YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

PARTICLE SWARM OPTIMIZATION FOR ELECTRONIC CIRCUIT DESIGN AUTOMATION

Revna ACAR VURAL, M.Sc in Electronics and Communication Engineering

Prepared within the framework of the Electronics Program at the Institute of Sciences Division of Electronics and Communication Engineering

PhD DISSERTATION

Date of Defence Examination	**:** 11.02.2011
Supervisor	**:** Prof. Dr. Tülay YILDIRIM (YTU)
Members of the Examining Committee	**:** Prof. Dr. Oğuzhan ÇİÇEKOĞLU (BU)
	: Prof. Dr. M. Oruç BİLGİÇ (KU)
	: Prof. Dr. Herman SEDEF (YTU)
	: Prof. Dr. Halit PASTACI (HU)

ISTANBUL, 2011

TABLE OF CONTENTS

Page

LIST OF SYMBOLS

A_V	Gain
$c_{1,2}$	Acceleration factors
C	Capacitor
C_{ox}	Oxide capacitance per unit area
CF	Cost function
CP	Crossover probability
d_i	Desired value for i^{th} output neuron
f_{-3dB}	Cut-off frequency
fj	Activation function of j^{th} hidden neuron
f_t	Unity-gain frequency
g_{best}	Global best value
g_m	Transconductance of MOSFET
H	Passband gain
I_{bias}	Bias current
L	Gate length of MOSFET
MP	Mutation probability
N	Number of particles
net_i	Net input to i^{th} neuron
p_{best}	Personal best value
P_{diss}	Power dissipation
Q	Quality factor
R	Resistor
SL	Search limit
SN	Maximum food number
V_{tn}	Threshold voltage for NMOS
V_{tp}	Threshold voltage for PMOS
V_{DS}	Drain-to-source voltage
V_{GS}	Gate-to-source voltage
v_{IC}	Common-mode input voltage
v_{ID}	Differential-mode input voltage
V_{OS}	Input offset voltage
V_{DD}	Positive DC supply voltage
V_{SS}	Negative DC supply voltage
w	Inertia weight parameter
w_{ij}	Weight from neuron i to j
W	Gate width of MOSFET
x_{pi}	Input i for p^{th} data
φ_{ij}	Uniformly distributed real random number within the range [-1,1]
ε^p	p-dimensional vector of error
δ	Local gradient
σ	Spread parameter
ω	Angular frequency
τ_p	Propagation delay time
$\mu_{n,p}$	Charge-carrier effective mobility
λ	Channel-length modulation parameter
α	Learning rate
Δ	Deviation

LIST OF ABBREVIATIONS

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
ADC	Analog/Digital Converter
AIA	Artificial Immune Algorithm
ANN	Artificial Neural Networks
BP	Backpropagation
BSIM	Berkeley Short-channel IGFET Model
CAD	Computer Aided Design
CCII	Second Generation Current Conveyor
CMOS	Complementary Metal Oxide Semiconductor
CMRR	Common Mode Rejection Ratio
CSA	Clonal Selection Algorithm
DAC	Digital/Analog Converter
DNL	Differential Nonlinearity
DE	Differential Evolution
EA	Evolutionary Algorithm
EC	Evolutionary Computation
FET	Field Effect Transistor
FPGA	Field Programmable Gate Array
GA	Genetic Algorithm
GP	Genetic Programming
GRNN	General Regression Neural Networks
IC	Integrated Circuit
ICMR	Input Common Mode Range
INL	Integral Nonlinearity
LNA	Low Noise Amplifier
LM	Levenberg-Marquardt
LSB	Least Significant Bit
MLP	Multilayer Perceptron
MOS	Metal Oxide Semiconductor
MSE	Mean Square Error
NMOS	N-channel Metal Oxide Semiconductor
PE	Processing Element
PSO	Particle Swarm Optimization
PMOS	P-channel Metal Oxide Semiconductor
PSRR	Power Supply Rejection Ratio
RBF	Radial Basis Functions
SA	Simulated Annealing
SI	Swarm Intelligence
SPICE	Simulation Program with Integrated Circuit Emphasis
SR	Slew Rate
SSP	Static Specification Parameters
SVF	State Variable Filter
TN	Total Number of Initial Particles
TS	Tabu Search
TSMC	Taiwan Semiconductor Manufacturing Company
VCVS	Voltage Controlled Voltage Source
VLSI	Very Large Scale Integration

LIST OF FIGURES

Page

LIST OF TABLES

Page

PREFACE

This dissertation was written during my time as a doctoral candidate at the Department of Electronics and Communication Engineering, Yildiz Technical University. Throughout my doctoral studies, I was supported by several people and Scientific Research Coordination Department, Yildiz Technical University. At this point, I should express my sincere thanks.

First and foremost, I would like to thank my supervisor Prof. Dr. Tulay Yildirim for her guidance and support from the very beginning of my academic studies. It has been over eight years since she has been supervising me, facilitating my research projects and academic development. Her constant support and encouragement kept me going at times where the burden of a PhD seemed unbearable. I also would like to thank Prof. Dr. Oguzhan Cicekoglu and Prof. Dr. Oruc Bilgic for their valuable advices and instructions during my doctoral research. I owe special thanks to my dear friends and colleagues Asst. Prof. Dr. Burcu Erkmen, Asst. Prof. Dr. Nihan Kahraman and Dr. Tuba Kiyan for their support and encouragement. Furthermore, I would like to thank to Scientific Research Coordination Department, Yildiz Technical University for sponsoring the ongoing academic research project with project number: 29-04-03-KAP01.

I would like to express my deepest gratitude to my beloved family; my dearest husband Cem Vural, our precious son Berk Vural, my brother Arda Acar and my parents Ayse Rezzan Acar and Yilmaz Acar. During my doctoral studies, they encouraged me with great patience and understanding. Without their love and everlasting support, I could have never succeeded. I also would like to thank to my best friends Yasemin Engin, Ozge Saracoglu and Cagla Yilmaz who individually have always been a true sister for me.

Finally, I dedicated this thesis to my beloved grandfather Mehmet Nezihi Onen who passed away during my final months in PhD. He was very willing to witness my graduation and always had faith in my decisions. I always counted on his support and endless compassion.

ABSTRACT

PARTICLE SWARM OPTIMIZATION FOR ELECTRONIC CIRCUIT DESIGN AUTOMATION

Together with the increase in electronic circuit complexity, the design and optimization processes have to be automated with high accuracy. Predicting and improving the design quality in terms of performance, robustness and cost is the central concern of electronic design automation. Generally, optimization is a very difficult and time consuming task including many conflicting criteria and a wide range of design parameters. The evolutionary algorithms were introduced as an efficient method for exploring the search space and handling multi objective optimization problems. Therefore, fast and accurate evolutionary methods are being utilized for accommodating required functionalities and performance specifications in electronic circuit design automation area.

Particle swarm optimization (PSO) is an evolutionary algorithm technique based on the social behavior, movement and intelligence of swarms searching for an optimal location in a multidimensional search area. In this dissertation, the applicability of PSO as a global optimization tool has been investigated comprehensively for electronic circuit design automation. For this purpose, PSO was utilized for fast and optimal design of discrete and integrated circuit design having a fixed topology for a particular process technology. However, due to the rapid evolution of process technologies, the expectation for changing process technologies need to be accounted in the design cycle. Therefore, technology independent circuit design methodologies need to be developed. Considering discrete circuits, performance of PSO has been investigated, whether optimized design error will be affected in case different manufactured series are utilized for selection of components of a particular filter topology. Following, a more difficult analog design problem is considered and PSO is used in a hybrid configuration. Performance of the hybrid method is compared with different approaches for technology independent design of a digital to analog converter.

Simulation results indicate that as a global optimization tool, PSO is a very efficient method for optimal component selection and sizing task in electronic circuit design automation by means of both high accuracy and short computation time. Since once programmed, no human intervention is required (e.g. to provide an initial “good design” or to interactively guide the optimization process), the proposed method yields completely automated sizing of optimal circuits by means of both discrete and integrated design concepts.

Keywords: Particle Swarm Optimization, Artificial Neural Networks, Multi-objective Optimization, Design Automation.

ÖZET

ELEKTRONİK DEVRE TASARIM OTOMASYONU İÇİN PARÇACIK SÜRÜ OPTİMİZASYONU

Elektronik devre karmaşıklığı arttıkça, tasarım ve optimizasyon süreçlerinin yüksek doğrulukla otomatikleştirilmesi gerekmektedir. Elektronik devre tasarım otomasyonunun merkezindeki konu, tasarım kalitesinin performans, güvenilirlik ve maliyet açısından kestirimi ve iyileştirilmesidir. Genel olarak optimizasyon, çok sayıda birbiriyle çelişen kriter ve çeşitli tasarım parametrelerini içeren zor ve çok zaman alıcı bir süreçtir. Evrimsel algoritmalar, arama uzayını araştırabilecek ve çok amaçlı optimizasyon problemlerinin üstesinden gelebilecek etkili bir yöntem olarak önerilmiştir. Bu sebeple hızlı ve doğru evrimsel metotlar, elektronik devre tasarım otomasyonu alanında gerekli fonksiyonelliği ve performans tanımlamalarını sağlamak üzere kullanılmaktadır.

Parçacık Sürü Optimizasyonu (PSO), çok boyutlu arama uzayındaki en iyi konumu araştıran sürülerin sosyal davranış, hareket ve zekasını temel alan evrimsel bir algoritmadır. Bu tezde, küresel bir optimizasyon aracı olan PSO'nun elektronik devre tasarım otomasyonuna uygunluğu kapsamlı olarak incelenmiştir. Bu amaçla PSO, belirli bir üretim teknolojisi için sabit bir topolojiye sahip ayrık ve entegre devrelerin hızlı ve optimum tasarımında kullanılmıştır. Bununla birlikte üretim teknolojilerinde hızlı gelişmeye bağlı olarak, üretim teknolojilerinin gelişeceği beklentisi de tasarım sürecine dahil edilmelidir. Bundan dolayı, teknolojiden bağımsız devre tasarım yöntemleri geliştirilmelidir. Ayrık devreler dikkate alındığında PSO'nun performansı, belirli bir filtre topolojisinin eleman seçiminde farklı üretim serileri kullanıldığı durumda, optimize edilmiş devrenin tasarım hatasının etkilenip etkilenmediğine bağlı olarak incelenmiştir. Ardından daha karmaşık bir analog tasarım problemi ele alınmış ve PSO karma bir düzenleme içerisinde kullanılmıştır. Karma düzenlemenin performansı, sayısal analog çevirici devresinin teknolojiden bağımsız tasarımı için farklı yaklaşımlarla karşılaştırılmıştır.

Simülasyon sonuçları, elektronik devre tasarım otomasyonunda optimum eleman seçimi ve boyutlandırma işi için küresel bir optimizasyon aracı olan PSO'nun etkili bir yöntem olduğunu göstermektedir. Bir kez programlandıktan sonra kullanıcı müdahalesi gerekmediğinden (iyi bir başlangıç noktası bulmak, optimizasyon sürecini yönlendirmek vs...) önerilen yöntem, hem ayrık hem de entegre tasarım açısından devrelerin tamamen otomatik olarak boyutlandırmasını sağlamaktadır.

Anahtar Kelimeler: Parçacık Sürü Optimizasyon, Yapay Sinir Ağları, Çok amaçlı Optimizasyon, Tasarım Otomasyonu.

GENİŞLETİLMİŞ TÜRKÇE ÖZET

Devre tasarım süreci entegre devrenin içindeki tek bir transistörden karmaşık elektronik sistemlere kadar geniş bir aralığı kapsamaktadır. Tasarım, devrenin çalışma prensiplerinin tanımlanması ile başlar. Bu tanımlamalar, devrenin yapısı karmaşıklaştıkça daha fazla tasarım parametresi içerecek; bununla birlikte tanımlamalar da birbiriyle çelişir hale gelecektir. Arama uzayı büyüdükçe kullanıcının optimum tasarıma ulaşmak için bu süreci otomatikleştirecek bir takım araçlar kullanması gerekmektedir. Günümüzdeki karmaşık devre yapıları göz önüne alındığında, devre topolojisinin belirlenmesi, uygun değer boyutlandırma ve serim optimizasyonu için tasarım araçlarının kullanılması kaçınılmazdır. Elektronik devre tasarım otomasyonu için kullanılan bu araçların yüksek doğrulukla çalışması ve hızlı işlem yapabilmesi büyük önem taşımaktadır.

Belirli bir devre topolojisi için optimum boyutlandırma işlemi çok amaçlı optimizasyon problemi olarak tanımlanabilir. Çok sayıda tasarım parametresi içeren çelişkili tanımlamaları ve tasarım kısıtlamalarını sağlayan optimum tasarım parametrelerini elde etmek için çeşitli optimizasyon yöntemleri önerilmiştir. Klasik ve buluşsal teknikler, algoritmalarda kullanılan değişkenlere, arama uzayının boyutuna ve dışbükeylik durumuna bağımlı olduğu için özellikle optimum boyutlandırma gibi çok amaçlı optimizasyon problemleri için yetersiz kalmaktadır. Kendi kendine organize olan sistemlerin birbirleri ve çevreleri ile etkileşimini inceleyen sürü zekası temelli sistemler bahsedilen bu eksikleri gidermek üzere önerilmiştir. Parçacık Sürü Optimizasyonu (PSO), çok boyutlu arama uzayındaki en iyi konumu araştıran sürülerin sosyal davranış, hareket ve zekasını temel alan evrimsel bir algoritmadır. Bu tezde, küresel bir optimizasyon aracı olan PSO'nun elektronik devre tasarım otomasyonuna uygunluğu kapsamlı olarak incelenmiştir. Bu amaçla PSO, belirli bir üretim teknolojisi için sabit bir topolojiye sahip ayrık ve entegre devrelerin hızlı ve optimum tasarımında kullanılmıştır. Tasarım kriterleri programa tanıtılarak belirlenen sınırlar içerisinde devrenin boyutlandırma işleminin PSO algoritması tarafından yapılması hedeflenmiştir. Bununla birlikte üretim teknolojilerinde hızlı gelişmeye bağlı olarak, üretim teknolojilerinin gelişeceği beklentisi de tasarım sürecine dahil edilmelidir. Ayrık devreler açısından farklı üretim serileri ve entegre devreler açısından farklı üretim teknolojileri kullanılarak aynı devre tasarımının minimum hata kriteri ile gerçeklenmesi hedeflenmiştir. PSO algoritması ile birlikte incelenen diğer yöntemler Bölüm 2'de verilmiştir.

Ayrık devre tasarımı çerçevesinde analog aktif filtrelerin pasif eleman değerlerinin üretim serilerine uygun olarak seçilmesi, bu devrelerin kullanılabilirliği açısından çok önemlidir. Filtre tasarımında kullanılan geleneksel yönteme göre, elemanlar ideal kabul edilir ve tasarım sırasında sınırsız değer alabilir. Halbuki kapasite ve direnç gibi ayrık elemanlar sabit ve tercih edilen değerlere göre üretilirler. Filtre tasarımında eleman değerleri, tasarımda kullanılan formüle ve tasarım kriterlerine (kesim frekansı, kalite faktörü) bağlı olarak hesaplanır. Bu hesaplamayı kolaylaştırmak için birbirine eşit seçilen bazı ayrık eleman değerlerine bağlı olarak elde edilen diğer ayrık eleman değerleri, tercih edilen değerlerle örtüşmeyebilir. Devrenin performansı, seçilen en yakın tercih edilen değerlere bağlı olarak düşebilir ve bu durum devre için belirlenen hata kriteri değerinin artmasına sebep olur. Optimize edilmiş bir tasarım gerçekleştirmek için tercih edilen değerlerin tüm olası kombinasyonları üzerine bir arama yapmak geçerli bir çözüm değildir. Örneğin, sekiz elemanlı devre yapısı göz önüne alındığında, eleman değerleri E12 üretim serisinden dört dekat aralığında seçilecekse, arama uzayı $3x10^{13}$ nokta içerir ki; mantıklı bir süre içerisinde bilgisayar destekli arama yapılıp bu noktaların değerlendirilmesi pek mümkün değildir. Bölüm 3'te incelenen bu problemde evrimsel algoritmalar, iki farklı filtre yapısının minimum hata ile tasarımı için farklı üretim

serilerine uygun pasif elemanların optimum seçiminde kullanılmıştır. Evrimsel algoritmaların performansı, Butterworth tabanlı filtre için belirli bir üretim serisine ait pasif elemanların en düşük tasarım hatası gözetilerek seçimi bakımından incelenmiştir. Durum değişkenli filtre yapısında ise evrimsel algoritmaların seçtiği pasif elemanlarla yapılan tasarım sonucunda elde edilen toplam hata değerinin farklı bir üretim serisi kullanılması durumunda önemli ölçüde etkilenmemesi hedeflenmektedir.

Entegre devreler sayısal ve analog olarak ikiye ayrılmaktadır. Anahtarlama karakteristiği, sayısal entegre devrelerin hızını belirleyen önemli bir etkendir. Bu nedenle tasarım aşamasının başında optimize edilmesi gereklidir. Analog entegre devrelerdeki tasarım problemi ise çok sayıda çelişkili tasarım kriterine ve tasarım kısıtlamalarına uygun en küçük MOSFET alanını elde etmektir. Günümüzde yaygın olarak kullanılan sayısal tasarım araçları olmasına rağmen analog entegre devre otomasyonu halen geliştirilme aşamasındadır. Sayısal entegre tasarımında başarıyla kullanılan genel amaçlı kütüphane tabanlı yaklaşım, uygulamaya çok bağlı olan analog yaklaşımda pek geçerli olmamaktadır. Literatürde önerilen çeşitli analog entegre devre tasarım araçları, MOSFET boyutlandırma problemi için çeşitli optimizasyon yöntemlerinden faydalanmaktadır. Bu yöntemler simülasyon veya denklem tabanlıdır. Simülasyon tabanlı yaklaşımlar oldukça uzun sürmekte, denklem tabanlı olanlar da bir önceki yaklaşım kadar yüksek doğruluğa ulaşamamaktır. Dolayısıyla yüksek doğruluklu ve hızlı optimizasyon yöntemlerine ihtiyaç duyulmaktadır. Bölüm 4'te PSO algoritması hem sayısal hem analog entegre devre tasarım problemlerine uygulanmıştır. Sayısal bir devre olan CMOS eviricinin anahtarlama karakteristiğini optimize edecek CMOS boyutları PSO yardımıyla 3 farklı senaryo için elde edilmiştir. Bununla birlikte, denklem tabanlı bir yöntem olarak, PSO analog entegre devre tasarımında kullanılmış, elde edilen sonuçlar SPICE simülasyonu ve literatürde önerilen yöntemler ile karşılaştırılmıştır. Bu amaçla, farksal kuvvetlendirici ve işlemsel kuvvetlendirici yapıları incelenmiş, birbiriyle çelişen tasarım kriterlerini ve tasarım kısıtlamalarını sağlayacak optimum MOSFET boyutları PSO yardımıyla belirlenmiştir. Burada tasarım problemini PSO'ya tanıtabilmek için tasarım kriterlerini ve MOS yapıların kanal genişliklerinin (W), kanal boylarına (L) oranlarını değişken olarak içeren bir denklem oluşturulmalıdır. PSO'nun bulması gereken, bu denklemin minimum noktası ve bu minimum noktayı sağlayan tasarım kriter değerleri ve W/L oranlarıdır.

Bununla beraber, devredeki transistör sayısı arttığı zaman sadece devre denklemleriyle tasarım yapmak mümkün olmamaktadır. Karmaşık devre yapıları için, tasarımda karşılaşılan bütün idealsizliklerin ve lineer olmayan durumların temsil edilebildiği hızlı ve yüksek doğruluklu çözümler önerilmelidir. Ayrıca, üretim teknolojilerinde hızlı gelişmeye bağlı olarak, üretim teknolojilerinin gelişeceği beklentisi de tasarım sürecine dahil edilmeli ve teknolojiden bağımsız devre tasarım yöntemleri geliştirilmelidir. Çok karmaşık, uzun ya da çok sayıda düzensiz bilgi taşıyan böylesi verilerin çözümlenebilmesinde, devre tanımlama denklemlerine ihtiyaç duymadan, simülasyon sonuçları ile oluşturulan veri kümesinin giriş ve çıkışlarını haritalayabilen yapay sinir ağları (YSA) yaygın olarak kullanılmaktadır. Ağa tanıtılan örnekler için doğru çıkışları üretecek, diğer bir deyişle ağ çıkışı ile hedef çıkış arasındaki hatayı en aza indirgeyecek ağırlık değerlerinin bulunması çok boyutlu optimizasyon problemidir. Bu işlemler ağ yapısına bağlı olarak tek adımda veya yinelemeli olarak gerçekleştirilebilir ve hedeflenen hata değerine ulaşıldığı durumda ağın "öğrendiği" kabul edilir. Yinelemeli işlemlerde kullanılacak olan optimizasyon algoritması, özellikle veri kümesinin büyük boyutlu olduğu durumda, problemin niteliğine uygun olarak seçilmelidir. Bölüm 5'te farklı ağ yapılarının performansı sayısal analog çeviricinin teknoloji parametrelerinden bağımsız tasarımı için incelenmiştir. İlk olarak problemi temsil edecek veri kümesi oluşturulmuştur. Daha sonra çok büyük boyutlu bu problemin çözümünde hem tek

adımda hem de PSO algoritmasının yanı sıra farklı klasik optimizasyon algoritmalarının kullanıldığı yinelemeli öğrenen ağ yapıları kullanılmış ve performansları ayrıntılı olarak incelenmiştir.

Bu tezde PSO algoritmasının elektronik devre tasarımında kullanılabilirliği farklı özellikte ve farklı boyutta tasarım problemleri açısından incelenmiştir. Türev hesabı gerektirmemesi ve ayarlanacak parametre sayısının az olması algoritmayı hızlandırırken, parçacıkların birbirleriyle ve çevreleriyle etkileşim prensibi arama uzayının etkili olarak araştırılmasına ve dolayısıyla küresel optimum noktanın keşfedilmesini sağlamaktadır. Simülasyon sonuçları ve diğer yöntemlerle yapılan karşılaştırmalar, elektronik devre tasarım otomasyonunda optimum eleman seçimi ve boyutlandırma işi için küresel bir optimizasyon aracı olan PSO'nun etkili bir yöntem olduğunu göstermektedir. Bir kez programlandıktan sonra kullanıcı müdahalesi gerekmediğinden önerilen yöntem, hem ayrık hem de entegre tasarım açısından devrelerin tamamen otomatik olarak boyutlandırmasını sağlamaktadır.

1. INTRODUCTION

The process of circuit design can cover systems ranging from complex electronic systems all the way down to the individual transistors within an integrated circuit and begins with the specification, which states the functionality that the finished design must provide, but does not indicate how it is to be achieved (Sherwani, 1999).The initial specification is basically a technically detailed description of what the customer wants the finished circuit to achieve and can include a variety of electrical requirements, such as what signals the circuit will receive, what signals it must output, what power supplies are available and how much power it is permitted to consume. The specification can also set some of the physical parameters that the design must meet, such as size, weight, moisture resistance, temperature range, thermal output, vibration tolerance and acceleration tolerance. Given a set of design specifications, a designer first initiates the circuit design by determining a circuit topology. It is most frequently the case that the designer modifies a known topology rather than creates a completely new one. The designer usually selects from well-characterized topologies or previously adopted ones, or builds one by combining them. Decision criteria rely on rough performance estimations of circuits. Next, the designer determines the values of designable parameters of a selected topology, such as transistor sizes and component values, to satisfy given specifications. Early in this stage, the designer may use analytical equations that express the relations between circuit performance and design parameters, and assign approximate values to the design parameters. Later these values are refined according to accurate performance evaluations performed by a circuit simulator such as SPICE.

