very much. For example, if the best chromosome fitness is 90% of all the roulette wheel then the other chromosomes will have very few chances to be selected. But, Rank selection first ranks the population and then every chromosome receives fitness from this ranking. The worst will have fitness 1, second worst will have 2 etc. and the best will have fitness N (number of chromosomes in population). After this all the chromosomes have a chance to be selected. But this method can lead to slower convergence, because the best chromosomes do not differ so much from other ones.

4.3.4.3 Steady-State Selection

This is not a particular method of selecting parents. The main idea of this selection is that big part of chromosomes should survive to the next generation. EP then works in a following way. In every generations a few (good with high fitness) chromosomes are selected for creating a new offspring. Then some (bad with low fitness)

chromosomes are removed and then, the new offspring is placed in their place. The rest of population survives to the new generation.

4.3.4.4 Elitism

Elistism is quite different. When creating a new population by crossover and mutation, we have a big chance that we will loose the best chromosome. Elitism is the name of a method which first copies the best chromosome (or a few best chromosomes) to the new population.

4.3.5 Mutation

The variation operator is commonly called mutation. It is applied to one genotype and delivers a (slightly) modified mutant, the child or offspring of it. A mutation operator is always stochastic: its output- the child- depends on the outcome of a series of random choices. A problem specific heuristic operator acting on one individual could be termed as mutation. However, in general mutation is to cause a random, unbiased change. It is worth noting that variation operators form the Evolutionary implementation of the elementary steps within the search space. Generating a child amounts to stepping to a new point in this space. From this point: mutation has a theoretical role too: it can guarantee that the space is connected.

4.3.6 Survivor Selection Mechanism

The role of survivor selection is to distinguish among individuals based on their quality. In that it is similar to parent selection, but it is used in different stage of evolutionary cycle. The survival selection mechanism is called after having created the offspring of the selected parents. A choice has to be made on which individuals will be allowed in the next generation. This decision is usually based on their fitness values, favoring those with higher quality. Survivor selection is also often called replacement selection or replacement strategy. In many cases, the two terms can be used interchangeably. The choice between the two is often arbitrary. The preference for using replacement can be motivated by the skewed proportion of the number of individuals in the population and the number of newly created children. In particular, if the number of children is quite small with respect to the population size e.g., 2 children and a population of 100. In this case, the survivor selection step is as simple as to choose the two old individuals that are to be deleted to make place for the new ones. In other words, it is

more efficient to declare that everybody survives unless deleted and to choose whom to replace. If the proportion is not skewed like this, e.g., 500 children made from a population of 100, and then this is not an option, so using the term survivor selection is appropriate. Each individual in the competing pool is evaluated for its fitness. All individuals compete with each other for selection. The best K individuals with maximum fitness values are retained to be parents of the next generation. The process of creating off spring and selecting those with maximum fitness are repeated until there is no appreciable improvement in the maximum fitness value or it is repeated up to a pre specified number of iterations.

4.4 EP approach for Hamming Distance Minimization of FIR Filter

Evolutionary programming is successfully used for the design of FIR filters. The problem is formulated as error minimization between the Ideal frequency response and the desired frequency response as per the design specifications in terms of pass band ripple, stopband attenuation and linear phase. Here one more objective added is the Hamming distance between the successive values of the designed filter should be minimum than the ideal filter coefficients. As the Hamming Distance is the measure of the signal switching activity it should also be minimized to reduce the power dissipation in the multipliers while implementing the FIR filtering operation on digital signal processors. So the problem is multiobjective optimization problem and it is solved by using weighted sum approach, converting the problem into single objective by assigning the appropriate weights.