Several types of parameters $p=(x,s)$ influence the dynamic and static behavior of electronic circuits and have to be taken into account when optimizing appropriate performance functions $f(p)$: design parameters x and manufacturing process parameters s. During optimization, performance function $f(p)$ should be minimized while also several constraints have to be satisfied. The performance function $f(p)$ and the constraint functions $c(p)$ can be costly to evaluate and are subject to noise (i.e. due to numerical integration effects). For both, the dependency on p can be highly nonlinear (Maten et al., 2006). Once the specialized constraints of particular circuit design problem have been properly formulated as an optimization problem, the techniques that then are utilized are totally unspecialized in character. Optimization thus provides a truly general design engineering scheme (Fig.1.1) and extensively used in following electronic design areas (Massara, 1991):

- Design accommodating complex device models
- Design accommodating non-standard specification constraints (minimum component spread, minimum sensitivity to component variations, maximum yield, etc.)
- Large-scale analog and digital circuit design; printed circuit board and interconnection; integrated circuit design and layout (e.g., minimum interconnection length, optimum placement of major functional blocks)
- Distributed device design (microwave, surface acoustic waveform structures, etc.)
- Passive component and topology selection in analog and digital filter design
- Neural network training for applications in electronic engineering

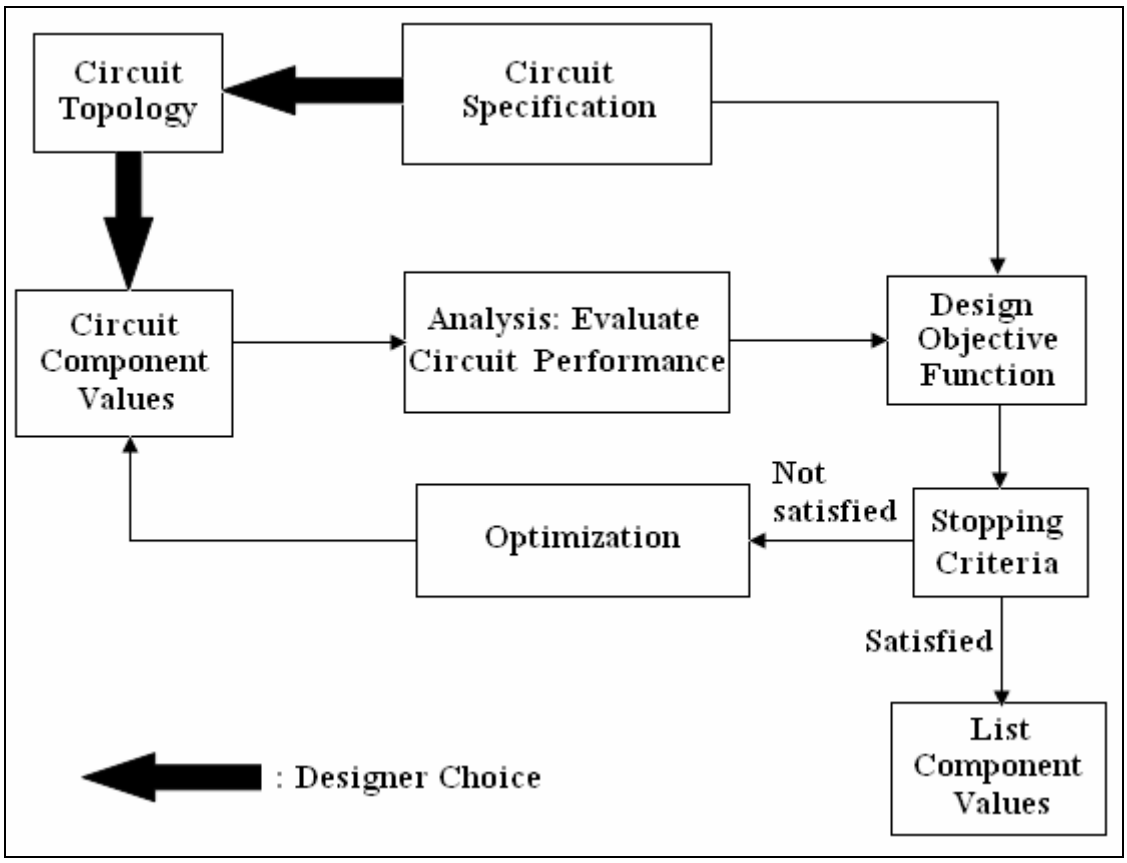

Fig. 1.1 Optimization scheme for electronic circuit design (Massara, 1991)

Integrated circuits (IC) are characterized by complex tradeoffs between multiple nonlinear objectives while satisfying multiple nonlinear and sometimes non convex constraints. Typical of IC objective and/or constraint functions are area, delay, and power. Expressed as functions of the usual IC designable parameters, such as stripe widths and supply voltages, these objectives are frequently, and, in physical limits, inherently, in competition with one another.

As an example wideband amplifier design may have performance functions as the noise figure and the bandwidth of the amplifier and specifications may be set that the input and output impedances are within 5 percent of 50 Ω over the band of the amplifier. Here, there exists two criteria to be evaluated and these are clearly conflicting. Moreover the specifications over the input and output impedances have to be satisfied for all frequencies in the band of the amplifier. Thus IC design problem with conflicting criteria can be characterized as constrained multi objective optimization problem. (Brayton et al., 1981)

Together with the increase in circuit complexity, the design and optimization complexity of today's ICs has increased drastically. Obtaining the optimal solution for a particular combination of design criteria by hand is unaffordable and time consuming. Selection of independent design parameters is quite important in design process. Ideally all design parameters are accepted as variables and the optimal solution is searched. However, tremendous growth of search space makes the search process inefficient. In addition; there are several relations that should hold between certain L, W, and W/L ratios to make the search space smooth and the optimization process reliable. Therefore efficient optimization methods for automation of optimal sizing of MOS transistors are required. Classical approaches are either deterministic or statistical-based techniques. Deterministic methods, such as Simplex (Nelder & Mead, 1965), Branch and Bound (Land and Doig, 1960), Goal Programming (Scniederjans, 1995), Dynamic Programming (Bellman, 2003) are effective only for small size problems. They are not efficient when dealing with multi-criteria problems. In addition, it has been proven that these optimization techniques impose several limitations due to their inherent solution mechanisms and their tight dependence on the algorithm parameters. Besides they rely on the type of objective, the type of constraint functions, the number of variables and the size and the structure of the solution space. Moreover they do not offer general solution strategies. Most of the optimization problems require different types of variables, objective and constraint functions simultaneously in their formulation. Statistical methods generally start with finding a "good" DC quiescent point, which is provided by the skilled analogue designer. Following, a simulation-based tuning procedure takes place. However these statistic-based approaches are time consuming and do not guarantee the convergence towards the global optimum solution (Talbi, 2002). Therefore, classic optimization procedures are generally not adequate. Heuristics are necessary to solve big size problems and/or with many criteria (Basseur et al., 2006). They can be 'easily' modified and adapted to suit specific problem requirements. Even though they don't guarantee to find in an exact way the optimal solution(s), they give 'good' approximation of it (them) within an

acceptable computing time (Chan and Tiwari, 2007). Some mathematical heuristics that were previously utilized were Local Search (Aarts and Lenstra, 2003), Simulated Annealing (SA) (Kirkpatrick et al., 1983; Siarry et al., 1997), Tabu Search (TS) (Glover, 1989; Glover, 1990), Genetic Algorithms (GA) (Grimbleby, 2000; Dreo et al., 2006), etc.

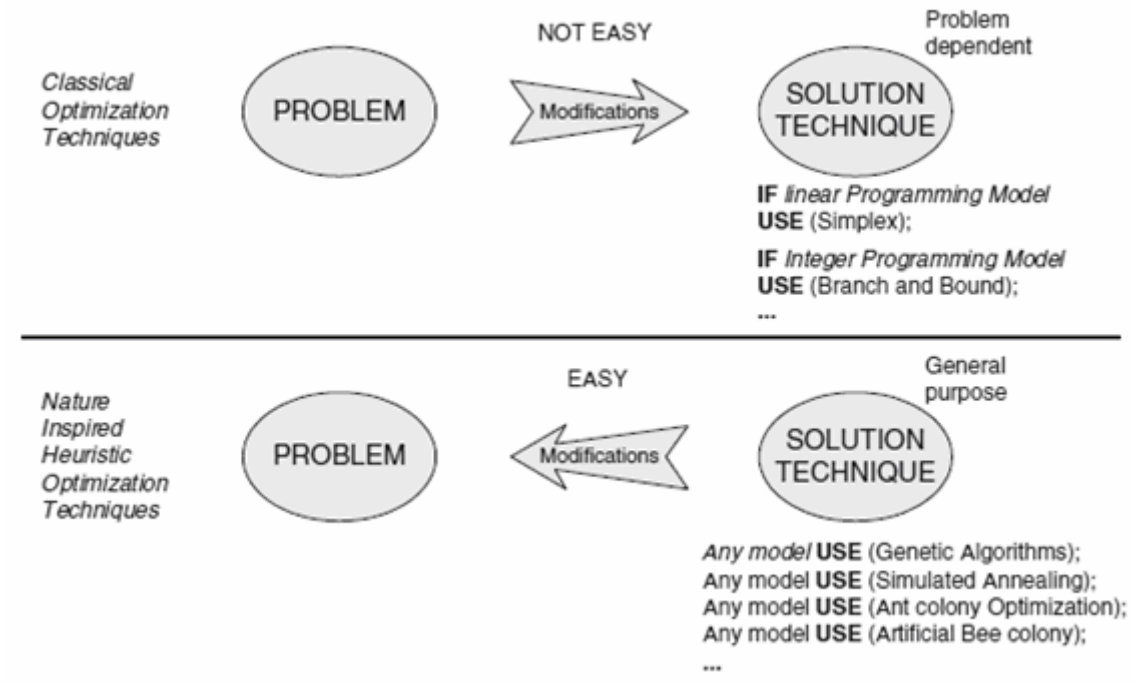

Fig. 1.2 Pictorial comparison of classical and modern heuristic optimization strategies (Chan and Tiwari, 2007)

However these techniques do not offer general solution strategies that can be applied to problem formulations where different types of variables, objectives and constraint functions are used. In addition, their efficiency is also highly dependent on the algorithm parameters, the dimension of the solution space, the convexity of the solution space, and the number of variables. Actually, most of the circuit design optimization problems simultaneously require different types of variables, objective and constraint functions in their formulation. Hence, the abovementioned optimization procedures are generally not adequate or not flexible enough. In order to overcome these drawbacks, a new set of nature inspired heuristic optimization algorithms were proposed. The thought process behind these algorithms is inspired from the collective behavior of decentralized, self-organized systems. It is known as Swarm Intelligence (SI) (Bonabeau et al. 1999). SI systems are typically made up of a population of simple agents interacting locally with each other and with their environment. These particles obey to very simple rules, and although there is no centralized control structure dictating how each particle should behave, local interactions between them lead to the emergence of

complex global behavior. Most famous such SIs are Ant Colony Optimization (ACO) (Dorigo et al., 1999), Artificial Bee Colony optimization (ABC) (Karaboga, 2005) and Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995; Clerc, 2006). PSO, in its current form, has been in existence for almost a decade, which is a relatively short period when compared to some of the well known natural computing paradigms, such as evolutionary computation. PSO has gained widespread demand amongst researchers and has been shown to offer good performance in an assortment of application domains (Banks et al., 2007).

Above mentioned optimization methods are incorporated into computer-aided design tools (CAD) for optimal sizing of complex ICs together with topology selection (Maulik et al., 1992) and actual circuit layout (Harvey et al., 1992). CAD tools are the key to managing this increased complexity while meeting the shortening time-to-market factor. Predicting and improving the design quality in terms of performance, robustness and cost is a central concern of computer aided circuit design. CAD tools are also needed to assist or automate many of routing and repetitive design tasks, taking away the tedium of manually designing these sections and providing the designer with more time to focus on the creative aspects of the design (Gielen and Rutenbar, 2000).

The analog design medium has proven to be a formidable domain to its digital counterpart for many high performance IC applications. In digital domain, CAD tools are fairly well developed and commercially available today, certainly for lower level of the design flow. While digital synthesis tools have evolved in phase with digital revolution, analog portion of design automation has not been able to keep up with its demand. Some of the main reasons for this lack of automation are that analog design in general is perceived as less systematic and more heuristic and knowledge-intensive in nature than digital design. The variety of circuit schematics and number of conflicting requirements and corresponding diversity of device sizes are also much larger. These differences from digital design explain why analog CAD tools cannot simply adapt digital algorithms but why specific analog solutions need to be developed that are targeted to the analog design paradigm and complexity. Historically, researchers developed two mainstreams of analog automation methodologies. One of the early approaches uses optimization-based method which optimizes a set of performance constraints characterized by complicated trade-offs and makes extensive use of detailed circuit simulator embedded in the inner loop of optimization engine. This mechanism works quite well for tuning purpose of the circuit that is already close to a good design. These techniques require many iterations to adjust transistor sizes and optimization engine needs to evaluate

corresponding performance at each cycle. This procedure is very time consuming and computationally expensive task. Second approach is equation-based method which is based on inverse process of circuit analysis technique. Since sizing of a circuit is done mathematically, the automation is much faster while accuracy is not as good as the first approach due to the simplified device equations and approximations. Comparison of previously proposed analog CAD tools is given in Table 1.1 (Gielen and Rutenbar, 2000; Lee and Kim, 2006).

Table 1.1 The error rate and synthesis time of various analog CAD tools (Lee and Kim, 2006)

Tool	**Synthesis Method**	**Error**	**Synthesis Time**
IDAC (Degrauwe et al.,1987)	Equation-based	15%	Few seconds
OASYS (Harjani et al., 1989)	Equation-based	25%	Few seconds
ISAID (Toumazou et al.,1990)	Equation-based + post optimization	14%	Not reported
STAIC (Harvey et al., 1992)	Equation-based	24%	3 min
DELIGHT.SPICE (Nye et al., 1988)	Optimization-based (Circuit simulator)	0%	18 h
MEALSTROM (Krasnicki et al., 1999)	Optimization-based (Circuit simulator)	0%	3.6 h
ASTRX/OBLX (Ochotta et al., 1996)	Optimization-based (AWE+equations)	30%	11.8 h
OPASYN (Koh et al., 1990)	Optimization-based (equations)	20%	1 min
ASLIC (Lee and Kim, 2006)	Equation-based	15-20%	Few seconds

Among the CAD tools tabulated above, STAIC, DELIGHT.SPICE, and OPASYN utilize classical optimization techniques while IDAC, MAELSTROM, ASTRX/OBLX, ASLIC, and OASYS are heuristic based systems. Kruiskamp and Leenaerts (1995) developed a GA based CMOS operational amplifier synthesis tool (DARWIN) for topology selection and circuit sizing. Massier et al. (2008) proposed sizing rules method for CMOS and bipolar analog IC synthesis. Sripramong and Toumazou (2002) introduced an automated circuit design system for the evolution and subsequent invention of CMOS amplifiers. This system utilized genetic programming for evolving new circuit topologies and current-flow analysis for screening and correcting circuits. Hershenson et al. (2001) proved that the design of CMOS op-amp can be

well approximated as polynomial convex optimization problem that can then be solved using geometric programming techniques, producing a close-by-first-cut design in an extremely efficient way. In (Liu et al., 2009), an evolution-based methodology named memetic single-objective evolutionary algorithm is developed for automated sizing of high-performance analog IC circuits. Guerra-Gomez et al. (2009) proposed multi-objective evolutionary algorithm based on decomposition (MOEA/D) for optimization of second generation current conveyors (CCII). Mentioned system uses HSPICE as circuit evaluator. Considering optimal CCII design without any circuit evaluator; a multi-objective heuristic (Salem et al.,2006; Fakhfakh et al., 2009a) and PSO algorithm (Cooren et al., 2007; Fakhfakh et al., 2009b; Fakhfakh et al., 2010) are utilized by formulating the requirements for the design of CCII in terms of boundaries on performance functions. Tawdross and König (2005) investigated PSO as an alternative to GA genetic algorithm for field programmable analog scalable device array reconfiguration. For this purpose an operational amplifier with particular design constraints was designed using PSO taking into different external influences such as high temperature and fabrication faults. Having successful results authors extended their PSO based dynamic hardware design environment to functional block level (Tawdross and König, 2006). A 3-bit ADC structure is developed using previously designed op-amps and resistors. Thakker et al. (2009) extended PSO algorithm to a hierarchical scheme for automatic sizing of low power analog circuits where simulation of circuits is performed with Cadence Spectre. Microwave circuit design is another area where optimization tools were widely used. Considering less computation time, PSO was chosen as the synthesis procedure for two sets of microwave circuits: micro stripe coupler and single shunt stub matching circuit (Ulker, 2008). Tulunay and Balkir (2008) proposed an automatic synthesis tool of a cascade low noise amplifier (LNA). Fakhfakh et al. (2010) utilized PSO technique for optimal sizing of CMOS LNA with inductive degeneration design. Choi and Allstod (2006) developed a SA algorithm based synthesis tool that includes an adaptive tunneling mechanism and post-optimization sensitivity analysis with respect to design, process and temperature variations. A very detailed investigation about the state of the art in applying EAs for the synthesis and sizing of analog ICs was presented by Tlelo-Cuautle et al. (2010).

Considering digital ICs, circuit delays often need to be reduced to obtain faster response times, with a minimal area penalty. A typical digital IC consists of multiple stages of combinational logic blocks that lie between latches, clocked by system clock signals. Delay reduction must ensure that the worst-case delay of the combinational blocks is such that valid signals reach a latch before any transition in the signal clocking the latch, with allowances for

set-up time requirements. In other words, the worst-case delay of each combinational stage must be restricted to be below a certain specification. Given the MOS circuit topology, the delay can be controlled by varying the sizes of transistors in the circuit. Here, the size of a transistor is measured in terms of its channel width, since the channel lengths in a digital circuit are generally uniform. Roughly speaking, the sizes of certain transistors can be increased to reduce the circuit delay at the expense of additional chip area (Sapatnekar, 1993).

As an optimization tool, PSO was utilized in field programmable gate arrays (FPGA) placement and routing for minimization of the distances between configurable logic blocks (Gudise and Venayagamoorthy, 2004). Venayagamoorthy et al. (2007) proposed PSO algorithm as a method for determining the parameters for circuit partitioning problem and this method optimized the critical path as well as the number of partitioning and test vectors. Gate level design of combinational logic circuits is another area where PSO was successfully utilized for minimizing the number of gates during realization of 100% feasible circuits (Coello Coello, 2003; Moo Moore and Venayagamoorthy, 2005; Moore and Venayagamoorthy, 2006). Vural et al. (2010c) utilized PSO for inverter design having symmetric output response. Having obtained successful results authors enhanced their work to a more detailed dynamic characterization of inverter design framework considering three different case studies (Vural et al., 2010d).

So far it has been mentioned about optimization in analog and digital IC design. Despite the extensive usage of ICs, discrete components are still preferred in analog active filter design. Conventionally, the values of passive components used in the active filters are chosen as equal to each other. This approach simplifies the design procedure but also limits the freedom of design. Moreover, components are assumed to be ideal and have infinite value during analog design process. However, discrete components such as resistors and capacitors are produced in approximate logarithmic multiples of a defined number of constant values such as E12 series. There are also E24, E48, E96 and E192 ranges for components of tighter tolerance with 24, 48, 96 and 192 different values within each decade, respectively. In order to reduce the costs and make the design more reliable, discrete components are chosen from these industrial series or other possible produced preferred values. Performing an exhaustive search on all possible combinations of preferred values for obtaining an optimized design is not feasible when components are selected from the tighter tolerance series over wide decade range. Therefore, intelligent search methods must be developed that requires short computation time with high accuracy.

The application of evolutionary techniques in filter design automation and optimization is a promising area which is based on concepts of natural selection and survival of the fittest. In (Wang et al., 2005), optimal microwave filter design with arbitrary geometries is aimed to be designed with particle swarm optimization (PSO) and finite element method. Coefficient optimization of digital filters with numerous evolutionary approaches was investigated in (Karaboga, 2009; Luitel and Venayagamoorthy, 2008). In (Zebulum et al., 1998), a detailed comparative study on analog passive filter design with different evolutionary methodologies was presented. An automated passive analog circuit synthesis procedure based on GA was utilized for the simultaneous generation of both the topology and the component value selection in (Das and Vemuri, 2007). Component value and topology evolution were also studied in (Koza et al., 1997) by means of genetic programming (GP) and in (Chang et al., 2006) by using GP based tree representation method. Unconstrained and constrained evolutions were applied towards design of analog LCR low pass filter (Sapargaliyev and Kalganova, 2006). This work has been a successful attempt to application of evolutionary strategy method for analog filter design. A robust design paradigm that exploits the open-ended topological synthesis capability of GP was developed in (Hu et al., 2005) to evolve robust low-pass and high-pass analog passive filters. In (Goh and Li, 2001), a GA based growing technique for component value optimization of analog passive filters was presented. Component value selections of analog passive and active filters were also investigated in (Sheta, 2010). Sheta (2010) explored the advantages of differential evolution (DE) over numerical optimization approaches to perform the operation of selecting the best values of circuit elements for various types of band-pass filters. In (Zebulum et al., 1999; Xu and Ding, 2009) component value selection and topology optimization of analog active filter has been performed using GA and adaptive immune GA, respectively. Moreover, some particular analog active filter types were also optimized using evolutionary approaches in the literature. A voltage controlled voltage source (VCVS) low pass Butterworth active filter circuit was designed using clonal selection algorithm (CSA) and results of CSA based design were compared with results of tabu search (TS) based, GA based, and conventional design methods (Jiang et al., 2007). In (Vural and Yildirim, 2010a), a PSO based component value selection method has been utilized for the optimal design of the same circuit topology used in (Jiang et al., 2007) and less design error was obtained when compared with results of (Jiang et al., 2007). In (Horrocks and Spittle, 1993; Kalinli, 2004; Kalinli, 2006), component values of a low pass state variable active filter (SVF) circuit was selected using GA, TS and artificial immune algorithm (AIA), respectively. However, (Horrocks and Spittle, 1993; Kalinli, 2004;

Kalinli, 2006) used a different calculation of cutoff frequency than regular and practical expression of cutoff frequency statement used in (Vural and Yildirim, 2010b).

Abovementioned electronic circuit design problems were all considered for a particular technology. However, the ability to accurately predict the performance of circuits made of MOSFETs or high-speed VLSI interconnects barely keeps pace with the technology that shrinks the FET and printed circuit board dimensions, increases the operating speed, and creates new devices. The arising problem is that as their dimensions are reduced or a new device is developed, the old device or circuit models are no longer suitable. Consequently, new models have to be developed in order to use and predict the performances of the new device or circuit. This, however, is not valid for neural networks which not only being computationally very efficient but also are flexible and general. Therefore, there is a growing interest in applying the potential of neural networks to many new fields especially in CAD of VLSI circuits.

In ANN modeling of VLSI circuits, complex semiconductor equations are not required and the parameter extraction step, which represents a difficult and time consuming method, can also be omitted. Some researchers have tried to predict the transistors' sizes for basic integrated circuits where the circuit outputs are used as inputs for ANN (Wolfe and Vemuri, 2003; Jianjun et al., 2003; Pratap, 2005; Zhongliang, 2004). However, technology independent neural network modeling for VLSI design automation has not been explored until (Kahraman, 2008). Kahraman (2008) proposed a method to predict the transistor sizes of fundamental circuits that correspond to design constraints, with minimum user effort and design knowledge for a newer design technology, using neural networks. In contrast to other modeling researches, the output specifications of integrated circuits are predicted for new technology designs. A huge database was constituted using AC and DC simulations of basic analog and digital circuits varying transistor sizes with Cadence Spectre Analog Environment. This database was applied to different ANN architectures so as to estimate the transistor sizes of the circuit that meet the design constraints in new technology successfully. While predicting the new transistor sizes, ANN does not need any technology parameters. ANN just learns from the database that was constituted before, calibrates its synaptic weights, calculates an error function and predicts the output for new data (Kahraman, 2008; Kahraman and Yildirim, 2008a, 2008b). Selection of the appropriate ANN architecture for high dimensional electronic circuit design database is very important. First of all, selected structure should be able to process input information with a massive feature. Method of training should be

considered as well, since error backpropagation method would require a very long time for training high-dimensional database due to the derivation procedures. Furthermore a multi-layer perceptron (MLP) trained by error backpropagation, the most commonly used ANN, requires that the analyst determine the appropriate number of hidden layer nodes, the initial connection weights, and the form of the activation function (Currit, 2002). Recently PSO algorithm was proposed as an alternative training method for MLP. Performances of the mentioned training algorithms were compared with PSO and more accurate results were obtained with PSO in shorter computation time (Mendes et al., 2002; Zhao et al., 2005; Kuo, 2007; Junyou, 2007; Al-Kazemi and Mohan, 2002; Braendler and Hendtlass, 2002; Zhang et al., 2000). The aim is to minimize classification and generalization error (Zhao et al., 2005), mean absolute percentage error (Junyou, 2007) and/or mean square error (Junyou, 2007; Kuo, 2007; Mendes et al., 2002; Zhao et al., 2005; Al-Kazemi and Mohan, 2002; Zhang et al., 2000). Besides software applications PSO is also applied to training of a neural hardware. The challenge of training a neural hardware is to store the bit values representing weight and bias values; increasing bit values enhances the accuracy as well as the hardware complexity. Braendler and Hendtlass (2002) reported that limited number of bits utilized by PSO for weight quantization did not affect accuracy significantly and fast execution capability of PSO is very advantageous on training neural hardware. Vural and Yildirim (2008) proposed a method for ANN based optimization of ICs by utilizing PSO as a training algorithm for MLP.