4.4.1 Problem Formulation

Minimax and least mean square errors are generally used to evaluate the fitness function of Chromosome (filter coefficients). But minimax error criteria produced better results with higher speed than least mean square error strategy. Digital FIR filter transfer function is defined as

$$H(z,\phi) = \sum_{n=0}^{N} a_n z^{-n} \tag{4.1}$$

Where ϕ is the vector of filter coefficients, [a_1 a_2 a_3 a_4 a_5……..a_{N-1} a_N] and N is the order of filter. The corresponding freq. response is obtained by substituting $Z = e^{-j\omega nT}$

$$H(\omega,\phi)=\sum_{n=0}^{N} a_n e^{-j\omega nT} \tag{4.2}$$

Where T and ω are the sampling period and angular frequency respectively.

4.4.2 Mini max Error

Mini max objective function is defined as follows. It searches for the maximum peak error throughout a discrete frequency domain by subtracting the magnitude response of the ideal filter to designed filter. The minimax criterion minimizes the maximum error between the filter frequency response $H_D(\omega,\phi)$ and ideal response $H_I(\omega)$. Besides the frequency response error, hamming distance is also minimized. So our problem becomes multi objective optimization problem having two objectives to minimize the maximum frequency error and to minimize the Hamming distance.

$$Error = \max\left|H_I(\omega) - H_D(\omega)\right| \tag{4.3}$$

Where $H_I(\omega)$ and $H_D(\omega,\phi)$ are defined as the frequency responses of the ideal and designed filters.

4.4.3 Least Mean Square (LMS) Error

Least mean square error objective function is defined as the sum of the errors throughout a discrete frequency domain by subtracting the magnitude response of the ideal filter to designed filter.

$$E_{tot} = Error = \left\{\sum\left[\left\|H_I(\omega) - \left|H_D(\omega,\phi)\right|\right\|\right]^2\right\}^{\frac{1}{2}} \tag{4.4}$$

Minimax strategy generates better results than least mean square error for binary encoding and roulette wheel selection.

4.4.4 Fitness Function

The Evolutionary Programming must find a design that satisfies user specifications and minimizes transition activity of digital gates. These two objectives, however, are hierarchical: specification fulfillment is mandatory, while reduction of Hamming distance is an additional quality.

Therefore, a two-step fitness function has been devised. First of all, we consider filter specifications (given as a frequency mask): the frequency range is sampled at N equally spaced frequencies ω, and the filter response $H(\omega)$ is calculated for every ω in the pass-band and in the stop-band. The partial error ϵ_i is set to zero, if $H(\omega)$ lies within the

specifications, otherwise it is proportional to the overshoot or undershoot with respect to the given tolerance.

$$f_M = \frac{1}{1 + E_{tot}} \tag{4.5}$$

It measures the extent to which the filter complies with frequency specifications and f_M = 1 when specifications are completely met.

The second step is the evaluation of the activity fitness, a measure of the Hamming distance for this the total Hamming distance (HD_I) of the ideal filter is calculated. The filter is designed such that the Hamming distance of the designed filter should be the less than that. The fitness function is defined as

$$f_H = HD_I - HD_D \tag{4.6}$$

where N_T is the number of weighted digital transitions per input sample and a is a constant. Its contribution is higher for filters with low transition activity, which is responsible for power consumption. Finally the overall fitness defined as

$$f = f_M + f_H \tag{4.7}$$

4.4.5 Solution Methodology

The intent of work is to optimize the coefficients of FIR filter, to minimize the Hamming distance and satisfying the desired filter characteristics in terms of passband ripple and stopband attenuation. This section is organized to give a solution methodology, which minimizes the Hamming distance and least mean square error simultaneously in multiobjective frame work.

The multi objective problem of minimizing the Hamming distance and mean square error is converted into a scalar problem by constructing a weighted sum of the objectives to generate Pareto optimal solutions. The Pareto optimal solutions for different simulated weight combinations are generated considering both the objectives simultaneously. To simulate weight combination, weights, w_i, $i = 1, 2, \ldots\ldots L$ are varied from 0.1 to 1.0 in steps 0.1, so that their sum is 1.0. The weighting coefficients w_1 and w_2 are used to select effect of error and Hamming distance.