This dissertation has two objectives: The first one is to explore evolutionary algorithms, particularly Particle Swarm Optimization algorithm on electronic circuit design automation. PSO-based method is applied to both discrete and integrated circuit design problems with particular technology parameters. The problem considered in this dissertation is the selection of component values and transistor dimensions, which is only a part of a complete analog circuit CAD tool. Other parts which are beyond the scope of this work are the topology selection (Maulik et al., 1992) and actual circuit layout (Harvey et al., 1992). The optimal component selection and transistor sizing of the CAD process remains between these two tasks. As reported in the literature, simulation-based optimization technique requires very long execution time and equation-based methods are less accurate than the former method. Therefore, optimization methods with high accuracy and short computation time are necessary for design automation. PSO as a global optimization method has fewer primitive mathematical operators than in GA (e.g reproduction, mutation and crossover) and those mathematical operations require more fine-tuning of own parameters which leads to longer computation time. Second objective of this dissertation is to investigate and compare ANN

methods including a PSO-trained one for a more complex circuit design problem; technology independent digital-to-analog converter (DAC) design automation.

The organization of this dissertation is as follows: Section 2 provides an insight of the evolutionary techniques utilized for both discrete and integrated circuit design cases and artificial neural network structures utilized for technology independent design of a selected DAC structure. Discrete circuit design with evolutionary algorithms is investigated in Section 3. Here, passive components of two different analog active filter structures are selected using EAs for two different manufactured series. Results are discussed in detail. Section 4 describes the PSO-based method for integrated circuit design. For this purpose, design equations of both differential amplifier with current mirror load source and two-stage operational amplifier are derived and equation-based PSO method is utilized for optimal sizing of MOS transistors for minimum occupied area. Section 5 is dedicated to technology independent neural network modeling of CMOS current-steering type DAC. Here, all ANN structures described in Section 2 are utilized for this very high-dimensional dataset and performance of the each structure is evaluated. Section 6 concludes with a discussion of simulation results and suggests possible extensions.

2. EVOLUTIONARY ALGORITHMS AND ARTIFICIAL NEURAL NETWORKS

In this section, Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and Artificial Bee Colony optimization (ABC) algorithm as evolutionary algorithms and Multilayer Perceptron (MLP), Radial Basis Function neural network (RBF) and General Regression Neural Network (GRNN) as artificial neural network structures utilized for electronic circuit design were investigated. Among evolutionary algorithms, PSO algorithm is the main concern of this thesis and is explained more detailed since its performance results reported in the literature is very promising and it has not been widely used for electronic circuit design area. PSO is also used as a training algorithm for MLP in order to investigate whether it can enhance MLP performance for very high dimensional electronic circuit design database when used instead of Backpropogation (BP) and Levenberg-Marquardt (LM) training algorithms that increases computational complexity.

2.1 Evolutionary Algorithms

In artificial intelligence, evolutionary computation (EC) refers to computer-based problem solving systems that use computational models of evolutionary processes, such as natural selection, survival of the fittest and reproduction, as the fundamental components of such computational systems (Engelbrecht, 2007).

Evolutionary algorithm (EA) is a subset of evolutionary computation, a generic population-based metaheuristic optimization algorithm. Each EA therefore maintains a population of candidate solutions. The first step in applying an EA to solve an optimization problem is to generate an initial population. The standard way of generating an initial population is to assign a random value from the allowed domain to each of the genes of each chromosome. The goal of random selection is to ensure that the initial population is a uniform representation of the entire search space. The size of the initial population has consequences in terms of computational complexity and exploration abilities. Large numbers of individuals increase diversity thereby improving the exploration abilities of the population. However, the more the individuals increase the computational complexity per generation. While the execution time per generation increases, it may be the case that fewer generations are needed to locate an acceptable solution. Smaller population sizes have the converse effect.

In order to determine the ability of an individual of an EA to survive, a mathematical function is used to quantify how good the solution represented by a chromosome is. The fitness function, *f*, maps a chromosome representation into a scalar value where *X* represents the data type of the chromosome.

$$f : X^n \rightarrow \mathbb{R} \tag{2.1}$$

The fitness function quantifies the quality of a structure, i.e. how close it is to optimality, with respect to the environment. It is therefore extremely important that the fitness function accurately models the problem, including all criteria to be optimized. Frequently the fitness function does not directly evaluate the elements of a structure, but their expression as physical features of behaviors. If the fitness function represents a cost (as usual), it should return lower values for better structures (Amoretti, 2009).

In addition to optimization criteria, the fitness function can also reflect the constraints of the problem through penalization of those structures that violate constraints. Constraints can be encapsulated within the fitness function, but also incorporated in the adaptation process.

The evolutionary operators, e.g. selection, cross-over, mutation or elitism, usually make use of the fitness evaluation of chromosomes. For example, selection operators are inclined towards the most-fit individuals when selecting parents for cross-over, while mutation leans towards the least-fit individuals. The pseudocode given in Fig.2.1 summarizes a general EA procedure (Engelbrecht, 2007).

Let t=0 be the generation counter
Create and initialize an n_x-dimensional population: P(0);

repeat
Evaluate the fitness, $f(x_i)$, of each individual, x_i, in the population, P(t);
Perform EA operation;
Select population P(t+1) of new generation;
Advance to the new generation, i.e t=t+1;
***until** stopping condition is true;*

Fig. 2.1 General EA procedures

In order to terminate the optimization process, EAs utilize a convergence criterion. Stopping conditions can be listed as below:

- A specified number of generations or iterations has been exceeded.
- There is no improvement in population fitness over a number of consecutive generations.
- An acceptable best individual is evolved.

Fig.2.1 demonstrates the major evolutionary processes; however, EA techniques differ in the implementation and operation details and the nature of the particular applied problem. Details of PSO, GA and ABC optimization methods are introduced in the following.

2.1.1 Particle Swarm Optimization

PSO is an evolutionary computation method based on the social behavior, movement and intelligence of swarms searching for an optimal location in a multidimensional search area which has been developed by Kennedy and Eberhart (1995). The approach uses the concept of population and a measure of performance similar to the fitness value used with evolutionary algorithms. Population consists of potential solutions called particles. Each particle is initialized with a random position value. In each iteration of simulation, the fitness function is evaluated by taking the current position of the particle in the solution space and two best values (p_{best}, g_{best}). Personal best value, namely p_{best}, is the location of the best fitness value obtained so far by the particle. Global best value, namely g_{best}, is the location of the best fitness value achieved so far considering all the particles in the swarm (Kennedy and Eberhart, 1995; Clerc, 2002;Kiink et. al. 2002).

In particle population matrix, containing N number of particles, i^{th} particle with a feature number of D is denoted as $x_i = [x_{i1}, x_{i2},\ldots,x_{iD}]$. For each iteration, the velocity and the position vector of the i^{th} particle in *NxD* dimension of the search space are updated as follows (Kennedy and Eberhart, 1995):

$$v_{id}^{k+1} = w.\, v_{id}^{k} + c_1.rand_1^{k}.(pbest_{id}^{k} - x_{id}^{k}) + c_2.rand_2^{k}.(gbest_d^{k} - x_{id}^{k}) \tag{2.2}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \tag{2.3}$$

Here, the range of *i*, *d* and *k* indices are defined as *{1...N}*, *{1…D}* and *{1…max_iteration_number}* respectively. The acceleration factors c_1 and c_2 indicates the relative attraction toward p_{best} and g_{best} respectively. Following $rand_1$ and $rand_2$ are random numbers uniformly distributed between zero and one. Inertia weight parameter *w* controls the tradeoff between the global and the local search capabilities of the swarm. Initially *w* should be chosen as less than one and should be decreased linearly in each iteration.

PSO algorithm used in this work has been built up for the global best (g_{best}) PSO model as shown in Fig. 2.2. The g_{best} model is chosen since it converges faster than local best (l_{best}) PSO (Engelbrecht, 2007). This is due to the larger particle connectivity of g_{best} PSO. Each

particle can interact with every other one in the swarm and can be attracted to the best position obtained by any other particle.

In (Hu et al., 2003) all particles were started with feasible solutions. Rather than such a restriction, it is experimented that during initializing process, at least one particle is started with a feasible solution and the position information of the rest is beyond the specified ranges. Hence, it is possible that as iterations are carried out, no matter how far from global solution, the position information of other particles will progress to feasible space due to update processes and may reach their best position in the specified ranges.

Generally PSO has the advantage of being very simple in concept, easy to implement and computationally efficient algorithm. Since updates in algorithm consist of simple adding and multiplication operators and no derivation operation is included, computation time is dramatically decreased compared to other heuristic algorithms. In order to avoid premature convergence PSO utilizes a distinctive feature of controlling a balance between global and local exploration of the search space which prevents from being stacked to local minimum.

```
For each particle
    Load its initial random vector;

For each particle
      Assign its initial vector as its pbest vector;

While maximum iteration is not attained
{
   Assign the first particle's pbest to gbest;
      For each particle except the first one {
         If its pbest satisfies all the constraints {
            If its pbest value fits to cost function  better than the gbest {
                              Assign its pbest as gbest;
                                                  }
                                              }
                                          }
   For each particle {
      Calculate particle velocity according to (2.2);
      Update particle position according to (2.3);
                  }
       For each particle {
          If the particle's current value and its pbest value  both satisfy the constraints {
              If the particle's current value fits the function better than its pbest value {
                  Assign its current value as its pbest value;
                                          }
                                      }

     Else if  the particle's current value and its pbest value both don't satisfy  the constraints {
          If the particle's current value fits the function  better than its pbest value          {
                  Assign its current value as its pbest value;
                                                  }
                                              }
     Else if only the particle's current value  satisfies the constraints                        {
                  Assign its current value as its pbest value;
                                                          }
                  }
}
```

Fig. 2.2 Procedures of PSO algorithm

2.1.2 Genetic Algorithm

Genetic Algorithms (GAs) are search algorithms based on the mechanics of natural selection and natural genetics which have been developed by John Holland (1975) and his students. GAs combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial strings is created using bits of the fittest of the previous generations; an occasional new part is tried for good measure (Goldberg, 2005). The process begins with a set of potential solutions or chromosomes (usually in the form of bit strings) that are randomly generated or selected. The entire set of these chromosomes comprises a population. The chromosomes evolve during several iterations or generations. New generations are generated using the crossover and mutation technique (Fig. 2.3). Crossover involves splitting two chromosomes and then combining one half of each chromosome with the other pair. Mutation involves flipping a single bit of a chromosome. Moreover, the inversion operator has the opportunity to place steps in consecutive order or any other suitable order in favor of survival or efficiency. The chromosomes are then evaluated using a certain fitness criteria and the best ones are kept while the others are discarded. This process repeats until one chromosome has the best fitness and thus is taken as the best solution of the problem. More details about GAs can be found in (Goldberg, 2005; Engelbrecht, 2007).

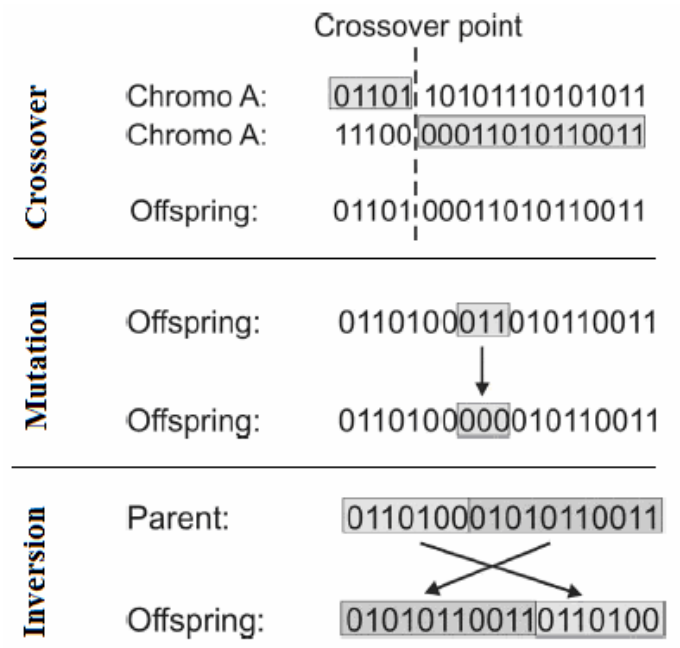

Fig. 2.3 Basic Computation Model of GA

2.1.3 Artificial Bee Colony Optimization

Artificial Bee Colony (ABC) algorithm, proposed by Karaboga (2005) for real parameter optimization, simulates the foraging behavior of bee colonies. In ABC algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. First of all, the food source positions are randomly initialized as x_i *(i=1,…,SN)* where *SN* is the maximum number of the food sources. Each employed bee, whose total number equals to the number of food sources, produces a new food source in her food source site as given in (2.4).

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \tag{2.4}$$

where φ_{ij} is a uniformly distributed real random number within the range [-1,1], *k* is the index of the solution chosen randomly from the colony and *j* is the index of the dimension of the problem. After producing v_{ij}, this new solution is compared to x_{ij} solution and the employed bee exploits a better source while each onlooker bee whose total number is equal to the number of employed bees selects a food source with the probability as given in (2.5).

$$p_i = \frac{fit_i}{\sum_{j=1}^{SN} fit_j} \tag{2.5}$$

where fit_i is the fitness of the solution x_{ij} and produces a new source in selected food source site by (2.5). After all onlookers are distributed to the sources, sources are checked whether they are to be abandoned. The employed bee associated with the abandoned source becomes a scout and makes random search in problem domain by (2.6). The best food source found so far has been memorized and the production steps are repeated until the stopping criterion is met (Karaboga,2005; Karaboga and Basturk, 2007). Procedures for ABC is given in Fig.2.4

$$x_{ij} = x_j^{\min} + (x_j^{\max} - x_j^{\min}) * rand \tag{2.6}$$

Initialize
Repeat
Move the employed bees onto their food source and evaluate the fitness
Move the onlookers onto the food source and evaluate their fitness
Move the scouts for searching new food source
Memorize the best food source found so far
Until (termination criteria satisfied)

Fig. 2.4 Procedures of ABC algorithm

2.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) are parallel operating systems and are capable of learning which the primary feature of a human brain is. The input/output training data are fundamental in ANN technology, because they convey the necessary information to discover the optimal operating point. The nonlinear nature of the neural network processing elements (PEs) provides the system with lots of flexibility to achieve practically any desired input/output map. ANNs' computation method is adaptive and fault tolerant. Moreover it is capable of deciding under uncertain conditions and handling with insufficient data (Zurada, 1995).

A basic neural network structure (Fig.2.5) consists of an input and output layer with at least one hidden layer. Each layer includes "neurons" which are the processing elements performing some specific operations. Input layer simply passes input data samples to the hidden layer. Hidden neurons in the hidden layer enable the network to learn complex tasks by extracting progressively more meaningful features from the inputs by applying a nonlinear activation function (Fig.2.6) which is differentiable everywhere. The set of output signals in the output layer of the network constitutes the overall response of the network to the activation pattern supplied by the source nodes in the input layer (Haykin, 1999).

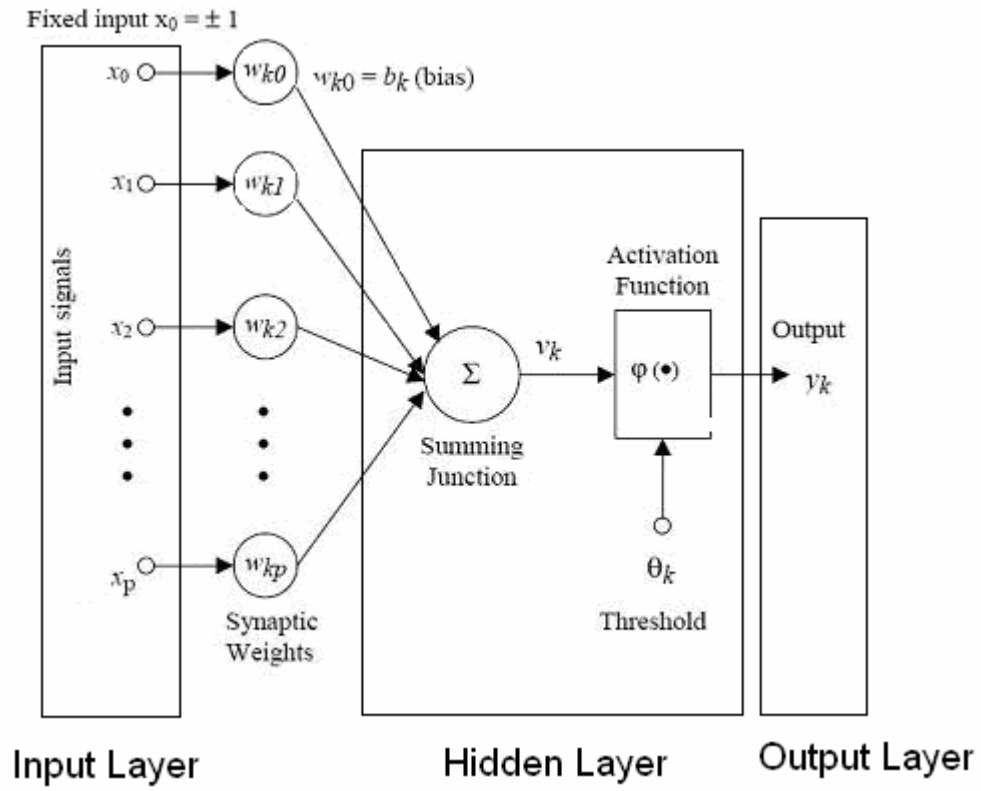

Fig. 2.5 ANN Architecture (Haykin, 1999)

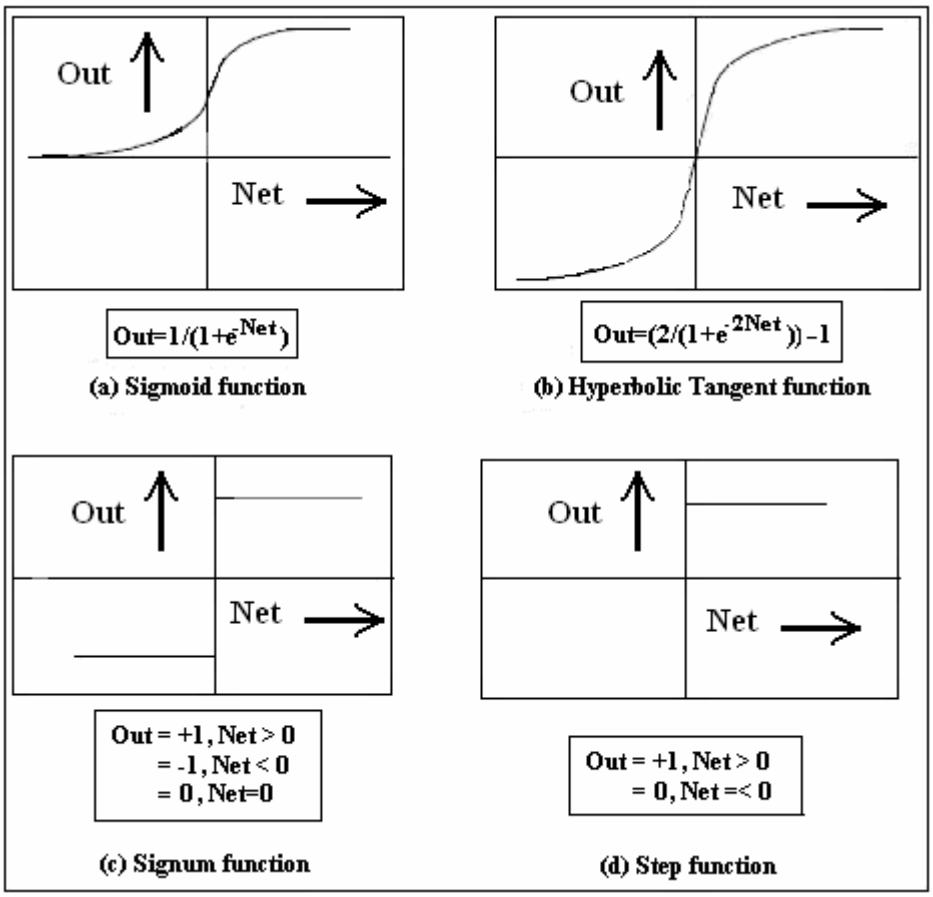

Fig. 2.6 Common nonlinear activation functions-Soft nonlinearity (a) Sigmoid and (b) Hyperbolic Tangent;- Hard nonlinearity (c) Signum and (d) Step

Interactions between layers are provided with coefficients called “weights”. The nonlinear mapping from input to output is realized by using connective weight matrixes from input layer to hidden layer and from hidden layer to output layer. Weights demonstrate the importance of knowledge at each layer and its impact over that layer. Determining the values of weights is called “the training process” of an ANN (Schalkoff, 1997).

Initially, weights are assigned random. ANN updates weights as data samples are introduced. Data samples are introduced to ANN by presenting only an input for unsupervised learning or an input together with a corresponding target response for supervised learning technique. Considering supervised learning technique, an error is composed from the difference between the target response and the system output. The aim is to obtain the weight values that can minimize the mentioned error value. Whole training data set is introduced repeatedly and this error information is fed back to the system in order to update weights to the most possible accurate values. Accurate weights ensure that ANN is able to generalize about the case that training data set represents (Haykin, 1999). Gathering generalization ability is called “the learning process” of an ANN (Fig.2.7).

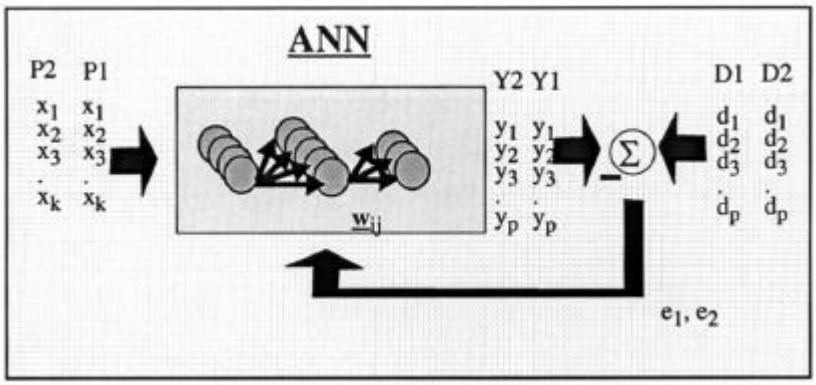

Fig. 2.7 Neural Computation [1]

Learning in ANN involves two phases. In the first phase, system output is determined considering the sample introduced to the input. According to the accuracy level of this output, weights between layers are updated as given in (2.7).

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij} \tag{2.7}$$

where w_{ij} is the weight from neuron i to j, n is some discrete, arbitrary time increment and Δw_{ij} is dependent to the training algorithm.

Completion of training process is followed by experiments for evaluating the performance of learning which is called “the testing process”. This process utilizes samples that have not been introduced to ANN during training process. Values of the weights are not adjusted during testing. First of all, test samples are introduced to the trained ANN. Trained ANN produces outputs for the test samples using weight values which were determined during training process. The accuracy of system outputs during testing proves that ANN structure has learned the case and is capable of generalizing about the case that whole training data set represents. General procedures of training and testing processes are provided in Fig.2.8

Initialize weights
Upload train inputs & train target outputs
Repeat
Calculate the weighted sum of train inputs for each hidden layer neuron
Execute nonlinear activation function in each hidden layer
Calculate the weighted sum of hidden layer outputs for each output layer neuron
Determine error between train target outputs and system output for train inputs
Update weights according the learning rule and error value
Until (error criteria satisfied)

Upload test inputs & test target outputs
Calculate system output for given test inputs according to the final weight values
Compare system outputs with test target outputs

Fig. 2.8 General Procedures of an Artificial Neural Network

2.2.1 Multilayer Perceptron

Multilayer perceptrons (MLPs) are probably the most widely used supervised feed-forward neural networks where the input signal propagates through the network in a forward direction. MLP structure utilizes at least one hidden layer depending on the complexity of the data set. This algorithm is based on the error-correction learning rule. The activation of a hidden unit (neuron *j*) is a function *fj* of the weighted inputs plus a bias, as given in (2.8).

$$x_{pj} = f_j(\sum_i w_{ji} x_{pi} + \theta_j) = f_j(y_{pj}) \tag{2.8}$$

where w_{ji} is the weight of input *i* to neuron *j*, x_{pi} is input *i*, that is, output *i* from the previous layer, for input data *p* and θj is the threshold value. The output of the hidden units is distributed over the next layer of $x_{h,2}$ hidden units until the last layer of hidden units, of which the outputs are fed into a layer of x_o output units. (Schalkoff, 1997; Haykin,1999)

Fig. 2.8 shows MLP structure with one hidden layer. The network shown here is fully connected, which means that a neuron in any layer of the network is connected to all the nodes/neurons in the previous layer.

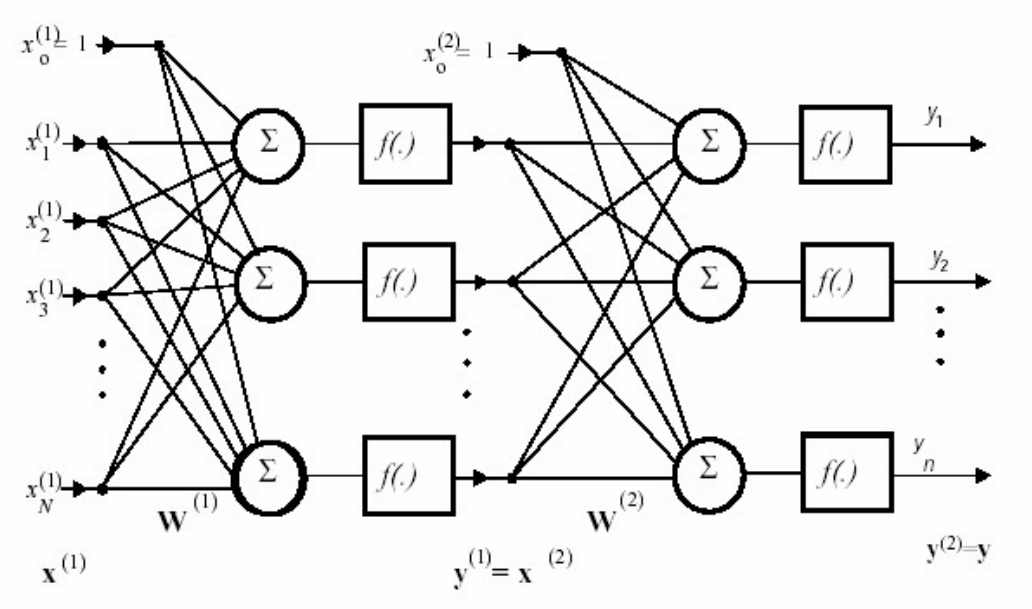

Fig. 2.9 MLP Structure

There are some issues involved in designing and training a multilayer perceptron network [2]:

- *Selecting how many hidden layers to use in the network*: For nearly all problems, one hidden layer is sufficient. Two hidden layers are required for modeling data with discontinuities such as a saw tooth wave pattern. Using two hidden layers rarely improves the model, and it may introduce a greater risk of converging to a local minimum.

- *Deciding how many neurons to use in each hidden layer*: One of the most important characteristics of a perceptron network is the number of neurons in the hidden layer(s). If an inadequate number of neurons are used, the network will be unable to model complex data, and the resulting fit will be poor. If too many neurons are used, the training time may become excessively long, and, worse, the network may *over fit* the data. When overfitting occurs, the network will begin to model random noise in the data. The result is that the model fits the training data extremely well, but it generalizes poorly to new, unseen data.

- *Finding a globally optimal solution that avoids local minimum:* A typical neural network might have a couple of hundred weighs whose values must be found to produce an optimal solution. If neural networks were linear models like linear regression, it would be a breeze to find the optimal set of weights. But the output of a neural network as a function of the inputs is often highly nonlinear; this makes the optimization process complex. If the error is plotted as a function of the weights, a rough surface with many local minimum would likely been seen. Therefore sophisticated optimization methods are required for obtaining the global minimum.

- *Converging to an optimal solution in a reasonable period of time*: Most training algorithms follow this cycle to refine the weight values: (1) Run a set of predictor variable values through the network using a tentative set of weights, (2) Compute the difference between the predicted target value and the actual target value for this case, (3) Average the error information over the entire set of training cases, (4) Propagate the error backward through the network and compute the gradient (vector of derivatives) of the change in error with respect to changes in weight values, (5) Make adjustments to the weights to reduce the error. Each cycle is called an epoch. Depending on the characteristics of the training algorithm, execution time of epochs varies. However; training should be concluded in a reasonable period of time.

2.2.1.1 Backpropagation Training Algorithm

The method of steepest descent, known in the neural network literature as backpropagation, updates the weights of the network in the direction of steepest descent. Backpropagation (BP) algorithm is perhaps the most widely use training procedure for feed forward neural networks. It is an iterative optimization of error function representing a measure of the performance of the network (Jabri et al., 1996; Haykin, 1999). Weight update is given in the following.

$$\Delta w_{ij} = \alpha . x_j . \delta_i \tag{2.9}$$

with

$$\delta_i = \begin{cases} f'(net_i).(d_i - x_i) & \text{if } i \text{ is an output neuron} \\ f'(net_i).\sum_k \delta_k . w_{ki} & \text{otherwise} \end{cases} \tag{2.10}$$

where net_i is the net input to a neuron, x_i is the output of neuron i, d_i is the desired value for output neuron i and α is the learning rate. Backpropagation requires the derivative of the neuron output with respect to the neuron input be computed which introduces computational complexity. Besides, learning should be adjusted. Moreover convergence is often excessively slow near local minimum.

2.2.1.2 Levenberg-Marquardt Training Algorithm

Levenberg-Marquardt (LM) is a variant of the Gauss-Newton optimization method which is very effective for varying smoothly between the extremes of the inverse-Hessian method and the steepest descent method. This method aims to reduce the mean square error between a model and data in the model parameter space. It works well in practice and has become the standard of nonlinear least-squares routines. The form of LM used here was 'half-Newton' in which second derivatives of the mean square error (MSE) with respect to fitting parameters are approximated.

The network weight and bias vectors are iteratively determined by Levenberg-Marquardt learning algorithm. The correction Δw_{ij} applied to w_{ij} is defined in (2.11).

$$\Delta w_{ij} = -(J^T J + \lambda I)^{-1} J^T \varepsilon(k) \tag{2.11}$$

where J is the Jacobi's matrix ($J_{pi} = \partial\varepsilon^p / \partial w_i$) and ε denotes a p-dimensional vector of error between ANN output and the desired output. For large value of λ, the algorithm approaches the gradient method (with learning step $\alpha = 1/\lambda$), for small λ it becomes the Newton method. This method has the advantages of fast result convergence with very low epoch number. The disadvantage of this method is an increased computational complexity (Dohnal, 2003).

2.2.2 Radial Basis Functions

Radial basis functions neural network structure (RBF) is a different approach by viewing the design of a neural network as a curve-fitting approximation problem in a high-dimensional space. According to this viewpoint, learning is equivalent to finding a surface in a multidimensional space that provides a best fit to the training data. The input layer is made up of source nodes. The second layer is a hidden layer of high enough dimension, which serves a different purpose from than in a multilayer perceptron. The output layer supplies the response of the network to the activation patterns applied to the input layer as given in (Fig. 2.10).

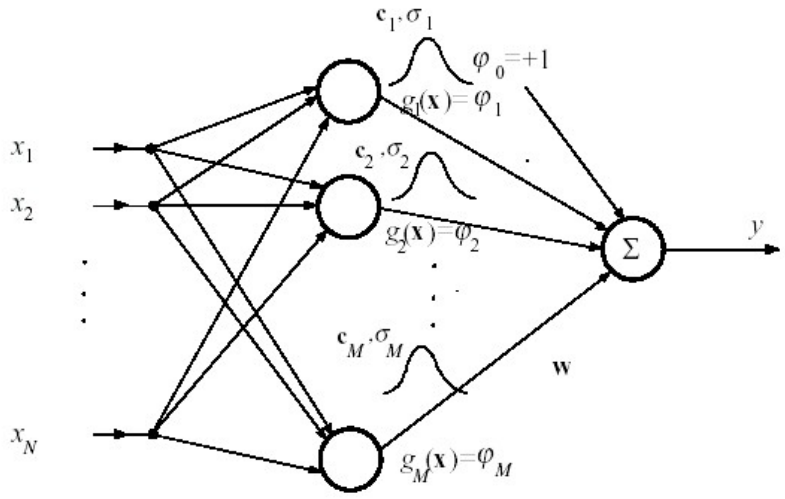

Fig. 2.10 RBF Structure

The transformation from the input space to the hidden-unit space is nonlinear, whereas the transformation from the hidden-unit space to the output space is linear (Haykin, 1999). The mathematical expression of an RBF is given in (2.12)

$$y = \sum_i w_i g_i(x) = \sum_i w_i g(\|x - c_i\|) = w^T x \tag{2.12}$$

where w_{ij} is the weight from the i^{th} neuron of the hidden layer to j^{th} neuron of the output layer. $g_i(x)$ is an activation function and in general Gaussian function. In Gaussian function,

x denotes input vector, c_i denotes center, where $\|x - c_i\|$ is the standard Euclidean distance, and σ_i as spread. Mathematical expression of Gaussian function is provided in (2.13)

$$\Phi(r) = \exp(-\frac{\|x - c_i\|^2}{2\sigma_i^2}) \tag{2.13}$$

2.2.3 General Regression Neural Network

GRNN is a memory-based network that provides estimates of continuous variables and converges to the underlying regression surface. GRNN is a one pass learning algorithm with highly parallel structure. The principal advantages of GRNN are fast learning and convergence to the optimal regression surface as the number of samples becomes very large. The learning is done by selecting the single radial basis function bandwidth that produces the lowest mean square error for only one iteration through the training data. GRNN approximates any arbitrary function between input and output vectors, drawing the function estimate directly from the training data. Furthermore, it is consistent; that is, as the training set size becomes large, the estimation error approaches zero, with only mild restrictions on the function. GRNN is particularly advantageous with sparse data in a real time environment, because the regression surface is instantly defined every where, even with just one sample (Specht, 1991a; 1991b).

The GRNN topology consists of 4 layers: the input layer, the pattern layer, the summation layer, and the output as shown in Fig.2.11.

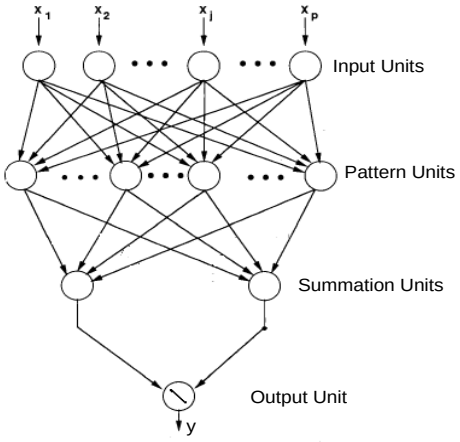

Fig. 2.11 GRNN Architecture (Specht, 1991a)

The input units are merely distributed units, which provide all of the (scaled) measurement variables to all of the neurons on the second layer, pattern units. The number of pattern layer

nodes is always equal to the number of samples in the training dataset. Each node in the pattern layer is assigned a unique training vector corresponding to one of the randomly selected training set cases. The distance is calculated between each pattern layer node vector and the input layer vector as given in (2.14) where p is the number of features, x_j is the j^{th} data value in the input vector, x_{ij} is the j^{th} data value in the i^{th} sample vector and σ is the spread parameter which is also defined as smoothing factor (Specht, 1991a; 1991b; Currit, 2002).

$$D_i = \sum_{j=1}^{p} \left(\frac{x_j - x_{ij}}{\sigma} \right)^2 \tag{2.14}$$

All pattern layer nodes are fully connected to the summation layer nodes. The summation layer nodes sum the values of all pattern layer nodes. There are two types of summation layer nodes: a numerator node and a denominator node. For those connections to the numerator node, the value of each pattern layer node is multiplied by the actual output value of that training case prior to summation. The predicted value is finally calculated in the output layer node by dividing the numerator node value by the denominator node value as given in (2.15) where N is the number of samples in the training set. (Specht, 1991a; 1991b; Currit, 2002).

$$y_{predicted}(x) = \frac{\sum_{i=1}^{N} y_{actual}{}^{i} \cdot \exp(d_i)}{\sum_{i=1}^{N} \exp(d_i)} \tag{2.15}$$

After determining the error between actual and predicted y values and depending on the optimization technique used to minimize the error between those values, the above calculation may be run numerous times with a different smoothing factor each time. Training stops once a threshold minimum error value is reached, or when the test set square error begins to rise (Specht, 1991a; 1991b; Currit, 2002).

2.3 Particle Swarm Optimization- Artificial Neural Networks: A Hybrid Structure

The type of algorithm most commonly employed for training ANNs are local-optimization algorithms. Local optimization algorithms generally require a large number of accuracy bits to perform well. This is because small variations in the error function are quantized to zero (Xie and Jabri, 1992). This leads to the formation of a large number of plateaus in the error surface in addition to those that already exist. These plateaus can be absolutely flat due to the quantization, which tends to trap the training procedure (Braendler and Hendtlass, 2002).

Weight and bias training of neural network involves actually complex continuous optimization of parameters. Notwithstanding BP has simple and flexible advantages, BP based on gradient descent is extremely sensitive to initial weight and bias vector. So, different initial weight and bias vectors may lead to totally different results or even falling into local optimum. In order to prevent the outputs turning into local optimum, the neural network generally allows for setting initial value randomly. In such case, it is required to consider if the weight and bias are optimum one after training. Trial and error method has been applied by Amir and Chuanyi (1997) to find out better initial weight settings. In addition, selection of dynamic optimal learning rate, etc., depends upon test and experience. The network cannot be converged if the value is improperly set, or even in the case of convergence, the slow convergence speed may lead to longer training and local optimum without optimal weight and bias distribution (Kuo, 2007).

Evolutionary computation (EC) methodologies have been applied to three main attributes of neural networks: network connection weights, network architecture (network topology, transfer function), and network learning algorithms. Most of the work involving the evolution of ANN has focused on the network weights and topological structure. Usually the weights and/or topological structure are encoded as a chromosome in GA. The selection of fitness function depends on the research goals. For a classification problem, the rate of misclassified patterns can be viewed as the fitness value. The advantage of the EC is that EC can be used in cases with non-differentiable PE transfer functions and no gradient information available. The disadvantages are listed below [3]:

- The performance is not competitive in some problems.
- Representation of the weights is difficult and the genetic operators have to be carefully selected or developed.

There are several papers (Junyou, 2007; Kuo, 2007; Mendes et al, 2002; Zhang et al, 2000; Zhao, 2005) that reported using PSO to replace the back-propagation learning algorithm in ANN in the past several years and stated that PSO is a promising method to train ANN. It is faster and gets better results in most cases than BP. In particular the global optimization method PSO is employed to provide a sense of the directivities of optimization of the weights and biases of neural networks, and seek a good starting weight vector for subsequent neural networks learning algorithm. It also avoids some of the problems that GA met (Braik et al., 2008). Especially, LM algorithm is slower than PSO as a training algorithm due to derivative operations performed in LM (Vilovic et al., 2009).

2.3.1 Representation of Particle Vector

Implementation of a problem in the PSO framework starts from the parameter encoding, i.e., the representation of the problem. Particle vector of PSO is constituted of the whole weight values of the neural network as given in Fig.2.12. The parameter $w_{h,i}^{HI}$ describes the weight value between the connection of the h^{th} neuron in the hidden layer and the i^{th} neuron in the input layer. Similarly, $w_{o,h}^{HO}$ represents the weight value between the connection of the o^{th} neuron in the output layer and the h^{th} neuron in the hidden layer.

$w_{1,1}^{HI}$	....	$w_{h,i}^{HI}$	$w_{1,1}^{OH}$	....	$w_{o,h}^{OH}$

Fig. 2.12 Particle vector representing weight values

2.3.2 Evaluation (Fitness) Function

Implementation of an optimization problem in PSO is realized within evolutionary process of an evaluation function. The function adopted is the mean square error (MSE) of training period that is given in (2.16).

$$MSE = \frac{1}{N}\sum_{p=1}^{N} E^{(p)} \tag{2.16}$$

where N is the total number of data samples in each epoch and $E^{(p)}$ is the instantaneous error that is given in (2.17)

$$E^{(p)} = \frac{1}{2}\sum_{o=1}^{q} e_o^{(p)^2} = \frac{1}{2}\sum_{o=1}^{q} (y_{d_o}^{(p)} - y_o^{(p)})^2 \tag{2.17}$$

with

$$y_o^{(p)} = \sum_{h=1}^{t} w_{oh}\, f\Big(\sum_{i=1}^{m} w_{hi} x_i^{(p)}\Big) \tag{2.18}$$

where q is the total number of output neurons, $e_o^{(p)}$ is the error between target output $y_{d_o}^{(p)}$ and system output $y_o^{(p)}$ at o^{th} output neuron for p^{th} data sample in the training set. f is the nonlinear activation function utilized in the hidden neurons (whose total number is t). Total number of neurons in the input layer is m which also corresponds to the dimension of each data sample represented by $x^{(p)}$ in the training and test set.

2.3.3 The Processing Steps of PSO-ANN Hybrid Algorithm

The following describes the detailed steps of PSO-ANN process:

1- Treat each weight values of MLP as the element of a particle vector.

2- Initialize particle vector with random number generator.

3- Evaluate each particle by MSE.

4- Modify g_{best} and p_{best} by simply comparing their fitness values.

5- Calculate velocities for each particle by (2.2)

6- Renew each particle to the new position by (2.3) that including random search phenomenon.

7- Evaluate modified particles by MSE.

8- If the evolution process reach to a satisfying condition (or maximum epochs are reached) then go on to step 9, else decrease inertia weight parameter, w, linearly and go back to step 4.

9- If maximum epochs are reached and a satisfying condition is obtained then go on to step 10, and if satisfying condition is not met, increase number of TC and adjust PSO design parameters and start again.

10- Utilize the best particles which minimizes MSE to a satisfying condition as the weight values for MLP and proceed with testing of the trained MLP.

2.3.4 PSO-MLP Application for Classification of Nonlinear Inputs: EXOR Problem

XOR is a connective in logic known as the "exclusive or" or exclusive disjunction. It is a logical operation on two operands that results in a logical value of true for only if one of the operands but not both has a value of true [4]. XOR is a basic dataset that is not linearly separable and is widely used to train and test ANN. The input output values for XOR are tabulated in Table 2.1.

Table 2.1 Truth table for XOR

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

PSO-MLP structure is constituted using MATLAB 7.1 without using any toolbox application. Since hybrid structure is completely adaptive, increment of the neurons depending on the case requires no hand-tuning of any variables in software.

Considering PSO-MLP hybrid structure, input layer has 2 neurons, since EXOR problem has 2 dimensions, with one bias and output layer has one output neuron. Number of neurons in the hidden layer is chosen as 5. This kind of a network configuration introduces 20 different weight values which constitutes the dimension of the particle vector. Total number of initial particles (TN) is set to 15 which is shown in (2.19). Activation function is sigmoid function and therefore weights are initialized random between 0 and 1.

$$\begin{bmatrix} w1_{11}^{HI}\ w1_{12}^{HI}\ w1_{13}^{HI}....w1_{41}^{HI}\ w1_{42}^{HI}\ w1_{43}^{HI}....w1_{11}^{OH}\ w1_{12}^{OH}\ w1_{13}^{OH}.... \\ . \\ . \\ . \\ w15_{11}^{HI}\ w15_{12}^{HI}\ w15_{13}^{HI}....w15_{41}^{HI}\ w15_{42}^{HI}\ w15_{43}^{HI}....w15_{11}^{OH}\ w15_{12}^{OH}\ w15_{13}^{OH}.... \end{bmatrix}_{15x20} \tag{2.19}$$

PSO-MLP has been run for 20 times and best results are provided in Table 2.2. Fitness function to be minimized is MSE. Maximum number of epochs is set to 50. MLP has been successfully trained with PSO for EXOR problem.

Table 2.2 Training Performance of PSO-MLP Structure

Inputs		**Target Output**	**PSO-MLP Output**	**Training Performance**	
0.1	0.1	0.1	0.1	**Network Structure**	2-5-1
0.9	0.1	0.9	0.9	**MSE**	$5x10^{-3}$
0.1	0.9	0.9	0.7	**Number of Epochs**	25
0.9	0.9	0.1	0.1	**Execution Time**	3.12 s

3. EVOLUTIONARY ALGORITHMS FOR DISCRETE CIRCUIT DESIGN

In this section, evolutionary algorithm methods described in the previous section are utilized for discrete circuit design. Component value selection is a very time consuming task when components are selected from multi-decade range for a particular manufactured series. Total error of the filter circuit may decrease when same topology is designed with different manufactured series. Therefore, performances of EA methods will be investigated on these analog active filter design problems.

3.1 Analog Active Filter Structures

Analog active filters are one of the key components in mixed-signal circuit designs and are widely used in separation of signals according to frequency bands, frequency selection decoding, estimation of a signal from noise, demodulation of signals and amplifying elements (Paarman, 2007). Analog active filters are comprised of operational amplifiers (op-amp), with resistors and capacitors in their feedback loops, to synthesize the desired filter characteristics. They can have high input impedance, low output impedance, and virtually any arbitrary gain. Possibly their most important attribute is that they lack inductors, thereby reducing the problems associated with those components. Still, the problems of accuracy and value spacing also affect capacitors, although to a lesser degree. Performance at high frequencies is limited by the gain-bandwidth product of the amplifying elements, but within the amplifier's operating frequency range, op amp-based active filter can achieve very good accuracy, provided that low-tolerance resistors and capacitors are used. Analog active filters are characterized by four basic properties; the filter type (low-pass, high-pass, band-pass, etc.), the passband gain (generally all the filters have unity gain in the passband), the cut-off frequency (the point where the output level has fallen by 3dB from the maximum level within the passband), and the quality factor Q (determines the sharpness of the amplitude response curve). A low-pass filter is a filter that passes low-frequency signals but attenuates signals with frequencies higher than the cutoff frequency. Low-pass analog active filters are widely used in biomedical instrumentation amplifiers, telecommunications and radio frequency systems (Paarman, 2007; Lacanette, 2010). Two of the low-pass analog active filter topologies used in this study is examined in the following.

3.1.1 Butterworth Filter

Butterworth filters are termed maximally-flat-magnitude-response filters, optimized for gain flatness in the pass-band. The transient response of a Butterworth filter to a pulse input shows moderate overshoot and ringing (Schaumann and Valkenburg, 2001). Ideally passband extends from 0 to ω_c and stopband extends from ω_s to ∞ where ω_c is the cutoff frequency and ω_s is the stopband frequency. Here, a low-pass Butterworth filter is of concern with passive components and op-amp structures. In order to make a true comparison with the results of (Jiang et al, 2007), same filter topology is used in this work.

The 4th order VCVS low-pass Butterworth filter can be realized by cascading two second order blocks, its transfer function is given in (3.1) where ω_{c1}, ω_{c2} and Q_1, Q_2 are the cutoff frequencies and the quality factors of two second order filters, respectively.

$$H(s) = \frac{{\omega_{c1}}^2}{s^2 + \frac{\omega_{c1}}{Q_1} s + {\omega_{c1}}^2} x \frac{{\omega_{c2}}^2}{s^2 + \frac{\omega_{c2}}{Q_2} s + {\omega_{c2}}^2} \tag{3.1}$$

The circuit of the 4th order VCVS low-pass Butterworth filter is shown in Fig. 3.1. According to the circuit in Fig. 3.1, the transfer function of this filter can be obtained as given in (3.2).

$$H(s) = \frac{1}{s^2 R_1 R_2 C_1 C_2 + s(R_1 C_1 + R_2 C_1) + 1} x \frac{1}{s^2 R_3 R_4 C_3 C_4 + s(R_3 C_3 + R_4 C_3) + 1} \tag{3.2}$$

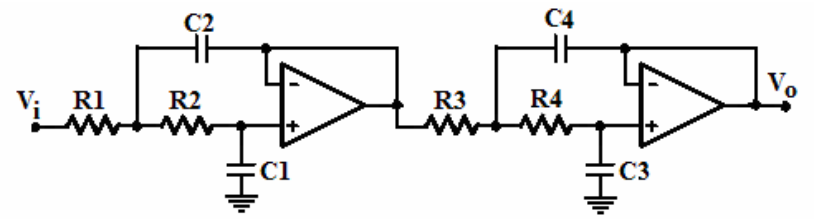

Fig. 3.1 Butterworth 4th order VCVS low-pass filter (Jiang et al., 2007)

According to (3.1) and (3.2), definitions of cutoff frequency and quality factor with respect to the circuit components are given in (3.3) and (3.4).

$$\omega_{c1} = \frac{1}{\sqrt{R_1 R_2 C_1 C_2}}, \omega_{c2} = \frac{1}{\sqrt{R_3 R_4 C_3 C_4}} \tag{3.3}$$

$$Q_1 = \frac{\sqrt{R_1 R_2 C_1 C_2}}{R_1 C_1 + R_2 C_1}, Q_2 = \frac{\sqrt{R_3 R_4 C_3 C_4}}{R_3 C_3 + R_4 C_3} \tag{3.4}$$

The 4th order Butterworth filter is implemented by cascading two second order ones, in this work and as in (Jiang et al, 2007), their cut-off frequency ω_{c1} and ω_{c2} are the same as 10k rad/s, their quality factors Q_1 and Q_2 are 1/0.7654 and 1/1.8478 respectively where quality factor values are determined from the table of low pass second order factors.

3.1.2 State Variable Filter

A state variable filter (SVF) realizes the state-space model directly. The instantaneous output voltage of one of the integrators corresponds to one of the state-space model's state variables. SVF can produce simultaneous low-pass, high-pass and band-pass outputs from a single input. A second order SVF is illustrated in Fig. 3.2 and is well described in (Schaumann and Valkenburg, 2001). The low-pass output is assumed here to be the desired output.