The weighted objective function is written as

$$f = w_1 f_M + w_2 f_H \tag{4.8}$$

Where f_M and f_H are the fitness functions for Mean square error and Hamming distance. The scalar optimization mentioned above is solved using Evolutionary Programming. In Evolutionary Programming the random number population is generated and the chromosomes are selected based upon the maximum fitness of the fitness function using the roulette wheel selection.

4.4.6 Algorithm for Hamming Distance Minimization using EP

Step 1 Compute filter coefficients $h_I(n)$ and freq. response $H_I(\omega)$ of ideal FIR filter for $0 \leq n \leq N-1$.

Step 2 Calculate the Hamming distance (HD_I) between the FIR filter coefficients $h_I(n)$.

Step 3 Set the Number of chromosomes *(k),* mutation rate *(m),* Stopping criteria.

Step 4 Populate *k* sets of possible designed solutions, to produce symmetric coefficients $H_D(n)$ $0 \leq i \leq k-1$ *and* $0 \leq n \leq N-1$

Step 5 Compute the frequency response of the coefficients chromosomes for $H_D(\omega)$ in population.

Step 6 Calculate the Hamming distance in each of the coefficient chromosome.

Step 7 Evaluate the fitness of the chromosomes

$$f(i) = f_M + f_H$$

Step 8 Apply Roulette wheel selection.

Step 9 Mutate at a desired rate.

Step 10 Evaluate again the fitness of the chromosomes.

Step11 If the Stopping criterion is met store the chromosomes according to the fitness, Else go to Step 8.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter presents the results of Hamming distance minimization by using the Steepest Decent and Evolutionary Programming algorithms as discussed in chapter 3, 4. The algorithms are implemented under the MAT LAB environment on two different FIR Filters. These filters vary in terms of the desired filter characteristics and consequently in the number of coefficients. These filters have been designed using the window method. The filters are designed using different widow for every filter; first filter is designed using Hamming window function, second using Blackman widow function. The coefficient values quantized to 16 bit 2's complement representation from the initial set of coefficients for optimization. The results are presented using '*Evolutionary Programming'* and are compared with *'Steepest Descent'* algorithm.

5.1 Hamming distance minimization using Steepest Descent method

Steepest decent search method produces a local optimal point in the neighborhood with respect to the reference point. In this approach new values of coefficients are searched such that the Hamming distance between the new values is optimized.

The names of the filters indicate the filter characteristics. The FIR filter having the following characteristics: Sampling frequency =16 KHz, passband frequency = 3 KHz, stopband frequency = 4 KHz, passband ripple = 0.1 db, stopband attenuation = 62db and number of coefficients = 30. The results are presented in terms percentage of Hamming distance reduction and number of signal toggles while satisfying the desired filter characteristics in terms of pass band ripples and stopband attenuation retaining and also the linear phase of the filter. Figures in the next section shows the frequency domain characteristics of FIR filters for different number of taps and frequency specifications corresponding to new and initial coefficients optimized with linear phase The ideal and optimized filter response is given in blue and red respectively.