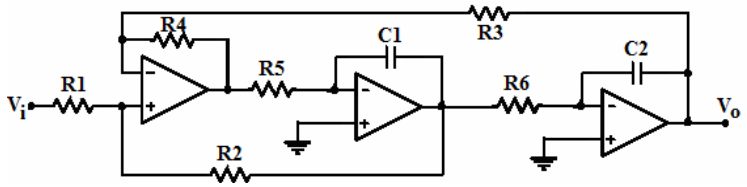

Fig. 3.2 State variable 2nd order low-pass filter (Schaumann and Valkenburg, 2001)

The response of a 2nd order low-pass circuit is specified by the passband gain (H_{SVF}), the cut-off frequency ($\omega_{SVF}=2\pi f_{SVF}$) and the selectivity factor (Q_{SVF}). These quantities are given in terms of passive component values in the following equations.

$$H_{SVF} = \frac{R_2(R_3 + R_4)}{R_3(R_1 + R_2)}, \omega_{SVF} = \sqrt{\left(\frac{R_4}{R_3}\right)\left(\frac{1}{C_1 C_2 R_5 R_6}\right)} \tag{3.5}$$

$$Q_{SVF} = \frac{R_3(R_1 + R_2)}{R_1(R_3 + R_4)}\sqrt{\frac{C_1 R_4 R_5}{C_2 R_3 R_6}} \tag{3.6}$$

The specification chosen here is ω_{SVF}=10k rad/s (f_{SVF}=10000/(2*π)=1591.55 Hz.) and Q_{SVF}=0.707 for reduced peak on low-pass response. The passband gain (H) is not very critical in most applications since it can be compensated by other cascaded analogue circuits. In the conventional design procedure (Schaumann and Valkenburg, 2001), H is fixed at some value; however for EA methods, it is unconstrained (Jiang et al, 2007).

3.2 Conventional Design Method

Conventional methods make all resistors equal to a normalized value of unity (1Ω), and then set the cut-off frequency of two second order Butterworth filters ω_{c1}, ω_{c2} to 10k rad/s, and quality factor of two second order Butterworth filters Q_1, Q_2 to 1/0.7654 and 1/1.8478 respectively. Similarly, all resistors are equalized to a normalized value and set cut-off frequency of second order state variable filter ω_{SVF} to 10k rad/s and quality factor Q_{SVF} to 0.707. Thus, the values of four capacitors can be obtained according to (3.3) and (3.4) for VCVS Butterworth filter and two capacitors according to (3.5) and (3.6) for state variable filter. As the values of components may not in the feasible range, for an exact design, a sensible way is firstly to multiply all values of resistors by a reasonable factor to make their values in the middle of the range. At the same time, all the capacitor values must be divided by the same factor (Schaumann and Valkenburg, 2001). Considering a design procedure where manufactured components are chosen, the exact component values are rounded to the nearest preferred values which will increase the total design error.

3.3 Evolutionary Algorithm Based Active Filter Design

In order to investigate the usage of evolutionary algorithms (EA) in active filter circuit design and to compare with previously used methods, two different low-pass analog active filter circuits given in Fig. 3.1 and Fig. 3.2 is selected (Vural et al., Article in Press). By establishing design criteria and design parameters to EA and satisfying desired constraints, the optimal circuit structure was aimed to be designed by the algorithm. Design problem has been introduced by composing an equation consists of design parameters as a cost function (CF). In the beginning of the algorithm a certain range was determined for design parameters by human designer. EA should minimize the given CF and obtain design criteria and design parameter values for the given range which gives minimum CF value.

In this work, GA, PSO and ABC algorithms are utilized for evolutionary algorithm based active filter design and performances of those are evaluated by means of computation time and accuracy. The aim is to estimate the preferred values of resistors and capacitors of the selected circuit with minimum design error. Each component used in filter design tasks was chosen to take value in the uttermost range of 10^3 to 10^6 ohms for the resistors and 10^{-9} to 10^{-6} farads for capacitors. Values outside these ranges were judged to lead to unwanted practical effects such as stray capacitance effects or large signal currents (Horrocks, 1993; Lacanette, 2010).

For both filter design tasks, the performance of evolutionary algorithms is investigated by varying own parameters in order to obtain the minimum total error value. Considering GA method, it was initiated with 15, 20 and 30 chromosomes with each is comprised of 8 genes in population. ‘Parent’ genes were selected with a roulette wheel selection. ‘Child’ genes were generated using random single-point cross-over applied with a probability of 0.5, 0.63 and 0.8 and mutation was applied to each bit in the gene with a probability of 0.01, 0.07 and 0.15. Uppermost number of iterations was determined as 10,000. PSO method was applied to filter design tasks such that initial population matrix size was *Nx16* where row number of *N* (*N*=5,10,15 respectively) indicates the number of particles in the population and column number of 16 is the dimension of particle vector which is given in (3.7). ABC optimization utilizes with a food number of 5, 50 and 500 with equal numbers of employer and onlooker bees. Maximum search limit is defined as 10,100 and 1000 cycles and maximum iteration number is 10000 for both filter design tasks where each of 8 components is represented with two parameters as similar to PSO method.

$$\begin{bmatrix} a_1 & al_1 & b_1 & bl_1 & c_1 & cl_1 & d_1 & dl_1 & e_1 & el_1 & f_1 & fl_1 & g_1 & gl_1 & h_1 & hl_1 \\ \vdots & & & & & & & & & & & & & & & \\ a_{10} & al_{10} & b_{10} & bl_{10} & c_{10} & cl_{10} & d_{10} & dl_{10} & e_{10} & el_{10} & f_{10} & fl_{10} & g_{10} & gl_{10} & h_{10} & hl_{10} \end{bmatrix}_{10x16} \tag{3.7}$$

Previous paragraph explains the common characteristics of each method used in both filter design tasks. However since circuit topologies differ in the following filter design tasks EA methods used in this work are utilized for each design task. In the following, design error definition for each filter structure and related component representation method for each EA technique is presented.

3.3.1 Butterworth Filter Design

In order to make a true comparison with previous methods, same error criterion with (Jiang et al, 2007) is used here. Total design error (3.8) is the summation of cost function error of cutoff frequency (CF_ω) and quality factor (CF_Q). Those definitions are given in (3.9).

$$Error_{total} = 0.5\ CF_\omega + 0.5\ CF_Q \tag{3.8}$$

$$CF_{\omega}=\frac{\left|\omega_{c1}-\omega_{c}\right|+\left|\omega_{c2}-\omega_{c}\right|}{\omega_{c}},CF_{Q}=\left|Q_{1}-\frac{1}{0.7654}\right|+\left|Q_{2}-\frac{1}{1.8478}\right| \tag{3.9}$$

In order to introduce the Butterworth filter design task to GA, PSO and ABC, a CF which includes values of discrete components ($R_{1...4}$, $C_{1...4}$) as design parameters is constituted as given in (3.10). The right side of (3.10) would constitute the CF which EA techniques would minimize. Here ω_c is set as 10k rad/s. GA, PSO and ABC should obtain the minimum value of CF, and the preferred values of design parameters that minimize CF. It is desired to obtain the exact values of design parameters ($R_{1...4}$, $C_{1...4}$) which equate CF to a very close value to zero. In each decade, any of twelve preferred values can be taken according to standard E12 series within the range of 10^3 to 10^6 ohms for resistors and 10^{-9} to 10^{-6} farads for capacitors.

$$Error_{total}=\left(\begin{array}{l}0.5\dfrac{\left|\dfrac{1}{\sqrt{R_1R_2C_1C_2}}-\omega_c\right|+\left|\dfrac{1}{\sqrt{R_3R_4C_3C_4}}-\omega_c\right|}{\omega_c}\\ +0.5\left(\left|\dfrac{\sqrt{R_1R_2C_1C_2}}{R_1C_1+R_2C_1}-\dfrac{1}{0.7654}\right|+\left|\dfrac{\sqrt{R_3R_4C_3C_4}}{R_3C_3+R_4C_3}-\dfrac{1}{1.8478}\right|\right)\end{array}\right) \tag{3.10}$$

3.3.1.1 Component Representation for GA

The values of resistors and capacitors constitute the dimension of the chromosome where each chromosome is comprised of 8 genes as given in Fig. 3.3. Each gene is binary coded 4 bits, representing the resistors and capacitors compatible with E12 series. As a result, considering VCVS Butterworth active filter design task, GA utilizes chromosomes with 32 bits.

R1	R2	R3	R4	C1	C2	C3	C4

Fig. 3.3 Component values in a chromosome for Butterworth filter

3.3.1.2 Component Representation for PSO and ABC

The values of capacitors and resistors constitute the dimension of particle vector. Since the probable values vary from three decade range, a coding scheme is used as in (3.11).

$$\begin{array}{ll}R_1=a\times100\times10^{a1}(\Omega), & R_2=b\times100\times10^{b1}(\Omega)\\ R_3=c\times100\times10^{c1}(\Omega), & R_4=d\times100\times10^{d1}(\Omega)\\ C_1=e\times100\times10^{e1}(pF), & C_2=f\times100\times10^{f1}(pF)\\ C_3=g\times100\times10^{g1}(pF), & C_4=h\times100\times10^{h1}(pF)\end{array} \tag{3.11}$$

Since each resistor should take an E12 serial value in the range of 10^3 to 10^6 ohms, the design constraint for resistors given in (3.12) must be satisfied. Similarly each capacitor should take E12 serial value in the range of 10^{-9} to 10^{-6} farads. If capacitor values are defined in picofarads (pF) then design constraint for capacitors given in (3.13) would be valid.

$$0.1 \le a, b, c, d \le 0.82 \quad 2 \le a1, b1, c1, d1 \le 4 \tag{3.12}$$

$$0.1 \le e, f, g, h \le 0.82 \quad 2 \le e1, f1, g1, h1 \le 4 \tag{3.13}$$

3.3.2 State Variable Filter Design

Design error (3.14) is the summation of cost function error of cutoff frequency (CF_ω) and quality factor (CF_Q). Those definitions are given in (3.15). Since pass band gain is unconstrained, it is not included to design error definition.

$$Error_{total} = 0.5\ CF_\omega + 0.5\ CF_Q \tag{3.14}$$

$$CF\omega = \frac{|\omega_{SVF} - \omega_0|}{\omega_0}, CF_Q = \frac{|Q_{SVF} - Q|}{Q} \tag{3.15}$$

In order to introduce the SVF design task to GA, PSO and ABC, a CF which includes values of discrete components ($R_{1...6}$, $C_{1,2}$) as design parameters is constituted as given in (3.16). The right side of (3.16) constituted the CF which PSO would minimize. Here ω_0 and Q are set to 10k rad/s and 0.707, respectively. GA, PSO and ABC should obtain the minimum value of the CF, and the preferred values of design parameters that minimize CF. It is desired to obtain the exact values of design parameters ($R_{1...6}$, $C_{1,2}$) which equate CF to a very close value to zero. In each decade, any of twenty four and ninety six preferred values can be taken according to standard E24 series and E96 series, respectively within the range of 10^3 to 10^6 ohms for resistors and 10^{-9} to 10^{-6} farads for capacitors.

$$Error_{total} = 0.5\frac{\left|\sqrt{\left(\frac{R_4}{R_3}\right)\left(\frac{1}{C_1C_2R_5R_6}\right)} - \omega_0\right|}{\omega_0} + 0.5\frac{\left|\frac{R_3(R_1+R_2)}{R_1(R_3+R_4)}\sqrt{\frac{C_1R_4R_5}{C_2R_3R_6}} - Q\right|}{Q} \tag{3.16}$$

3.3.2.1 Component Representation for GA

The values of resistors and capacitors constitute the dimension of the chromosome. Each chromosome is comprised of 8 genes as given in Fig. 3.4. Each gene is binary coded 5 bits,

representing the resistor and capacitor values for E24 series and 7 bits representing the resistor and capacitor values for E96 series. As a result, for state variable active filter design task, GA utilizes chromosomes with 40 bits and 56 bits for state variable active filter design with components selected from E24 and E96 series, respectively.

R1	R2	R3	R4	R5	R6	C1	C2

Fig.3.4 Component values in a chromosome for SVF

3.3.2.2 Component Representation for PSO and ABC

In each decade, any of twenty four and ninety six preferred values can be taken according to standard E24 series and E96 series, respectively. Since the probable values vary from three decade range, a coding scheme is used as in (3.17).

$$\begin{aligned} R_1 &= a \times 100 \times 10^{a1}\ (\Omega), & R_2 &= b \times 100 \times 10^{b1}\ (\Omega) \\ R_3 &= c \times 100 \times 10^{c1}\ (\Omega), & R_4 &= d \times 100 \times 10^{d1}\ (\Omega) \\ R_5 &= e \times 100 \times 10^{e1}\ (pF), & R_6 &= f \times 100 \times 10^{f1}\ (pF) \\ C_1 &= g \times 100 \times 10^{g1}\ (pF), & C_2 &= h \times 100 \times 10^{h1}\ (pF) \end{aligned} \quad (3.17)$$

First, SVF is designed with components that are compatible with E24 series. Since each resistor should take an E24 serial value in the range of 10^3 to 10^6 ohms, the design constraint for resistors given in (3.18) must be satisfied. Similarly each capacitor should take E24 serial value in the range of 10^{-9} to 10^{-6} farads. If capacitor values are defined in picofarads (pF) then design constraint for capacitors given in (3.19) would be valid.

$$0.1 \le a, b, c, d, e, f \le 0.91 \qquad 2 \le a1, b1, c1, d1, e1, f1 \le 4 \quad (3.18)$$

$$0.1 \le g, h \le 0.91 \qquad 2 \le g1, h1 \le 4 \quad (3.19)$$

Following, PSO and ABC algorithms estimated the component values of SVF circuit that are compatible with E96 series. Design constraints are specified similarly as explained for E24 series. The difference is the upper limit constraints for resistors and capacitors as given in (3.20) and (3.21).

$$0.1 \le a, b, c, d, e, f \le 0.976 \qquad 2 \le a1, b1, c1, d1, e1, f1 \le 4 \quad (3.20)$$

$$0.1 \le g, h \le 0.976 \qquad 2 \le g1, h1 \le 4 \quad (3.21)$$

3.4 Simulation Results

The performances of EAs on different types of filter design are evaluated by means of accuracy and computation time. Simulation results of GA, PSO and ABC based filter design tasks are investigated in the following. In order to evaluate the feasibility of component values obtained by EA based design method, each filter topology is redesigned in SPICE simulation environment with corresponding component values and an appropriate op-amp macromodel.

3.4.1 Butterworth Filter Design Results

In Butterworth filter design with components selected from E12 series, the target CF result with EA techniques is aimed to be smaller than 0.018. Considering GA method, this requirement has been met at the 7768^{th} iteration and the exact value of CF is obtained as 0.0166 with mutation probability of 0.01, cross-over probability of 0.63 and chromosome number of 20. Effects of GA's own parameters over total error values (CF values) are given in Table 3.1. Computation time for GA is 4.1 minutes due to the search of exact component values compatible with E12 series.

Different from GA, PSO utilizes a search space within the constraints as given in (3.12) and (3.13). This method shortens the computation time required (3.7 s) and obtained component values are in the provided ranges; however these values do not fit to the E12 values. Therefore PSO based results of discrete component values are rounded to the nearest preferred E12 serial value. Rounded values are evaluated whether they meet the target CF result. If not, PSO algorithm is rerun until satisfying CF results have been obtained when ideal values found by PSO is rounded to the nearest preferred E12 series. The duration of total process is 3.2 minutes for Butterworth filter design with a total design error of 0.0076 utilizing acceleration factors of 1.7 and particle number of 10. Effects of PSO algorithm's own parameters over total error values (CF values) are given in Table 3.2. Details of PSO based Butterworth filter design are explained in (Vural and Yildirim, 2010a).

ABC optimization method obtained a design error of 0.0113 at 525^{th} iteration in 0.7 seconds with a search limit of 100 and total bee population of 1000. Effects of ABC algorithm's own parameters over total error values (CF values) are given in Table 3.3.

Table 3.1 Effects of Genetic Algorithm's Own Parameters on Butterworth Filter Performance

CF Values (GA-E12) $x10^{-3}$		**Mutation Probability (MP)**								
		MP=0.01			MP=0.07			MP=0.15		
		Crossover Probability (CP)			**Crossover Probability (CP)**			**Crossover Probability (CP)**		
		CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8
# of Chromosemes (*N*)	N=15	19	37.3	44	18	41	17	53	24	46
	N=20	37	**16.6**	25	17	18	29	72	21	83
	N=30	19	64	23	63	79	67	56	65	63

Table 3.2 Effects of PSO Algorithm's Own Parameters on Butterworth Filter Performance

CF Values (PSO-E12)		**Acceleration Factors (c_1, c_2)**		
		$c_1=c_2=1.5$	$c_1=c_2=1.7$	$c_1=c_2=2$
# of Particles (*N*)	N=5	0.014	0.0097	0.0796
	N=10	0.011	**0.0076**	0.0895
	N=15	0.014	0.0091	0.0139

Table 3.3 Effects of ABC Algorithm's Own Parameters on Butterworth Filter Performance

CF Values (ABC-E12)		**Search Limit (SL)**		
		SL=10	SL=100	SL=1000
# of Bees (*NP*)	NP=10	0.2611	0.0467	0.1569
	NP=100	0.0951	0.0193	0.0268
	NP=1000	0.0225	**0.0113**	0.0251

Following, performance of evolutionary algorithms is explored over 20 runs with optimal own parameters as provided in Table 3.1, Table 3.2 and Table 3.3. The resulting CF values obtained in each run were used to produce box and whisker plots to show the median performance as well as outliers as given in Fig. 3.5. Upper and lower ends of boxes represent 75^{th} and 25^{th} percentiles. Median is depicted by the red line. The whiskers are lines extending from each end of the boxes to show the extent of the rest of the data. Outliers are data with values beyond the ends of the whiskers. CF values versus iteration number for 5 independent runs are plotted in Fig. 3.6, Fig. 3.7 and Fig. 3.8 for GA, ABC and PSO algorithms, respectively. These figures depict that number of iterations required to achieve the quality requirements are slightly different in each run.

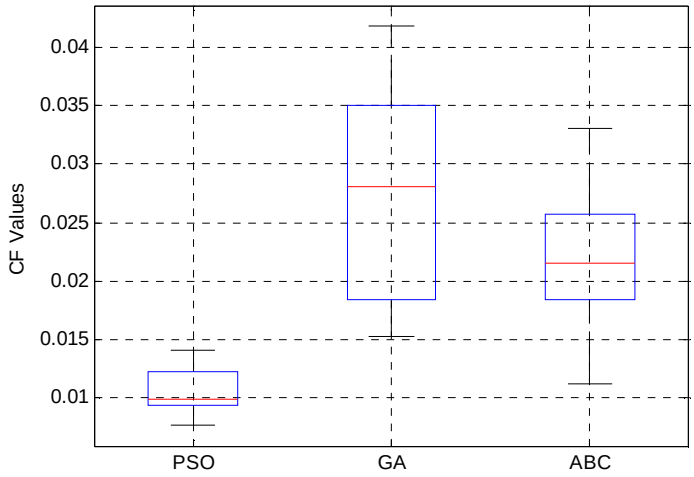

Fig. 3.5 Box and whisker plots for Butterworth filter design with E12 series over 20 runs

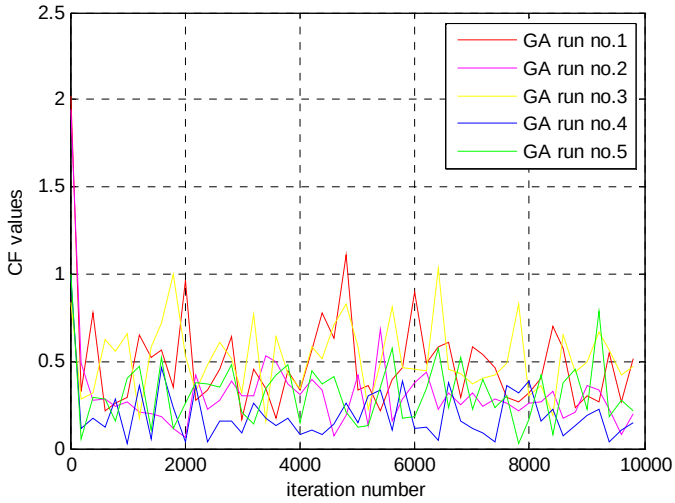

Fig. 3.6 CF values vs. iteration number for GA method (E12 series)

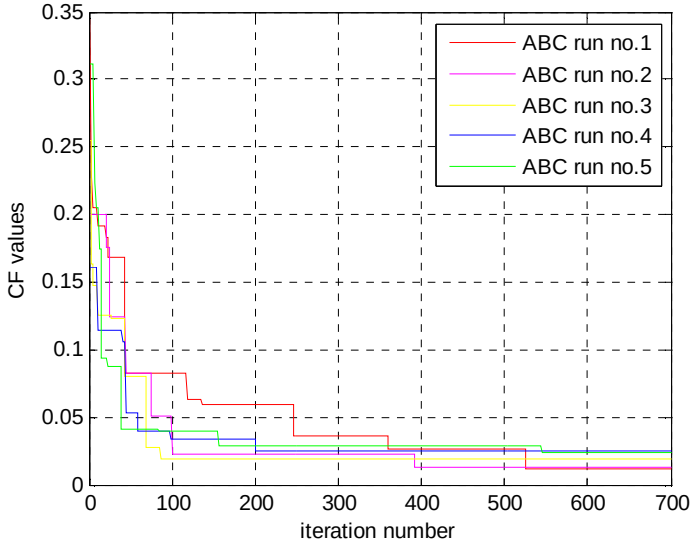

Fig. 3.7 CF values vs. iteration number for ABC method (E12 series)

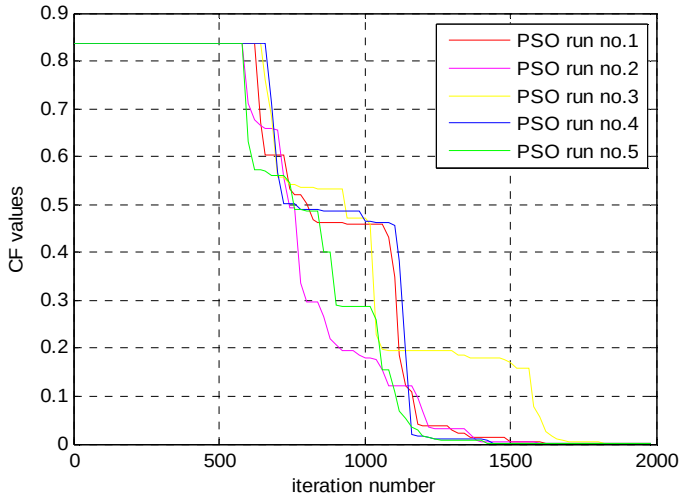

Fig. 3.8 CF values vs. iteration number for PSO method (E12 series)

Exact values of discrete components, deviations and total error of GA, PSO and ABC based design and previously used methods are tabulated in Table 3.4 where the results of (Jiang et al,2007) are tabulated from second to sixth column and results of (Vural and Yildirim,2010a) are given at the last column. Compared to the previous methods and other techniques, PSO algorithm achieved a smaller design error in a shorter computation time than GA even if the ideal results are rounded to the nearest preferred values. However in terms of execution time, ABC outperforms both GA and PSO.

Table 3.4 Previous Methods and GA, ABC, PSO Techniques for Butterworth Filter Design

	Conventional		**(Jiang et al., 2007)**			**(Vural et al., Article in Press)**		
	Ideal Values	**Nearest Preferred**	**TS**	**GA**	**CSA**	**GA**	**ABC**	**PSO**
R1 (kΩ)	1	1	27	4.7	4.7	6.8	4.7	4.58
R2 (kΩ)	1	1	0.27	1.8	4.7	6.8	4.7	4.7
C1 (nF)	38.27	39	2.7	12	8.2	5.6	8.2	8.2
C2 (nF)	26.13	270	470	100	56	39	56	56
R3 (kΩ)	1	1	220	100	0.27	39	1	1.1
R4 (kΩ)	1	1	0.82	4.7	27	1	39	1
C3 (nF)	92.39	100	82	1.8	6.8	4.7	4.7	87.6
C4 (nF)	261.3	100	0.68	12	200	56	56	102.2
Δω	0	0.02549	0.01291	0.01503	0.01141	0.0179	0.0201	0.0135
ΔQ	0	0.05026	0.04264	0.02130	0.00436	0.0153	0.0024	0.0018
Total Error	0	0.03788	0.02778	0.01817	0.00789	0.0166	0.0113	**0.0076**

The frequency responses of the filter achieved by the EA techniques are shown in Fig.3.9. Butterworth filter is realized with E12 compatible results of GA, ABC and PSO methods and LM741 op-amp macro model with SPICE simulator. SPICE simulation proves that all proposed methods provide a maximally flat response in the pass band. In this figure, x-axis represents the amplitude response in decibels and y-axis is the frequency. Since ω_c is the frequency point which the output of the circuit is -3 dB of the nominal passband value, those points are also marked on Fig.3.9. Yet, it should be noted that cut-off frequency obtained at the output, should be lower than each designed cascade stages. Due to the non-idealities of the LM-741 op-amp macromodel, this decrement was not observed in Fig.3.9.