5.1.1 Low pass filter

Sampling Frequency =16 KHz

Passband Frequency =3 KHz

No. of Coefficients = 30

Stop band frequency = 4 KHz

Pass band ripple = 0.1 db

Stop band attenuation = 62db

Initial hamming distance = 224

Final hamming distance =170

Hamming distance reduction = 54

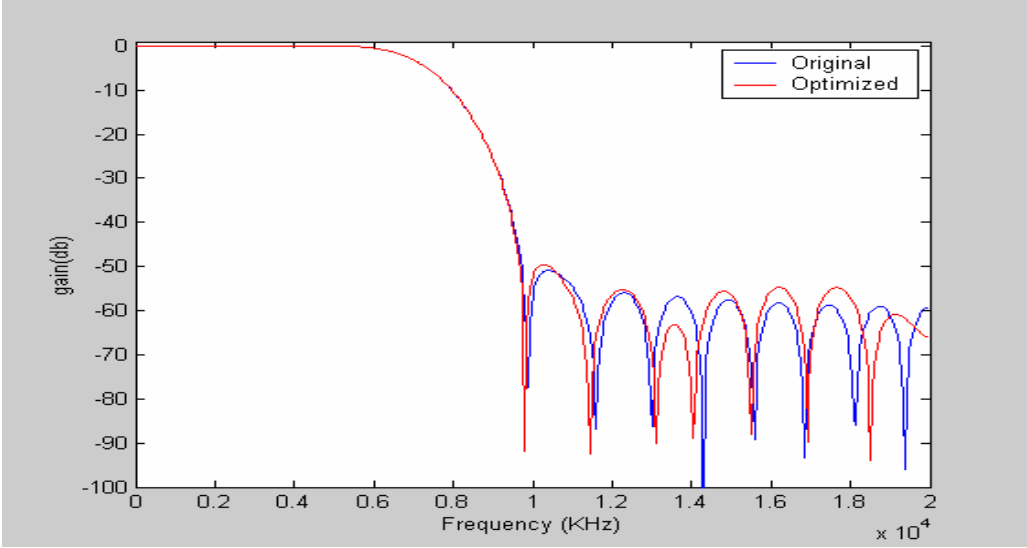

Figure 5.1 Original and Optimized Response of Lp_16K_3K_4K_.1_62_30 FIR Filter

5.1.2 Low pass filter

Sampling Frequency =10 KHz

Passband Frequency =2 KHz

No. of Coefficients = 29

Stop band frequency = 3 KHz

Pass band ripple = 0.5 db

Stop band attenuation = 40db

Initial hamming distance = 208

Final hamming distance = 168

Hamming distance reduction = 40

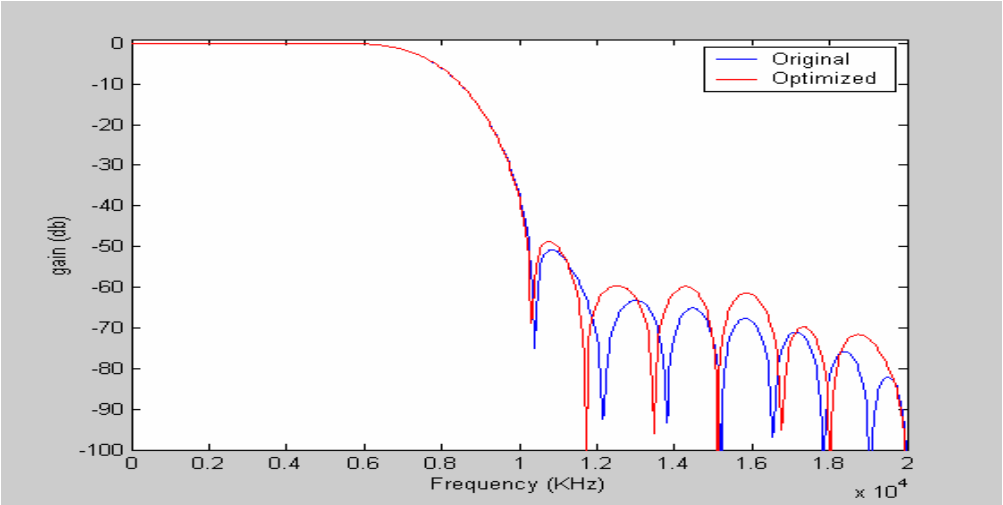

Figure 5.2 Original and Optimized Response of Lp_10K_2K_3K_.05_40_29 FIR Filter

The results of two FIR filters in terms of percentage Hamming distance and number of signal

Toggling reduction using steepest decent Approach is summarized in a table given below.