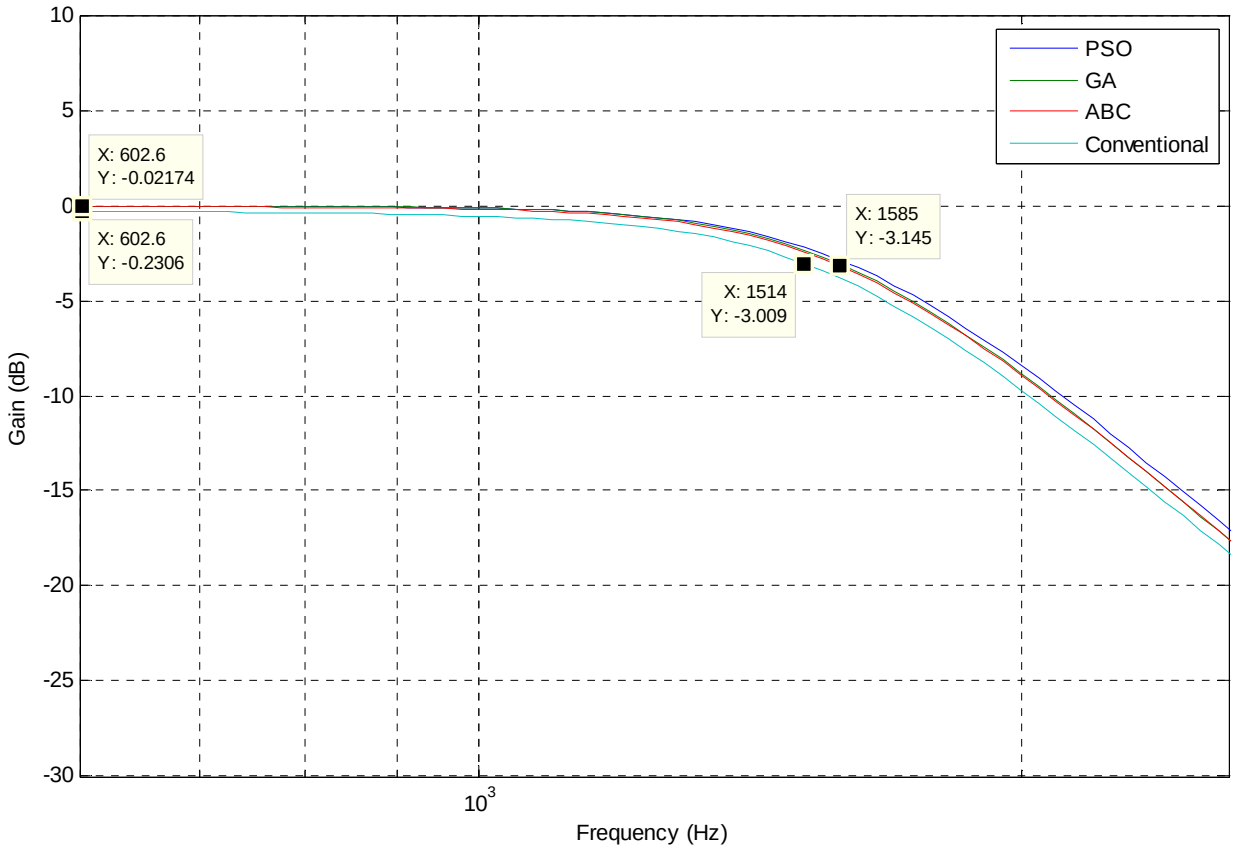

Fig. 3.9 Frequency responses of conventional and EA based design of Butterworth filter (E12)

3.4.2 State Variable Filter Design Results

In state variable filter design with components selected from E24 and E96 series, the target CF result with EA techniques is aimed to be smaller than $1x10^{-3}$. Considering GA method for design with E24 series, this requirement has been met at the 1541^{th} iteration and the exact value of CF is obtained as $2.1735x10^{-4}$ with mutation probability of 0.01, cross-over

probability of 0.63 and chromosome number of 20. Effects of GA's own parameters over total error values (CF values) are given in Table 3.5. Computation time for GA is 5.2 minutes due to the search of exact component values compatible with E24 series. GA based design with E96 series possessed a CF value of 1.0449×10^{-4} at 4441^{th} iteration with mutation probability of 0.01, cross-over probability of 0.63 and chromosome number of 20. Effects of GA's own parameters over total error values (CF values) are given in Table 3.6. The search of exact component values compatible with E96 series requires 7.4 minutes.

Table 3.5 Effects of Genetic Algorithm's Own Parameters on SVF Performance (E24)

CF Values (GA-E24) $\times 10^{-4}$		**Mutation Probability (MP)**								
		MP=0.01			MP=0.07			MP=0.15		
		Crossover Probability (CP)			**Crossover Probability (CP)**			**Crossover Probability (CP)**		
		CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8
# of Chromosemes (*N*)	N=15	26	14	12	18	11	14	9.5	4.94	6.57
	N=20	3.4	**2.17**	8.26	4.1	6.98	3.8	2.6	9.95	4.93
	N=30	8.7	3.26	5.52	5.8	3.66	7.1	11	7.09	9.91

Table 3.6 Effects of Genetic Algorithm's Own Parameters on SVF Performance (E96)

CF Values (GA-E96) $\times 10^{-4}$		**Mutation Probability (MP)**								
		MP=0.01			MP=0.07			MP=0.15		
		Crossover Probability (CP)			**Crossover Probability (CP)**			**Crossover Probability (CP)**		
		CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8	CP=0.5	CP=0.63	CP=0.8
# of Chromosemes (*N*)	N=15	3.3	3.34	5.4	5.1	1.94	2.71	2	8.1	6.94
	N=20	8.1	**1.05**	4.6	2.1	1.43	7.2	8	2.5	3.89
	N=30	3.2	3.41	4.3	3.8	2.5	6.56	2	3.16	5.32

PSO and ABC design method was explained in Butterworth Filter Design Results section and utilized for E24 and E96 series. The duration for PSO based design with E24 series is 4.5 minutes with a total design error of 3.6603×10^{-4} utilizing acceleration factors of 1.7 and particle number of 10. Effects of PSO algorithm's own parameters over total error values (CF values) are given in Table 3.7. Considering E96 series, PSO possessed a CF result of 3.1084×10^{-4} in 5.6 minutes with acceleration factors of 1.7 and particle number of 10. Effects of PSO algorithm's own parameters over total error values (CF values) are given in Table 3.8.

Table 3.7 Effects of PSO Algorithm's Own Parameters on SVF Performance (E24)

CF Values (PSO-E24)		**Acceleration Factors (c_1, c_2)**		
		$c_1=c_2=1.5$	$c_1=c_2=1.7$	$c_1=c_2=2$
# of Particles (*N*)	N=5	$6.99x10^{-4}$	$4.76x10^{-4}$	$2.05x10^{-3}$
	N=10	$6.54x10^{-4}$	**$3.66x10^{-4}$**	$3.53x10^{-3}$
	N=15	$5.26x10^{-4}$	$3.72x10^{-4}$	$1.05x10^{-2}$

Table 3.8 Effects of PSO Algorithm's Own Parameters on SVF Performance (E96)

CF Values (PSO-E96)		**Acceleration Factors (c_1, c_2)**		
		$c_1=c_2=1.5$	$c_1=c_2=1.7$	$c_1=c_2=2$
# of Particles (*N*)	N=5	$6.01x10^{-4}$	$5.71x10^{-4}$	$2.36x10^{-2}$
	N=10	$8.01x10^{-4}$	**$3.11x10^{-4}$**	$4.36x10^{-2}$
	N=15	$9.89x10^{-4}$	$4.52x10^{-4}$	$7.34x10^{-2}$

ABC optimization of SVF design task results in 3.4 seconds after 110 iterations with a total error of $3.81x10^{-4}$ utilizing search limit of 100 and bee population of 1000 when components are selected from E24 series. Effects of ABC algorithm's own parameters over total error values (CF values) are tabulated in Table 3.9. However, when considering E96 series, ABC is the most successful algorithm among the others with the shortest computation time and minimum total design error. ABC based SVF design is concluded in 2.6 seconds after 175 iterations with a total error of $0.171x10^{-4}$ utilizing search limit of 100 and bee population of 1000. Effects of ABC algorithm's own parameters over CF values are given in Table 3.10.

Table 3.9 Effects of ABC Algorithm's Own Parameters on SVF Performance (E24)

CF Values (ABC-E24)		**Search Limit (SL)**		
		SL=10	SL=100	SL=1000
# of Bees (*NP*)	NP=10	0.00944	0.00621	0.01572
	NP=100	0.00070	0.00068	0.00363
	NP=1000	0.00048	**0.00038**	0.00056

Table 3.10 Effects of ABC Algorithm's Own Parameters on SVF Performance (E96)

CF Values (ABC-E96)		**Search Limit (SL)**		
		SL=10	SL=100	SL=1000
# of Bees (*NP*)	NP=10	0.08794	0.001124	0.00219
	NP=100	0.00057	0.000176	0.00049
	NP=1000	0.0006	0.000017	0.00088

Following, the performance of EAs over 20 runs is explored with optimal own parameters as provided in Table 3.5-Table 3.10. The resulting CF values obtained in each run were used to produce box and whisker plots for E24 series and E96 series in Fig.3.10 and Fig.3.11 respectively. Detailed explanation of this type of plots was given for Fig.3.5. Different from Fig.3.5, outliers are marked with red plus.

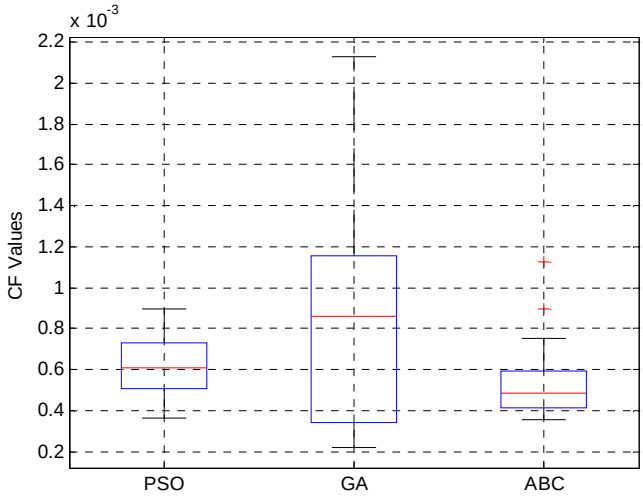

Fig. 3.10 Box and whisker plots for SVF design with E24 series over 20 runs

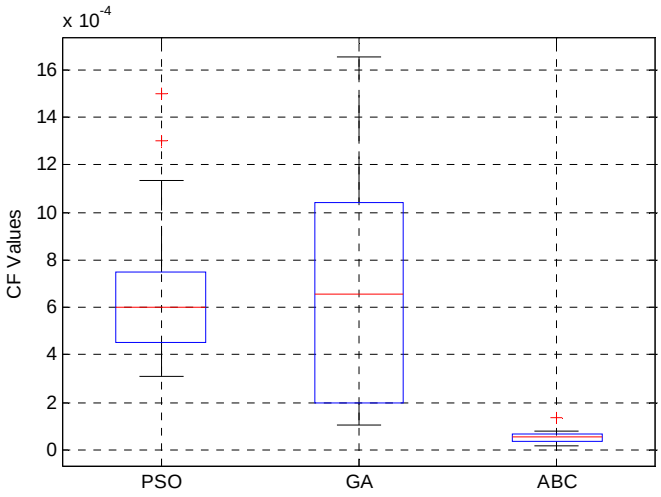

Fig. 3.11 Box and whisker plots for SVF design with E96 series over 20 runs

Considering SVF design, CF values versus iteration number for 5 independent runs are plotted in Fig. 3.12, Fig. 3.13, Fig. 3.14 for GA, ABC and PSO algorithms utilized for E24 series, and in Fig. 3.15, Fig. 3.16 and Fig. 3.17 for the same evolutionary algorithms utilized for E96 series respectively. In those figures, it can be seen that number of iterations required to achieve the quality requirements are slightly different in each run.

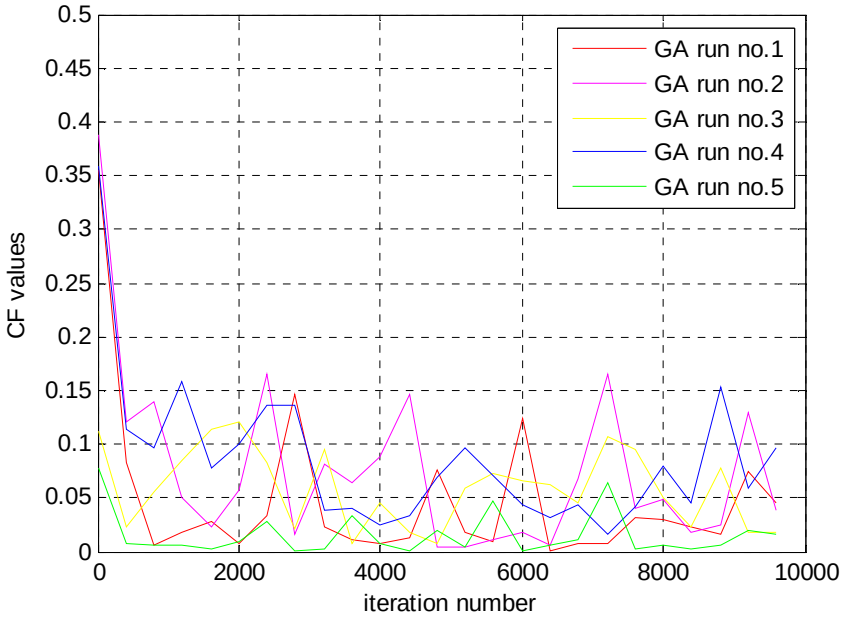

Fig. 3.12 CF values vs. iteration number for GA method (E24 series)

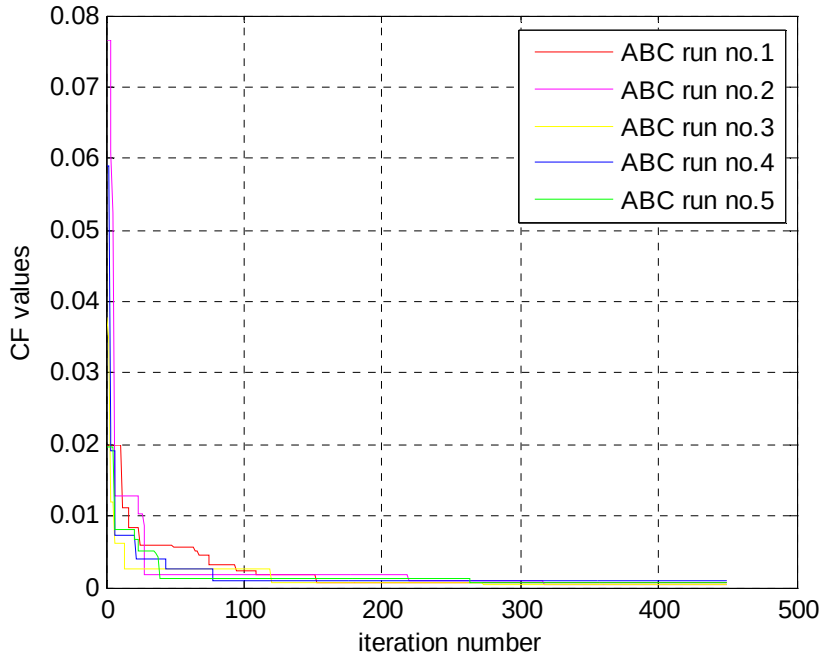

Fig. 3.13 CF values vs. iteration number for ABC method (E24 series)

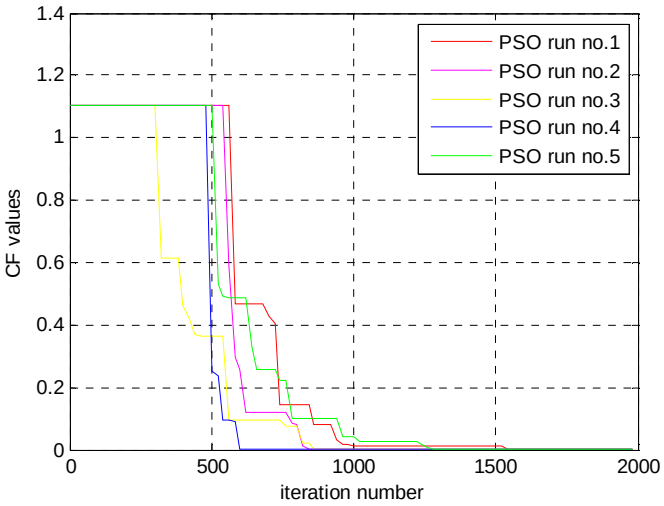

Fig. 3.14 CF values vs. iteration number for PSO method (E24 series)

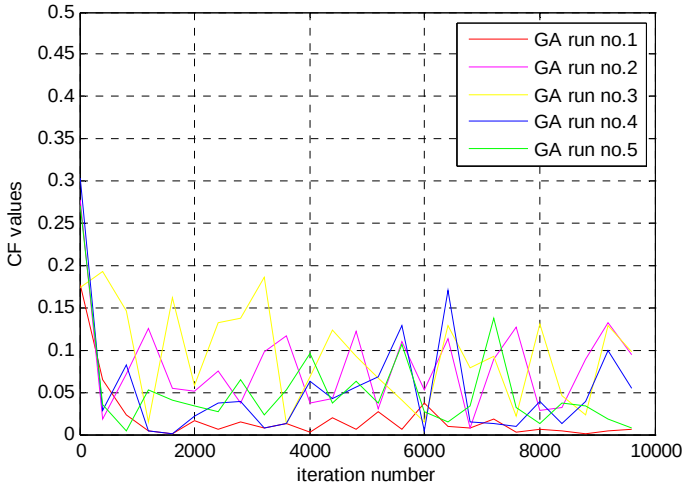

Fig. 3.15 CF values vs. iteration number for GA method (E96 series)

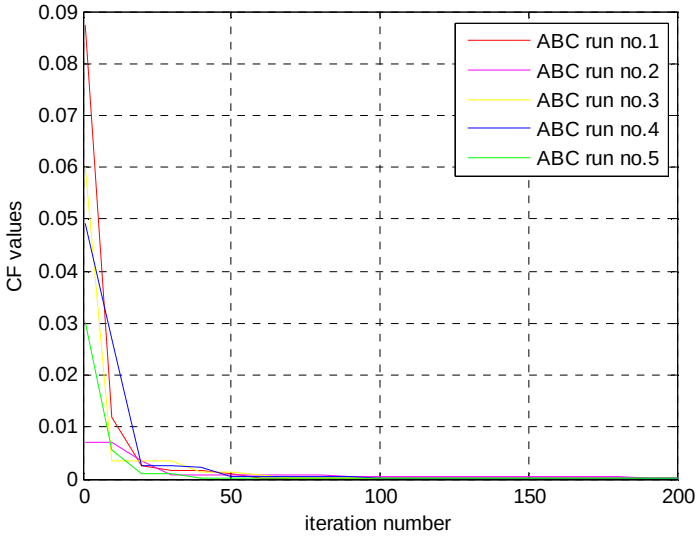

Fig. 3.16 CF values vs. iteration number for ABC method (E96 series)

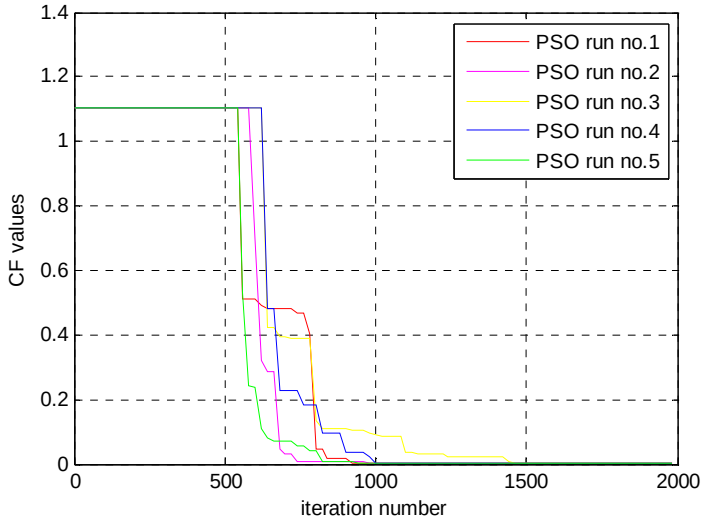

Fig. 3.17 CF values vs. iteration number for PSO method (E96 series)

Exact values of discrete components for both E24 and E96 series, deviations and total error of GA, PSO and ABC based design are tabulated in Table 3.11 where results of (Vural and Yildirim, 2010b) are provided in second, third, sixth and last column. Considering SVF design task with components selected from E24 series, GA results in minimum total error value. However, execution time of GA based design is the longest among others. ABC outperforms other EA techniques by means of accuracy and computation time when components are selected from E96 series.

Table 3.11 Previous Methods and GA, ABC, PSO Techniques for SVF Design (Vural et al., Article in Press)

	Conventional		**GA** (E24)	**ABC** (E24)	**PSO** (E24)	**GA** (E96)	**ABC** (E96)	**PSO** (E96)
	Ideal Values	**Nearest Preferred**						
R1 (Ω)	4000	4120	43000	62000	10000	69000	59000	10200
R2 (Ω)	1656	1690	5600	1000	1650	2550	88700	8660
R3 (Ω)	4000	4120	24000	91000	30110	65300	54900	14700
R4 (Ω)	4000	4120	280000	4300	212000	237000	90900	187000
R5 (Ω)	4000	4120	4400	27000	1039	2870	10000	1130
R6 (Ω)	4000	4120	9200	1800	3900	1430	51100	2940
C1 (nF)	25	25.5	180	2.7	470	110	7.5	464
C2 (nF)	25	25.5	16	3.6	37	80.4	4.32	82.5
$\Delta\omega$	0	0.049	3.61×10^{-4}	1.43×10^{-4}	4.08×10^{-4}	3.63×10^{-5}	0.29×10^{-4}	1.46×10^{-4}
ΔQ	0	0.003	0.73×10^{-4}	6.17×10^{-4}	3.24×10^{-4}	1.73×10^{-4}	0.05×10^{-4}	4.76×10^{-4}
Total Error	0	0.026	**2.17×10^{-4}**	3.80×10^{-4}	3.66×10^{-4}	1.05×10^{-4}	**0.17×10^{-4}**	3.11×10^{-4}

In order to demonstrate the accuracy of EA based SVF design methods, SVF is simulated with LM741 op-amp macro model and E24/E96 compatible results of EA methods using SPICE. The frequency responses of SVF are given in Fig. 3.18 and Fig.3.19 for E24 and E96 compatible results, respectively. Here, gain values are different since gain was unconstrained at the beginning of the design procedure. From SPICE simulation results, it is evident that evolutionary approaches provide a maximally flat response and less cut-off frequency (-3dB point) deviation than conventional method.