FIR Filters	Initail		Steepest Desent		% Reduction	
	HD	Togs	HD	Togs	HD	Togs
Lp_16K_3K_4K_.1_62_30	224	43	169	29	24.91%	30.95%
Lp_10K_2K_3K_.05_40_29	208	35	168	20	19.23%	42.86%

Table 5.1 Results for four FIR filters using Steepest Descent Approach

The results show that with linear phase constraint upto 25% reduction in the total Hamming distance and upto 45% reduction in adjacent signal toggling in opposite direction (Togs) is achieved using steepest decent technique.

5.2 Hamming distance minimization using Evolutionary programming

The Hamming distance minimization problem using evolutionary programming is formulated as multiobjective minimization problem which minimizes the mean square error and Hamming distance simultaneously between the ideal and desired FIR filter. To generate the non inferior solutions, weighting method is used which convert the multiobjective problem into single objective problem by assigning the appropriate weights to both the objectives. The solution of the scalar objective problem is achieved by evolutionary programming. The non inferior solutions are generated by making the

trade off between two objectives. The procedure is repeated for various simulated weight combinations to get number of solutions. Various Solutions generated with respect to various weight combinations are given in Appendix. The best solution is selected on the basis of best fitness and Hamming distance reduction or corresponds to the minimum square error and maximum Hamming distance reduction. Evolutionary Programming is applied to the 2 FIR filters with same specification in section (5.1).