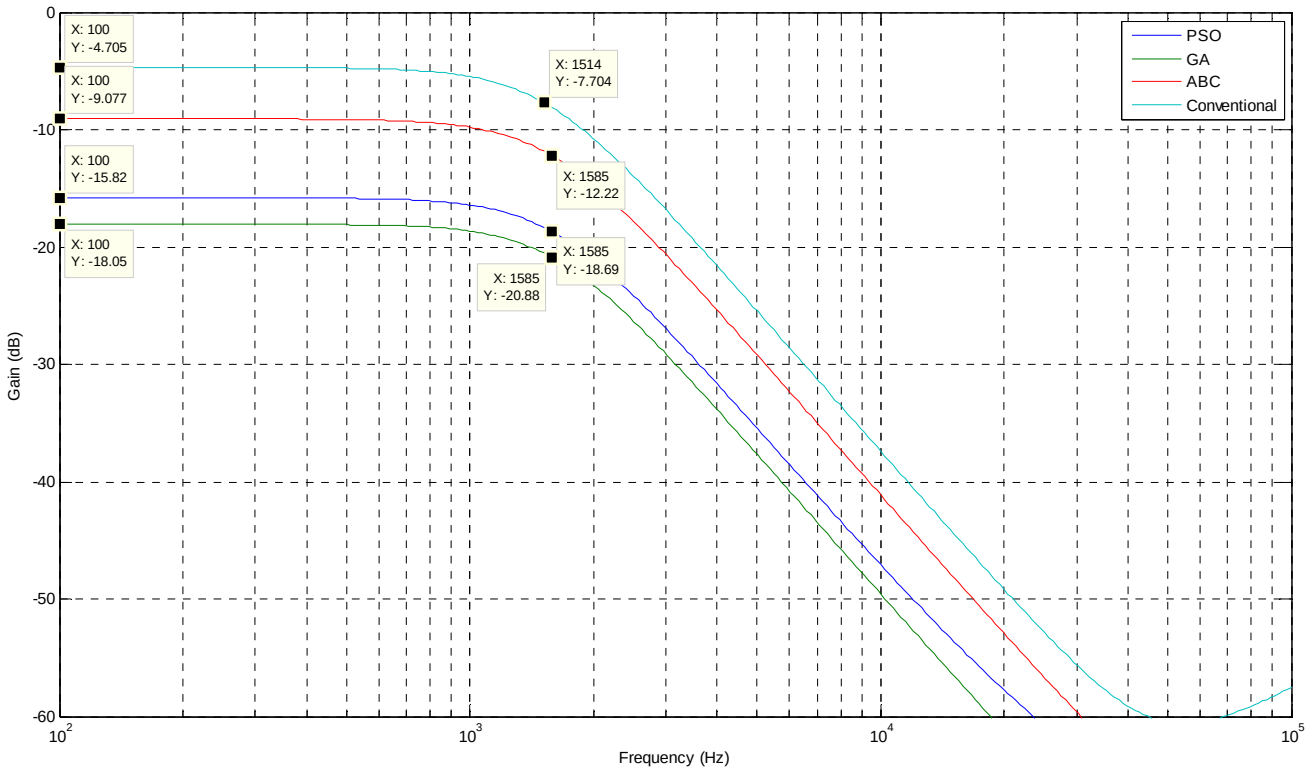

Fig. 3.18 Frequency responses of conventional and EA based design of SVF filter (E24)

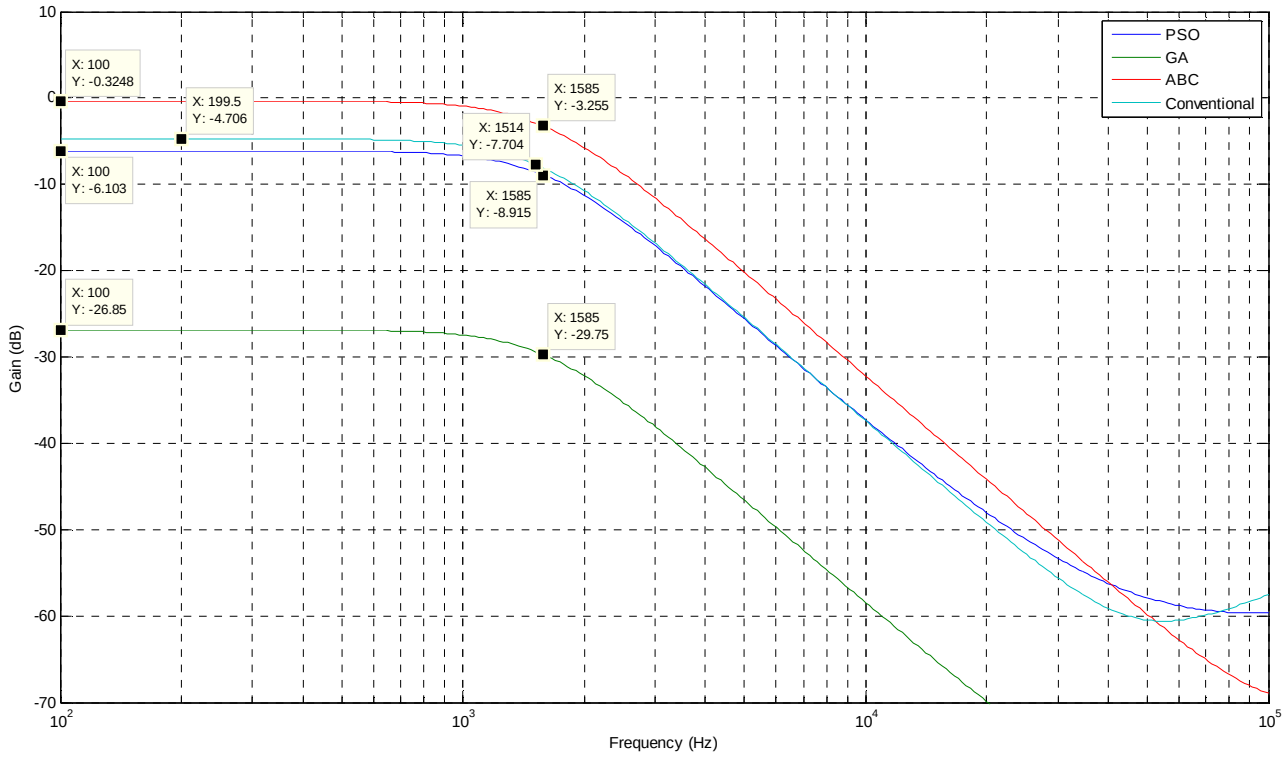

Fig. 3.19 Frequency responses of conventional and EA based design of SVF filter (E96)

3.5 Summary

The performances of evolutionary algorithms on analog active filter design have been investigated comprehensively. GA, ABC and PSO algorithms were utilized for both 4th order Butterworth low-pass analog filter and 2nd order SVF design and was investigated for the selection of passive components from different manufactured series by means of accuracy and execution time. Selection of the optimal own parameters is very crucial on minimizing total design error value thus effecting filter performance. Among the evolutionary approaches utilized in this work GA requires more fine-tuning of the own parameters and increasing number of chromosomes decreases total error values at the cost of execution time. Considering ABC, increasing search limit value facilitates obtaining better CF values when bee population is small. However, when bee population is crowded than selection of bigger search limits decelerates the algorithm and worsens the performance. Increasing particle number in PSO improves the performance unless the acceleration factors are selected as 2.

Following, 20 runs were performed with optimal own parameters obtained previously. The resulting CF values were used to produce box and whisker plots which shows that CF values

obtained with GA method varies the most among the other methods. The iteration number required to achieve the quality restrictions are slightly different in each run for each method which can be seen in the plots of CF values versus iteration number for 5 different runs.

Considering Butterworth filter design with components selected from E12 series for a true comparison with (Jiang et al, 2007); PSO achieved the smallest design error with respect to previous methods and other EA methods. Moreover, less design error is obtained with GA than the previously used one in (Jiang et al, 2007).

Components of SVF are selected from two different manufactured series in order to investigate whether performance of EA methods will increase or not when same topology is designed with different series. GA algorithm achieved the smallest design error but the longest execution time when selecting components from E24 series. This is mainly due to the fact that PSO and ABC has fewer primitive mathematical operators than in GA (e.g reproduction, mutation and crossover). Those mathematical operations require more fine-tuning of own parameters. However when tolerances of components became tighter in E96 series, ABC algorithm obtained the most successful results by means of both accuracy and execution time. Choosing optimal search limit value avoids local minimum trapping and increasing number of bee population improves the probability of converging to global minimum. Therefore, selecting optimal own parameters for ABC improves the accuracy as well as the convergence rate. Moreover, the performance of PSO was not affected significantly due to the usage of different manufactured series for same filter topology as other EA methods were. Therefore it has been proved that the selection of components from different series had an influence on the performance of EA methods. ABC outperforms other methods by means of execution time for all design cases, however when considering design accuracy, each method has achieved the smallest design error depending on the design case. From Spice simulation results, it is clearly observed that the conventional method does not provide a maximally flat response in the pass band and results in bigger cut-off frequency deviation. Moreover, SVF design with conventional method screens stop band ripples during frequency analysis unlike with EA techniques.

4. PARTICLE SWARM OPTIMIZATION FOR INTEGRATED CIRCUIT DESIGN

In this section, particle swarm optimization is utilized for both digital and analog integrated circuit design cases. In order to investigate PSO performance for digital circuit design, switching characterization of an inverter is considered. Switching performance is one of the quality metrics when considering a digital design. From a system designer's perspective, the performance of a digital circuit expresses its computational ability (Rabaey et al., 2003). While the amount of digital design activity far outpaces that of analog design, most digital systems require analog modules for interfacing to the external world. One of the key features of designing an analog circuit is the optimal sizing of transistors where the chosen schematic must be dimensioned to comply with the required specifications (Allen and Holberg, 2002). PSO performance for analog circuit design with constraints is evaluated for MOS transistor sizing of a differential amplifier with current mode load and a two stage operational amplifier.

4.1 Particle Swarm Optimization for Digital Integrated Circuit Design

The switching characteristics of digital integrated circuits and in particular, of inverter circuits, essentially determine the overall operating speed of digital systems. The transient performance requirements of a digital system are usually among the most important design specifications that must be met by the circuit designer. Therefore, the switching speed of the circuit must be estimated and optimized very early in the design phase (Kang and Leblebici, 2005; DeMassa and Ciccone, 1996).

4.1.1 Dynamic Characteristics of an Inverter

The inverter is truly the nucleus of all digital designs. A CMOS inverter consists of a PMOS and a NMOS transistor connected in cascade and output is terminated with a load capacitance as shown in Fig. 4.1.

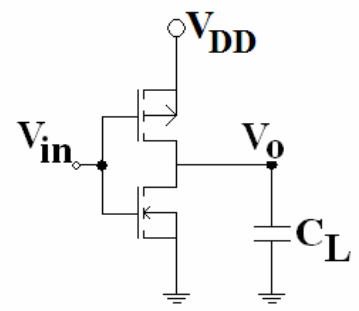

Fig. 4.1 Inverter Structure

Here V_{DD} stands for the supply voltage and input and output voltages are represented with V_{in} and V_o respectively. The dynamic behavior of the CMOS inverter is determined by the time it takes to charge and discharge the load capacitance through the PMOS and NMOS transistors respectively (Rabaey et al., 2003).

The closed-form delay expressions have been derived under the assumption of pulse excitations for lumped load capacitances. Before derivation of delay expressions, definitions of output voltage fall and rise times and propagation delay times are explained. The fall time, t_f, is defined as the time required for the output voltage to drop from $V_{90\%}$ level to $V_{10\%}$ level. Similarly the rise time, t_r, is the time required for the output voltage to rise from the $V_{10\%}$ level to $V_{90\%}$ level. The propagation delay times τ_{pHL} and τ_{pLH} determine the input-to-output signal delay during the high-to-low and low-to-high transitions of the output respectively. By definition τ_{pHL} is the time delay between the $V_{50\%}$ transition of the rising input voltage and the $V_{50\%}$ transition of the falling output voltage. Similarly, τ_{pLH} is defined as the time delay between the $V_{50\%}$ transition of the falling input voltage and the $V_{50\%}$ transition of the rising output voltage (Kang and Leblebici, 2005; DeMassa and Ciccone,1996)

Delay expressions can be obtained by solving the state equation of the output node in the time domain. In order to calculate fall time of the output voltage, output capacitance should be discharged through active NMOS transistor, as given in Fig. 4.2(a), considering PMOS is cut-off. As seen from Fig. 4.2(b), output fall time is calculated considering that NMOS operates in saturation mode during t_1 and in linear mode during t_2-t_1 time interval.

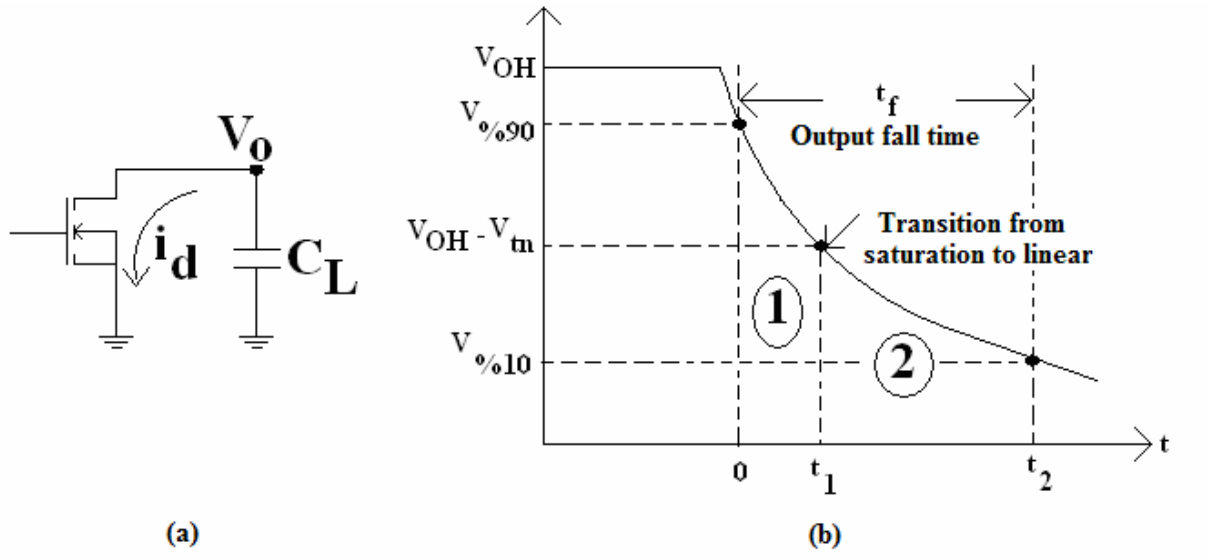

Fig. 4.2(a) Load capacitance discharges through NMOS transistor (b) Output voltage waveform during high-to-low transition (Kang and Leblebici, 2005)

Accordingly, total output fall time, t_f, is given as follows (DeMassa and Ciccone, 1996):

$$t_f = \frac{C_L}{\mu_n C_{ox}\left(\frac{W}{L}\right)_n (V_{DD} - V_{tn})}\left[\frac{2(V_{tn} - 0.1V_{DD})}{(V_{DD} - V_{tn})} + \ln\left(\frac{(2(V_{DD} - V_{tn})) - 0.1V_{DD}}{0.1V_{DD}}\right)\right] \quad (4.1)$$

In order to calculate rise time of the output voltage, output capacitance should be charged through active PMOS transistor, as given in Fig. 4.3(a), considering NMOS is cut-off. As seen from Fig. 3.b, output rise time is calculated considering that PMOS operates in saturation mode during t_3 and in linear mode during t_4-t_3 time interval.

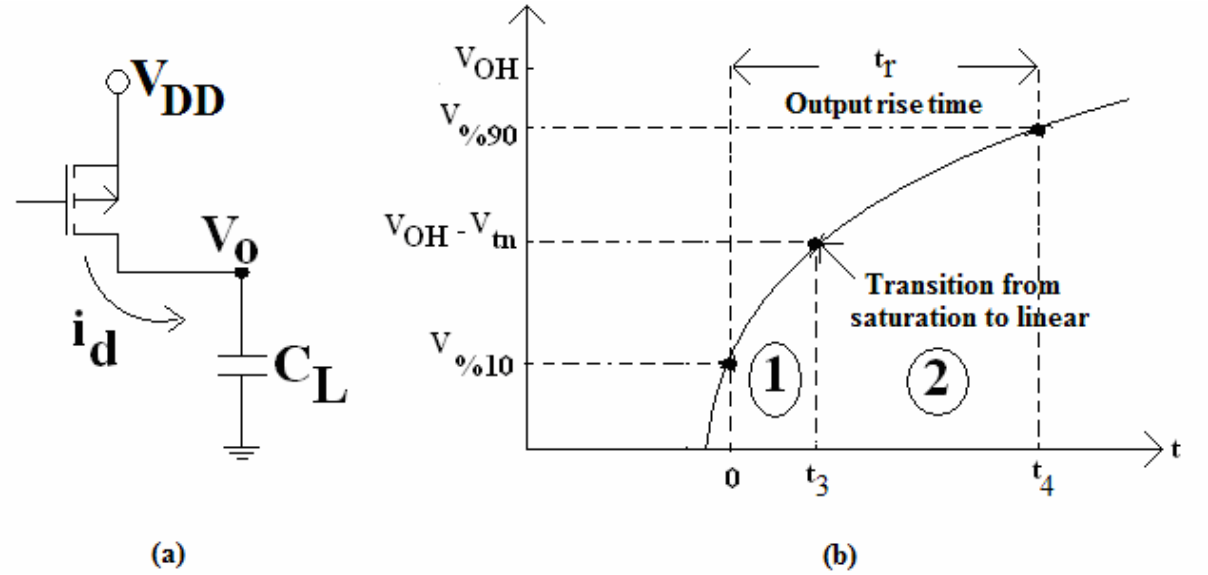

Fig. 4.3(a) Load capacitance charges through PMOS transistor (b) Output voltage waveform during low-to-high transition (Kang and Leblebici, 2005)

Accordingly, total output rise time, t_r, is given as follows (DeMassa and Ciccone, 1996):

$$t_r = \frac{C_L}{\mu_p C_{ox}\left(\frac{W}{L}\right)_p (V_{DD} - |V_{tp}|)}\left[\frac{2(|V_{tp}| - 0.1V_{DD})}{(V_{DD} - |V_{tp}|)} + \ln\left(\frac{(2(V_{DD} - |V_{tp}|)) - 0.1V_{DD}}{0.1V_{DD}}\right)\right] \quad (4.2)$$

Analysis of propagation delay times τ_{pHL} and τ_{pLH} also involves discharging output capacitance through active NMOS and charging output capacitance through active PMOS respectively. In order to simplify the derivation of delay expressions, the input voltage

waveform is usually assumed to be an ideal step pulse with zero rise and fall times. Under this assumption, τ_{pHL} becomes the time required for the output voltage to fall from V_{OH} to the $V_{50\%}$ level and τ_{pLH} becomes the time required for the output voltage to rise from V_{OL} to the $V_{50\%}$ level (Kang and Leblebici, 2005; DeMassa and Ciccone, 1996).

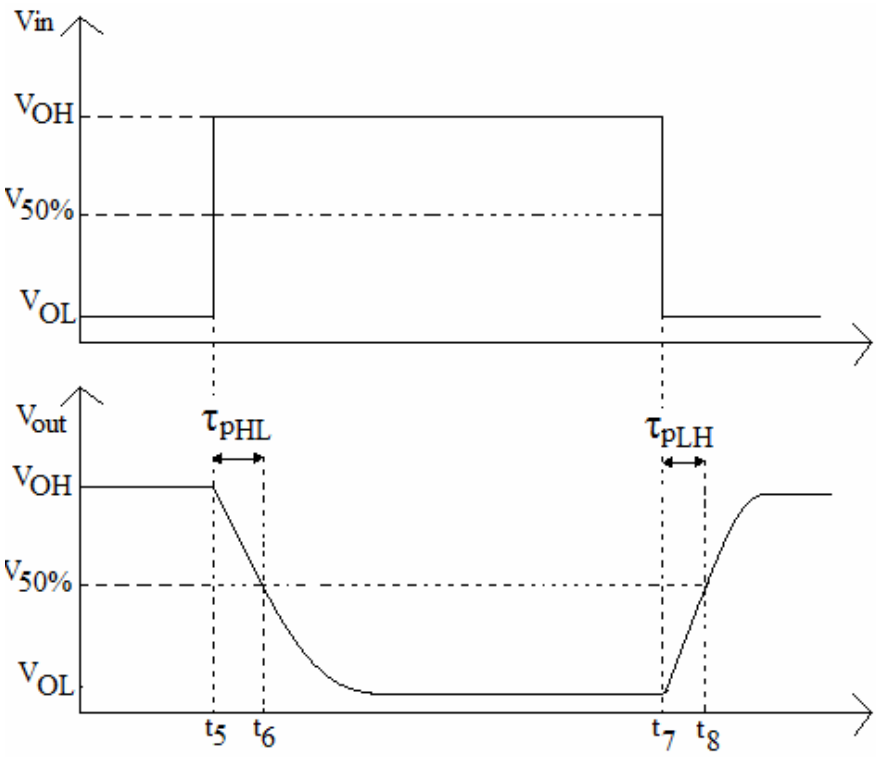

Fig. 4.4 Input and output voltage waveforms of a typical inverter and definitions of propagation delay times (DeMassa and Ciccone, 1996)

As seen from Fig. 4.4, the output propagation high-to-low delay time corresponds to t_6-t_5 time interval. Similarly t_8-t_7 time interval is equal to the output propagation low-to-high delay time. According to that, propagation delay times are given below (DeMassa and Ciccone, 1996):

$$t_{pHL} = \frac{C_L}{\mu_n C_{ox}\left(\frac{W}{L}\right)_n (V_{DD} - V_{tn})}\left[\frac{2V_{tn}}{(V_{DD} - V_{tn})} + \ln\left(\frac{4(V_{DD} - V_{tn})}{V_{DD}} - 1\right)\right] \tag{4.3}$$

$$t_{pLH} = \frac{C_L}{\mu_p C_{ox}\left(\frac{W}{L}\right)_p (V_{DD} - |V_{tp}|)}\left[\frac{2|V_{tp}|}{(V_{DD} - |V_{tp}|)} + \ln\left(\frac{4(V_{DD} - |V_{tp}|)}{V_{DD}} - 1\right)\right] \tag{4.4}$$

4.1.2 PSO Based Inverter Design

In order to investigate the usage of PSO in digital integrated circuit design, three case studies were carried out. In each case study, by establishing design criteria and design parameters to PSO and satisfying desired constraints, the optimal circuit structure was aimed to be designed by the algorithm. Design problem has been introduced to PSO algorithm by composing an equation consists of input variables and design parameters as a cost function (CF). In the beginning, a certain range was determined for both design criteria and design parameters by human designer. Input variables were also determined by the human designer and dependent to preferential technology parameters. PSO should minimize the given CF and obtain design criteria and design parameter values for the given range which gives minimum CF value.

4.1.2.1 Output Fall Time Estimation

In this case study, the aim was to estimate output voltage fall time of an inverter, as given in (4.1), with minimum error value (Vural et al, 2010d). During design process, values of design parameters fall time (t_f), output capacitance (C_L) and dimension ratio of MOS structures (equal sized NMOS and PMOS for this case-W/L) should be kept in certain ranges and PSO algorithm should find the solution set that consists the exact values of t_f, C_L and $(W/L)_n$ ratio for given ranges.

In order to obtain the cost function (CF) of this problem, (4.1) was reorganized that right side of it would be equal to zero, as given below:

$$CF = \left| (\mu C_{ox}) x \left(\frac{W}{L} \right)_n x\left(t_f\right) - \frac{C_L}{(V_{DD} - V_{tn})} x \left[\frac{2\left(0.1V_{DD} - V_{tn}\right)}{V_{DD} - V_{tn}} + \ln\left(\frac{2\left(V_{DD} - V_{tn}\right) - 0.1V_{DD}}{0.1V_{DD}} \right) \right] \right| \qquad (4.5)$$

The left side of (4.5) would constitute the CF which PSO algorithm would minimize. It was desired to obtain the exact values of design parameters which equate CF to a very close value to zero. Equating CF to zero means that error is also equal to zero and output voltage fall time is successfully estimated depending on the design parameters.

CF consists of fabrication technology parameters (V_{DD}, V_{tn}, $\mu_n C_{ox}$), design parameters ($(W/L)_n$, C_L) and design criterion (t_f) as given in Table 4.1.

Table 4.1 PSO inputs and outputs for Case Study-I

Components in CF	Information	Input/Output for PSO
V_{DD} V_{tn} $\mu_n C_{ox}$	• Set by human designer • Fabrication technology dependent	INPUT
C_L $(W/L)_n$ t_f	• PSO would find exact results for the given ranges	OUTPUT

Since TSMC 0.25 µm fabrication technology parameters [5] have been used, inputs of PSO were set as V_{DD}=2.5V, V_{tn}=0.3655V and $\mu_n C_{ox}$ = 243.6 µA/V^2. However, inputs would change for different technology parameters.

Initial population matrix size was 10x3 where row number of 10 indicates the number of particles in the population and column number of 3 is the dimension of particle vector which can be denoted as x = $[(W/L)_n, C_L, t_f]$. Velocity update parameters c_1, c_2 and w were 1.7, 1.7 and 0.99, respectively. The algorithm runs for upper limit of 250 iterations. Target error value was taken as 0.000001 and in all design experiments, CF results in less than the target error.

In this case study, estimation of output voltage fall time was performed for considering 10 different ranges of design parameters (C_L, (W/L)) and design criterion (t_f). Specified ranges and global best results (g_{best}) obtained as PSO synthesized results are tabulated in Table 4.2.