5.2.1 Low pass filter

Sampling Frequency =16 KHz

Passband Frequency =3 KHz

No. of Coefficients = 30

Stop band frequency = 4 KHz

Pass band ripple = 0.1 db

Stop band attenuation = 62db

Initial hamming distance = 224

Final hamming distance =146

Hamming distance reduction = 78

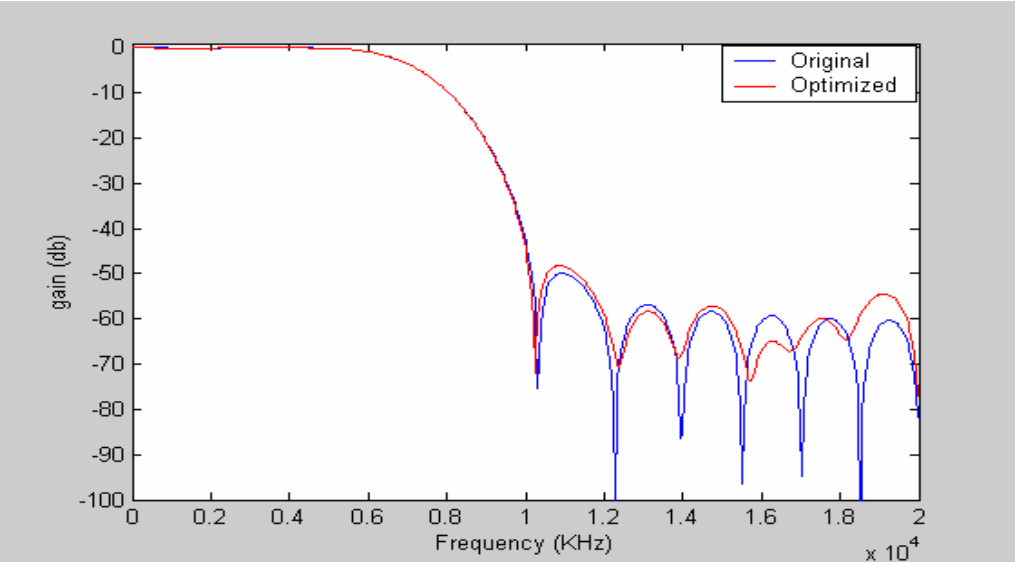

Figure 5.3 Original and Optimized Response of Lp_16K_3K_4K_.1_62_30 FIR Filter

5.2.2 Low pass filter

Sampling Frequency =10 KHz

Passband Frequency =2 KHz

No. of Coefficients = 29

Stop band frequency = 3 KHz

Pass band ripple = 0.5 db

Stop band attenuation = 40db

Initial hamming distance = 208

Final hamming distance = 132

Hamming distance reduction = 76

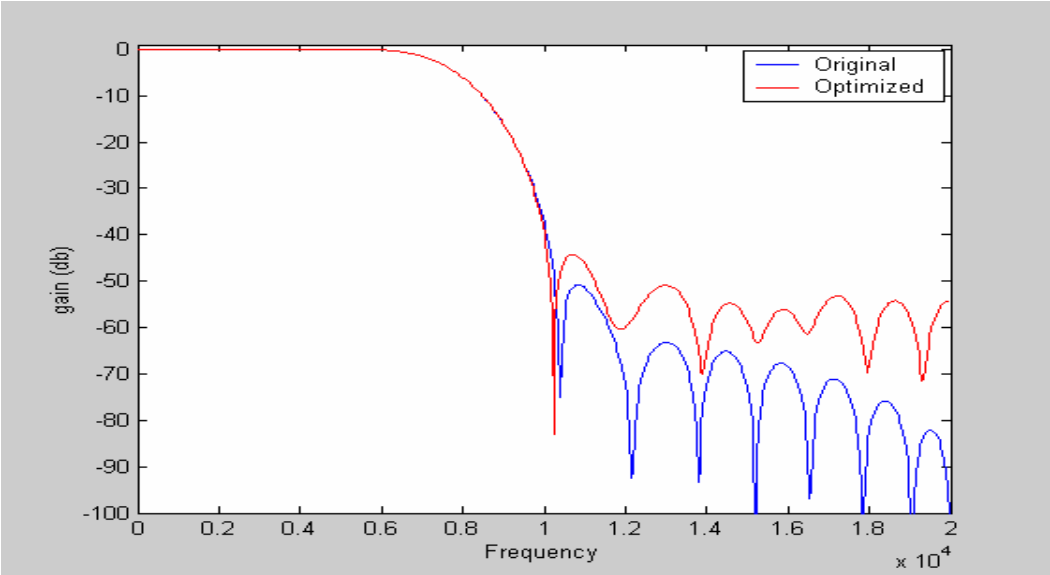

Figure 5.4 Original and Optimized Response of Lp_10K_2K_3K_.05_40_29 FIR Filter

The results of four FIR filters in terms of percentage Hamming distance and number of signal toggling reduction evolutionary programming are summarized in a table given below.

FIR Filters	Initial		Evolutionry prog		% Reduction	
	HD	Togs	HD	Togs	HD	Togs
Lp_16K_3K_4K_.1_62_30	224	43	146	19	34.82%	55.81%
Lp_10K_2K_3K_.05_40_29	208	35	132	18	36.54%	48.57%

Table 5.2 Results for Four FIR Filters Using evolutionary programming

The results show that with linear phase constraint upto 36% reduction in the total Hamming distance and upto 61% reduction in adjacent signal toggling in opposite direction (Togs) is achieved using evolutionary programming. The comparison results of Hamming distance minimization of two techniques is given in table below:

FIR Filters	% Reduction using Steepest Decent technique		% Reduction using GA	
	HD	Togs	HD	Togs
Lp_16K_3K_4K_.1_62_30	24.91%	30.95%	34.82%	55.81%
Lp_10K_2K_3K_.05_40_29	19.23%	42.86%	36.54%	48.57%

Table 5.3 Comparison of Steepest descent and evolutionary programming results

In terms of optimization strategies the evolutionary programming approach performs better in most of the cases than steepest decent approach. Evolutionary programming searches the whole optimization space to find a global value so the better Hamming distance reduction is achieved with this technique. The comparison of two techniques show that much better results for Hamming distance and signal toggle minimization are obtained by using evolutionary programming.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

In this present work Hamming distance minimization algorithms are presented for the low power realization of FIR filters on programmable DSPs. Analysis is presented to show that the Hamming distance and the total number of adjacent signals toggling between successive values, forms the main measure of power dissipation.

Optimization algorithms are presented in detail so as to minimize the Hamming distance and signal toggling. Two such techniques *'Steepest descent'* and *'evolutionary programming'* are presented to minimize these measures of power dissipation.

The Hamming distance minimization results for two low pass FIR filter show that the total Hamming distance can be reduced upto 25% and total number of signal toggles can be reduced upto 45% by Steepest Descent technique. For the same two FIR filters Hamming distance can be reduced upto 36% and total number of signal toggles can be reduced upto 61% by evolutionary programming. So, best optimization results are obtained by evolutionary programming. This Hamming distance reduction directly translates into power saving in multipliers while implementing FIR filter on Digital Signal Processors (DSPs).

6.2 Future Scope

Hamming distance is a common measure of power dissipation for the data buses. The minimization of Hamming distance and number of signal toggle using Fuzzy Logic and Artificial Neural Networks (ANN) with evolutionary programming is expected to receive more intension in the future.

REFRENCES

[1] Henry Samueli, *"An Improved Search Algorithm for the Design of Multiplierless FIR Filters with Power-of-Two Coefficients*", IEEE transactions circuits and systems vol. 36, no.7, July1989. pp. 1044-1047.

[2] Dusan Kodek and Kenneth Steiglitz, *"Comparison of Optimal and Local Search methods for Designing Finite Wordlength FIR Digital Filters",* IEEE transactions circuits and systems vol. CAS-28, no.1, January 1981.

[3] Farid N. Najam, *"Transition density a new Measure of Activity in Digital circuits",* IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, vol. 12, No. 2, January 1993. pp. 310-323.

[4] Anantha P. Chandrakasan, Miodrag Potkonjak, Renu Mehra, Jan Rabaey, and Robert W. Brodersen, *"Optimizing Power Using Transformations",* IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, vol. 14, No. 1, January 1995. pp. 12-31.

[5] Chetana Nagendra, Robert Michael, Owens Mary and Jane Irwin, *"Low Power Consideration in the design of Pipelined FIR filters"*, IEEE symposium on Low Power Electronics, October 9-11, 1995. pp. 32-33.

[6] Erdogan A.T. and T. Arslan, *"Low Power Multiplication Scheme for FIR Filter Implementation on Single Multiplier CMOS DSP Processors"*, IEE Electronics Letters, Vol. 32, No. 21, October.10, 1996. pp 123-125.

[7] R. Hezar and V. K. Madisetti, *"Low Power Digital Filter Implementation Using ternary Coefficients"*, IEEE Workshop on VLSI Signal Processing, Oct. 30, Nov. 1 1996. pp. 179-188.

[8] Mahesh Mehendale, S.D. Sherlekar and G. Venkatesh, *"Low Power Realization of FIR Filters Using Multirate Architectures"*, IEEE 9th conference on VLSI Design January 1996.pp 370-375.

[9] Mahesh Mehendale, S.D. Sherlekar and G.Venkatesh, *"Algorithmic and Architectural Transformations for Low Power Realization of FIR Filters"*, IEEE 11th International Conference on VLSI Design, January. 4-7, 1996. pp. 12-17.

[10] Merakos P. K., K. Masselos, O. Koufopavlou, S. Nikolaidis and C. E. Goutis, *"A Novel Transformation for Reduction of Switching Activity in FIR Filters Implementation*", IEEE 13th International Conference on Digital Signal Processing, July. 2-4 1997. pp. 653-656.

[11] Sherlekar S.D., G.Venkatesh and Mahesh Mehendale, *"Low Power Realization of FIR Filters on Programmable DSP's",* IEEE transactions on VLSI Systems, Vol. 6, No. 4, January 1998. pp 546-553.

[12] Erdogan A. T. and T. Arslan, *"Low Power Coefficient Segmentation Algorithm for FIR filter Implementation"*, IEEE Electronics letters, Vol. 34, Issue 19, Sept.17, 1998. pp. 1817-1819.

[13] Vijay Sundararajan and Keshub K. Parhi, *"Synthesis of Low Power Folded Programmable Coefficient FIR Digital Filters"*, IEEE Asia and South Pacific Design Automation Conference, January, 25-28, 2000. pp. 153-156.

[14] Zhan Yu, Meng Lin Yu, Kamran Azadet and Alan N. Willson, *"A Low Power FIR Filter Design Technique using Dynamic Reduced Signal Representation*", IEEE International Symposium on VLSI Technology, Systems and applications, April 18-20, 2001.pp. 113-116.

[15] Chang Young Han, Hyoung Joon Park, and Lee Sup Kim, *"A Low Power Array Multiplier using Separated Multiplication Technique"*, IEEE Transactions on circuits and systems-II: Analog and digital signal processing, Vol. 48, No. 9, September 2001. pp. 866-871.

[16] Harry J. M. Veendrick, "*Short Circuit Dissipation of Static CMOS Circuitry and Its Impact on the Design of Buffer Circuits*", IEEE Journal of Solid State Circuits, Vol. SC-19, No. 4, August 1984. pp. 468-473.

[17] Keshab K. Parhi, *"Approaches to Low-Power Implementations of DSP Systems"*, IEEE Transactions on circuit and system-I, fundamental theory and application, Vol. 48, no.10, October 2001.pp. 1214-1224.

[18] Youngbeom Jang and Sejung Yang, *"Low Power CSD Linear Phase FIR Filter Structure using Vertical Common Sub-expression"*, IEE electronic letters online,Vol. 38, No. 15,July18, 2002. pp. 777-779.

[19] Alberto Garcia, Lukusa D. Kabulepa and Manfred Glasner, *"Efficient Estimation of Signal Transition Activity in MAC Architectures"*, ACM's ISLPED 2002 Monterery, California, USA, August 12-14, 2002. pp. 319-322.

[20] Kyung Saeng Kim and Kwyro Lee, *"Low Power and Area Efficient FIR Filter Implementation Suitable for Multiple Taps*", IEEE Transactions on VLSI Systems, Vol. 11, No. 1, February 2003. pp 323-3325.

[21] Erdogan A.T., M. Hasan and T. Arslan, *"Algorithmic Low Power FIR Cores"*, IEE Proceedings of Circuit, System and Devices, Vol. 150, No. 3, June 2003. pp. 23-27.

[22] Jongsun Park, Woopyo Jeong, Hunsoo Choo, Hamid Mahmoodi Meimand, Yongtao Wang, Kaushik Roy, *"Computation Sharing Programmable FIR Filter for Low Power and High Performance Applications*", IEEE Journal of Solid State Circuits, Vol. 39, No. 2, February 2004. pp. 348-35.

[23] Sathyanarayan **S.** Rao and Arun Ramasubrahmanyan, "*Design of discrete coffficient FIR filters by simulated evolution*" IEEE Signal Processing Letters, Vol. 3, No. 5, May 1996. pp. 138-39

[24] Xiaoping Lai, "*constrained chebyshev design of FIR filters*" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: Express Briefs, Vol. 51, No. 2, February 2004. pp. 146-47.

[25] Hong Yang, Yiding Wang, *"A LBP-based Face Recognition Method with Hamming Distance Constraint*" Fourth International Conference on Image and Graphics. pp. 646-47.

[26] Dragan Cvetkovic and Ian C. Parmee, *"Preferences and Their Application in Multiobjective Optimization"*, IEEE Transactions on Evolutionary Computation, Vol. 6, No.1, February 2002. pp. 42-57.

[27] Christakis Charalambous, *"A New Approach to Multicriterion Optimization Problem and Its Application to the Design of 1-D Digital Filters"*, IEEE Transactions on Circuits and Systems, Vol. 36, No.6, June 1989. pp. 773-784.

[28] Michel. R. Lightner and Stephen. W. Director, *"Multicriterion Optimization for The Design of Electronic Circuits"*, IEEE Transactions on Circuits and Systems, Vol. CAS-28, No.3, March 1981. pp 169-179.