Table 4.2 Specified Ranges and PSO Synthesized Results for Case Study-I

	Specified Ranges			PSO Results		
	C_L (pF)	(W/L)	t_f (ns)	C_L (pF)	(W/L)	t_f (ns)
1	0.1-2.4	0.3-3.3	0.5-6.7	0.925	2.888	1.777
2	0.2- 5.6	0.4-2.3	0.3-6.0	0.861	1.352	3.536
3	0.6-3.4	0.9-5.0	0.6-8.6	3.322	4.745	3.885
4	0.5-3.6	1.2-4.1	0.9-11.0	3.379	4.052	4.627
5	**0.7-1.8**	**0.7-4.9**	**1.2-15.0**	**0.991**	**1.962**	**2.805**
6	0.3-2.4	2.2-3.2	1.4-12.0	1.395	2.847	2.719
7	0.4-1.87	0.3-4.9	0.4-7.7	0.638	4.274	0.828
8	1.1-2.3	1.8-6.9	0.4-6.5	1.305	2.349	3.083
9	0.7-2.3	0.7-3.0	1.6-5.7	0.947	2.591	2.028
10	0.6-1.9	1.5-3.5	1.0-8.15	1.653	2.473	3.709

As mentioned earlier, an initial population is configured for PSO algorithm and each particle in the population is a candidate for the solution. Regardless of how far the particle is away from the global solution; depending on velocity and position updates, most particles in the population gather around the global best value. For detailed investigation of PSO performance, fifth design ranges have been selected as an example. Table 4.3 shows the initial

population and personal best (p_{best}) results for all particles of these ranges. Final results of particles are all approximated to global best result (g_{best}).

Table 4.3 Evolution of Particles for 5th design of Case Study-I

Initial Population			**PSO result (p_{best} values)**		
C_L (pF)	***(W/L)***	***t_f (ns)***	***C_L (pF)***	***(W/L)***	***t_f (ns)***
0.9	2.44	3.77	0.991	1.962	2.805
72	27	19	0.991	1.962	2.805
38	17	14	0.991	1.962	2.805
48	17	11	0.982	1.937	2.813
16	24	48	0.993	1.963	2.808
467	62	12	0.986	1.952	2.802
5	6	7	0.991	1.962	2.805
54	12	24	0.991	1.958	2.810
63	14	28	1.071	2.101	1.822
35	342	68	1.061	1.847	3.190

For the fifth design ranges, PSO algorithm achieved CF value equal to target error at the iteration number of 247. In the first fifty iterations a sharp descent of error was observed. After the fiftieth iteration CF value approximated more slowly to target error. Fig. 4.5 shows how CF value (error information) decreases due to iteration number.

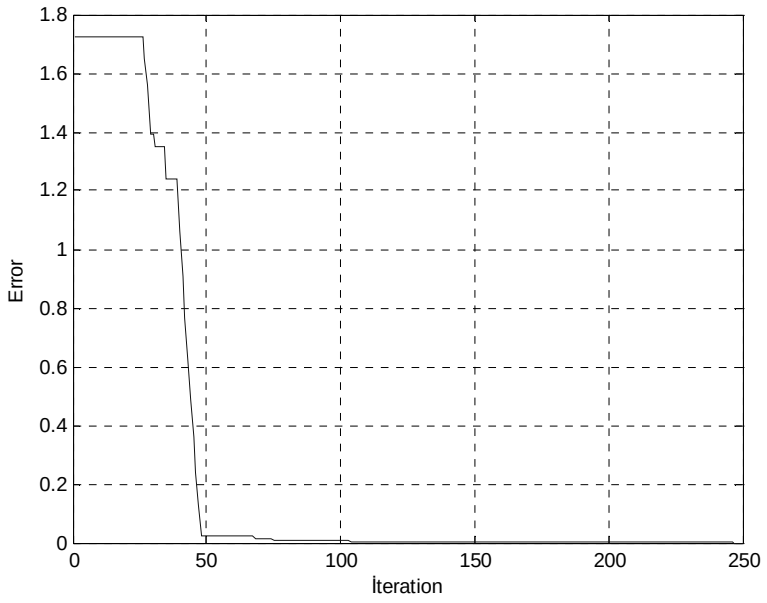

Fig. 4.5 CF value vs. iteration number for the 5th design ranges for Case Study-I

4.1.2.2 Symmetric Output Response: Identical Fall and Rise Times

Second case study is the PSO based design of a CMOS inverter having symmetrical output (Vural et al, 2010c). In order to achieve a symmetric dynamic response, the CMOS inverter should be designed in a way that fall time (t_f) and rise time (t_r) should be identical. However in practice, due to some second order effects, substrate-bias voltages or fabrication faults, a particular error between rise and fall time values is accepted. Our aim was to find design parameters which minimize the error between delay times (t_f, t_r). Design criteria are dynamic characteristics of inverter (t_f, t_r). Output capacitance (C_L) and MOS transistor dimensions ($(W/L)_n$, $(W/L)_p$) are design parameters. This case involves more constraints than the previous case. The design problem can be specified as follows:

Minimize
CF = absolute value of ($\boldsymbol{t_f}$ $(C_L, (W/L)_n) - \boldsymbol{t_r}$ $(C_L, (W/L)_p)$)
Subject to
Desired min $t_f <$ $\boldsymbol{t_f}$ $(C_L, (W/L)_n) <$ *Desired max* t_f
Desired min $t_r <$ $\boldsymbol{t_r}$ $(C_L, (W/L)_p) <$ *Desired max* t_r
Where
Desired min $C_L < \boldsymbol{C_L} <$ *Desired max* C_L
Desired min $(W/L)_n < \boldsymbol{(W/L)_n} <$ *Desired max* $(W/L)_n$
Desired min $(W/L)_p < \boldsymbol{(W/L)_p} <$ *Desired max* $(W/L)_p$

Algorithm would also provide design criteria results using the obtained design parameters which satisfy desired constraints as given in Table 4.4. Since TSMC 0.25µm fabrication technology parameters have been used, inputs of PSO algorithm were set as V_{DD}=2.5V, V_{tn}=0.3655V, $|V_{tp}|$ = 0.5466V, $\mu_n C_{ox}$ = 243.6 µA/V^2 and $\mu_p C_{ox}$ = 51.6 µA/V^2.

Table 4.4 PSO Inputs and Outputs for Case Study-II

Components in CF	Information	Input/Output for PSO
V_{DD} V_{tn} V_{tp} $\mu_n C_{ox}$ $\mu_p C_{ox}$	➢ Set by human designer ➢ Fabrication technology dependent	INPUT
C_L $(W/L)_n$ $(W/L)_p$ t_f t_r	➢ PSO would find exact results for the given ranges	OUTPUT

Initial population matrix size was 10x3. For this example, particle vector information can be denoted as x = [C_L, $(W/L)_n$, r] where $r = (W/L)_p/(W/L)_n$. Velocity update parameters c_1, c_2 and

w were 2, 2 and 0.99, respectively. The algorithm runs for upper limit of 9999 iterations. Maximum error value was taken as 0.15 ns and in all design experiments, CF results in less than the target error.

Here, inverter designs having symmetrical output voltage with different delay time and design parameter constraints are carried out using PSO. For each configuration, specified ranges and PSO synthesized results are tabulated in Table 4.5. As seen from this table, PSO obtained optimum design parameter values which minimize CF for all examples of design ranges.

Table 4.5 Specified Ranges and PSO Synthesized Results for Case Study-II

	Specified Ranges					**PSO Results**				
	C_L **(pF)**	$(W/L)_n$	r	t_f **(ns)**	t_r **(ns)**	C_L *(pF)*	$(W/L)_n$	r	t_f *(ns)*	t_r *(ns)*
1	**0.5-2.5**	**1.0-3.0**	**2.0-7.0**	**0.1-15**	**0.1-15**	**0.95**	**1.02**	**5.10**	**5.17**	**5.17**
2	0.5-1.5	1.0-2.5	2.0-5.5	0.1-15	0.1-15	0.77	1.13	5.02	3.77	3.78
3	0.5-1.5	1.0-3.0	2.0-7.0	0.1-15	0.1-15	1.17	1.05	5.00	6.18	6.19
4	1.0-3.0	1.5-3.5	2.5-6.0	0.1-15	0.1-15	1.63	1.63	5.01	5.54	5.54
5	1.5-3.5	1.5-3.0	2.0-6.2	0.1-10	0.1-10	2.54	1.81	5.00	7.79	7.79
6	0.3-2.0	1.0-2.0	2.0-6.0	0.1-8.0	0.1-8.0	0.95	1.02	5.00	5.17	5.18
7	0.6-1.9	1.5-3.5	2.0-5.8	0.1-7.5	0.1-7.5	1.12	2.83	5.00	2.20	2.20

The initial population of the first design ranges (as an example) and the corresponding p_{best} values at the maximum iteration number is tabulated in Table 4.6. For these design ranges, the final positions of all particles approximate to the global best value.

Table 4.6 Evolution of Particles for first design of Case Study-II

Initial Conditions			**PSO results (Global Best Value)**		
C_L*(pF)*	$(W/L)_n$	r	C_L*(pF)*	$(W/L)_n$	r
0.2	1.5	4.5	0.9505	1.0191	5.1029
7	27	9	0.9505	1.0190	5.1029
38	7	4	0.9509	1.0189	5.1028
48	17	1	0.9500	1.0190	5.1030
6	24	48	0.9478	1.0193	5.1041
467	62	2	0.9506	1.0190	5.1029
97	22	64	0.9501	1.0192	5.1040
54	12	24	0.9419	1.0194	5.1050
63	14	28	0.9139	1.0209	5.1122
35	342	68	0.9505	1.0190	5.1029

Here, PSO achieved CF value equal to target error at the iteration number of 150. In the first 45 iterations, a constant error (4.587) was obtained. Then, a sharp descent of error was observed. After the 50^{th} iteration, CF value approximated more slowly to target error. Fig. 4.6 depicts the decrement of CF value (error information) with the iteration number.

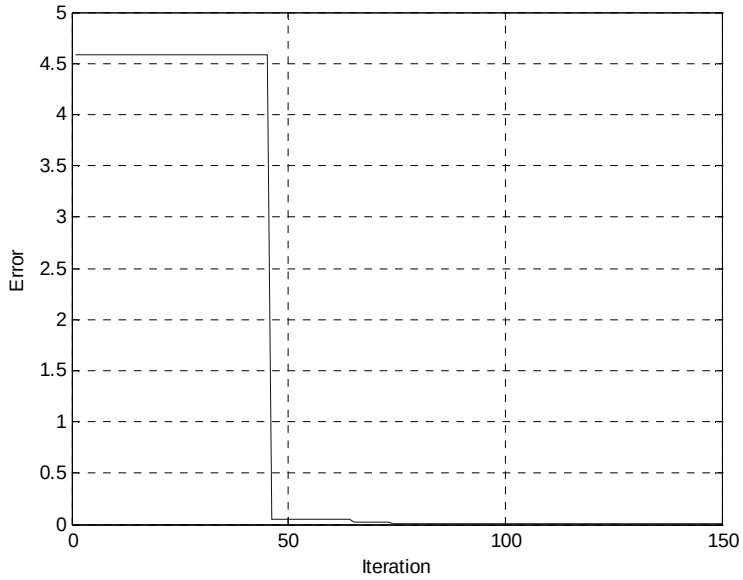

Fig. 4.6 CF value vs. iteration number for the first design in case study-II

4.1.2.3 Symmetric Output Response & Identical Propagation Delay Times

Previously, PSO algorithm has successfully estimated output voltage fall time in case study-I and has minimized the error between rise and fall times in case study-II. Here, error between rise and fall times and the error between propagation delay times (t_{pHL}, t_{pLH}) were minimized.

In this case, PSO should obtain design parameters which minimize both the error between fall and rise times (t_f, t_r) and the error between propagation delay times (τ_{pHL}, τ_{pLH}) (Vural et al, 2010d). PSO should find design parameters which minimize the error between output voltage fall and rise times and the error between propagation delay times. Design criteria were dynamic characteristics of inverter (t_f, t_r, τ_{pHL}, τ_{pLH}). Output capacitance (C_L) and MOS transistor dimensions ($(W/L)_n$, $(W/L)_p$) were design parameters. Design problem can be summarized as follows:

Minimize
CF = (absolute value of ($\boldsymbol{t_f}$ (C_L, $(W/L)_n$) – $\boldsymbol{t_r}$ (C_L, $(W/L)_p$)) +
absolute value of ($\boldsymbol{\tau_{pHL}}$ (C_L, $(W/L)_n$) – $\boldsymbol{\tau_{pLH}}$ (C_L, $(W/L)_p$)))
Subject to
Desired min t_f < $\boldsymbol{t_f}$ (C_L, $(W/L)_n$) < Desired max t_f
Desired min t_r < $\boldsymbol{t_r}$ (C_L, $(W/L)_p$) < Desired max t_r
Desired min τ_{pHL} < $\boldsymbol{\tau_{pHL}}$ (C_L, $(W/L)_n$) < Desired max τ_{pHL}
Desired min τ_{pLH} < $\boldsymbol{\tau_{pLH}}$ (C_L, $(W/L)_p$) < Desired max τ_{pLH}
Where
Desired min C_L < $\boldsymbol{C_L}$ < Desired max C_L
Desired min $(W/L)_n$ < $\boldsymbol{(W/L)_n}$ < Desired max $(W/L)_n$
Desired min $(W/L)_p$ <$\boldsymbol{(W/L)_p}$ < Desired max $(W/L)_p$

Depending on the given problem PSO algorithm would result in exact values of design parameters (C_L, $(W/L)_n$, $(W/L)_p$) which minimize CF values and satisfy specified constraints. PSO would also give exact values of t_f, t_r, τ_{pHL} and τ_{pLH} depending on the desired ranges.

For this case, PSO inputs and outputs were the same for case study-II; however, t_{pHL} and t_{pLH} have been included to outputs of PSO as given in Table 4.4. Fabrication technology parameters were the same as used in the previous case study.

Initial population matrix size was 10x3. Particle vector information can be denoted as $x = [C_L, (W/L)_n, (W/L)_p]$ as in the previous example. Velocity update parameters c_1, c_2 and w were 1.7, 1.7 and 0.99, respectively. The algorithm runs for upper limit of 500 iterations. Maximum error value was taken as 0.05 ns and in all experiments, except the eighth design ranges, CF resulted in less than the target error.

Here, inverter designs having symmetrical output voltage with different output voltage delay time, propagation delay time and design parameter constraints were carried out using PSO. For each configuration, specified ranges and PSO synthesized results are tabulated in Table 4.7. As seen from the table, PSO found optimum design parameter values which minimize CF for nine examples of design ranges; but in the eighth example, between specified ranges the best solution that PSO could find was slightly over the target error.

Table 4.7 Specified Ranges and PSO Synthesized Results for Case Study-III

	Specified Ranges							**PSO Results**						
	C_L (pF)	**$(W/L)_n$**	**$(W/L)_P$**	**t_f (ns)**	**t_r (ns)**	**t_{pHL} (ns)**	**t_{pLH} (ns)**	**C_L (pF)**	**$(W/L)_n$**	**$(W/L)_P$**	**t_f (ns)**	**t_r (ns)**	**t_{pHL} (ns)**	**t_{pLH} (ns)**
1	0.2-4	1.1-6.1	2.8-19.3	1.1-13	1.1-13	0.5-10	0.5-10	0.749	3.525	17.651	1.18	1.18	0.55	0.50
2	0.1-5.1	1.6-7.1	1.8-18	1.1-15	1.1-15	0.5-8	0.5-8	0.403	1.900	9.515	1.18	1.18	0.55	0.50
3	0.47-2	1.4-6.7	3.2-38	0.5-12	0.5-12	0.2-9	0.2-9	0.550	6.131	30.705	0.50	0.50	0.23	0.21
4	0.1-1.1	1.2-7.0	1.5-17.5	0.5-5	0.5-5	0.2-4	0.2-4	0.134	1.492	7.470	0.50	0.50	0.23	0.21
5	**0.2-1.4**	**1.9-5.0**	**2.7-17**	**0.7-6**	**0.7-6**	**0.4-5**	**0.4-5**	**0.512**	**3.011**	**15.079**	**0.94**	**0.94**	**0.44**	**0.40**
6	0.3-3.6	1.3-3.5	3.5-16.2	0.25-7	0.25-7	0.3-4.5	0.3-4.5	0.430	3.070	15.373	0.78	0.78	0.36	0.33
7	0.2-4.9	1.1-5.8	2.2-25.3	0.5-6.6	0.5-6.6	0.2-7.7	0.2-7.7	0.430	4.500	22.535	0.53	0.53	0.25	0.23
8	0.3-2.5	1.7-6	4.8-31	0.9-8.7	0.9-8.7	0.7-5.9	0.7-5.9	1.620	5.448	27.279	1.65	1.65	0.77	0.70
9	0.2-3.5	0.3-7.6	1.3-39	0.3-6.6	0.3-6.6	0.1-4.4	0.1-4.4	0.423	7.327	36.691	0.32	0.32	0.15	0.14
10	0.1-3.9	0.4-8	0.6-40	0.2-6	0.2-6	0.1-5.5	0.1-5.5	0.314	7.385	36.983	0.24	0.24	0.11	0.10

The initial population of the fifth design ranges (as an example) and the corresponding p_{best} values at the end of the maximum iteration number is given in Table 4.8. For these design ranges, the final positions of all particles gather around to the global best value. For the fifth design ranges, PSO algorithm achieved CF value (error information) equal to target error at the iteration number of 300. During first hundred iterations a sharp descent of error was observed in patches. After the hundredth iteration, CF value approximated more slowly to target error. Fig. 4.7 shows how CF value decreases due to iteration number.

Table 4.8 Initial and Final Positions of Particles for 5th design of Case Study-III

Initial Population			PSO Results (p_{best} values)		
C_L(pF)	$(W/L)_n$	$(W/L)_p$	C_L(pF)	$(W/L)_n$	$(W/L)_p$
1.7	5.5	9.5	0,512	3,011	15,079
72	2.7	1.9	0,512	3,011	15,079
3	33	14	0,512	3,011	15,079
5.98	0.17	11	0,513	3,017	15,107
16	24	48	0,512	3,011	15,079
743	8.4	1.2	0,512	3,011	15,079
25	17	70	0,512	3,011	15,079
54	12	24	0,512	3,011	15,079
6.3	14	2.8	0,512	3,011	15,079
35	2.2	68	0,512	3,011	15,079

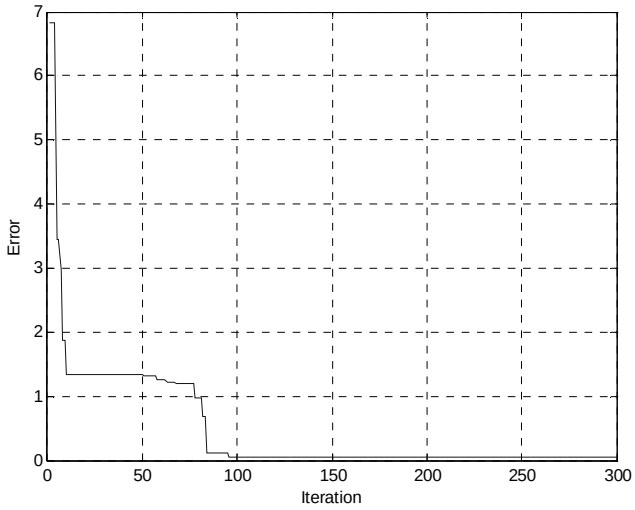

Fig. 4.7 CF value vs. iteration number for the 5th design in case study-III

4.1.3 Validation of PSO results with SPICE simulation

Using PSO results of output capacitance value and transistor dimensions, inverters were redesigned in SPICE simulator for each case, considering PSO results as SPICE inputs and obtained SPICE results were tabulated in Table 4.9, Table 4.10 and Table 4.11, respectively.(Vural et al. , 2010c; 2010d)

The difference between SPICE results and PSO-based design results arise from the fact that SPICE computes using more complex circuit equation sets than used in theoretical calculations. PSO uses delay expressions which were derived using simple current-voltage relationships originally developed for long-channel transistors. These current expressions based on the gradual channel approximation can still be used for sub-micron MOS transistors with proper parameter adjustments; therefore delay analysis used here remains largely valid for small-geometry devices as well. Yet it should be noted that the current driving capability

of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor can not be expected to have the same maximum charge/discharge current as a long-channel transistor with the same *(W/L)* ratio (Kang and Leblebici, 2005). Thus, SPICE results in greater delay times compared to PSO-based inverter design.

Table 4.9 SPICE Results vs. PSO Results for Case-I

	SPICE inputs		SPICE results (t_f-ns)	PSO results (t_f-ns)
	C_L (pF)	(W/L)		
1	0.925	2.888	4.846	1.777
2	0.861	1.352	8.016	3.536
3	3.322	4.745	10.128	3.885
4	3.379	4.052	11.677	4.627
5	0.991	1.962	6.846	2.805
6	1.395	2.847	7.103	2.719
7	0.638	4.274	2.339	0.828
8	1.305	2.349	7.824	3.083
9	0.947	2.591	5.257	2.028
10	1.653	2.473	9.487	3.709

Table 4.10 SPICE Results vs. PSO Results for Case-II

	SPICE inputs			SPICE results		PSO results	
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_f (ns)	t_r (ns)
1	0.95	1.02	5.2	10.94	8.58	5.17	5.17
2	0.77	1.13	5.67	8.32	6.58	3.77	3.78
3	1.17	1.05	5.25	12.95	9.99	6.18	6.19
4	1.63	1.63	8.17	13.59	9.32	5.54	5.54
5	2.54	1.81	9.06	19.27	12.87	7.79	7.79
6	0.95	1.02	5.10	10.944	8.58	5.17	5.18
7	1.12	2.83	14.16	6.09	4.09	2.20	2.20

Table 4.11 SPICE Results vs. PSO Results for Case-III

	SPICE inputs			SPICE results				PSO results			
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)
1	0.749	3.525	17.651	3.32	1.91	1.75	0.89	1.18	1.18	0.55	0.50
2	0.403	1.900	9.515	2.97	2.01	1.58	0.87	1.18	1.18	0.55	0.50
3	0.550	6.131	30.705	1.50	0.82	0.79	0.39	0.50	0.50	0.23	0.21
4	0.134	1.492	7.470	1.22	0.89	0.64	0.38	0.50	0.50	0.23	0.21
5	0.512	3.011	15.079	2.69	1.64	1.37	0.73	0.94	0.94	0.44	0.40
6	0.430	3.070	15.373	2.14	1.43	1.14	0.60	0.78	0.78	0.36	0.33
7	0.430	4.500	22.535	1.53	0.89	0.82	0.42	0.53	0.53	0.25	0.23
8	1.620	5.448	27.279	4.73	2.78	2.52	1.27	1.65	1.65	0.77	0.70
9	0.423	7.327	36.691	0.98	0.60	0.53	0.26	0.32	0.32	0.15	0.14
10	0.314	7.385	36.983	0.81	0.45	0.40	0.19	0.24	0.24	0.11	0.10

4.2 Particle Swarm Optimization for Analog Integrated Circuit Design

Analog integrated circuit (IC) design in general is perceived as less systematic and more heuristic and knowledge-intensive in nature than digital IC design. The variety of circuit schematics and the number of conflicting requirements and corresponding diversity of device sizes are also much larger. In addition analog circuits are more sensitive to nonidealities and all kinds of higher order effects and parasitic disturbances (Gielen and Rutenbar, 2000).

The problem considered here is the optimal selection of transistor dimensions, which is only a part of a complete analog circuit CAD tool. Other parts which are beyond the scope of this work are the topology selection (Maulik et al., 1992) and actual circuit layout (Harvey et al., 1992). The optimal component selection and transistor sizing of the CAD process remains between these two tasks (Medeiro et al., 1994). Actually, analog sizing is a constructive procedure that aims at mapping the circuit specifications (objectives and constraints on performances) into the design parameter values (Fakhfakh et al., 2009). In other words, the performance metrics of the circuit, such as gain, power dissipation, occupied area, etc. have to be formulated in terms of the design parameters (Tulunay and Balkir, 2004). Then, these design parameters such as device sizes and bias currents should be adjusted under multiple design objectives and constraints. The many degrees of freedom in parameter space as well as the need for repeated circuit performance evaluation made this a lengthy and tedious process (Koh et al., 1990). Here, particular specifications for specified topologies of a differential amplifier and an operational amplifier are aimed to be met by adjusting design parameters such as device sizes and bias currents with PSO algorithm.

4.2.1 Analog Integrated Circuit Structures

The most often used analog building block is the operational amplifier (op-amp). It is at the heart of many interface circuits in particular DAC, ADC and filters. An efficient design of optimal op-amp is thus a cornerstone of a design environment for many applications. Differential amplifier serves as the input stage of op-amps. The objective of a differential amplifier is to multiply the difference between two inputs by some constant factor regardless of the average of the inputs.

4.2.1.1 Differential Amplifier Structure

The differential amplifier is one of the most versatile circuits in analog circuit design. It is also very compatible with IC technology and serves as the input stage to most op-amps. The configuration considered in this study is a differential amplifier with a current mirror load which is provided in Fig.4.8. The design specifications and also general characteristics for a differential amplifier (and also for op-amp) are stated below (Allen and Holberg, 2002).

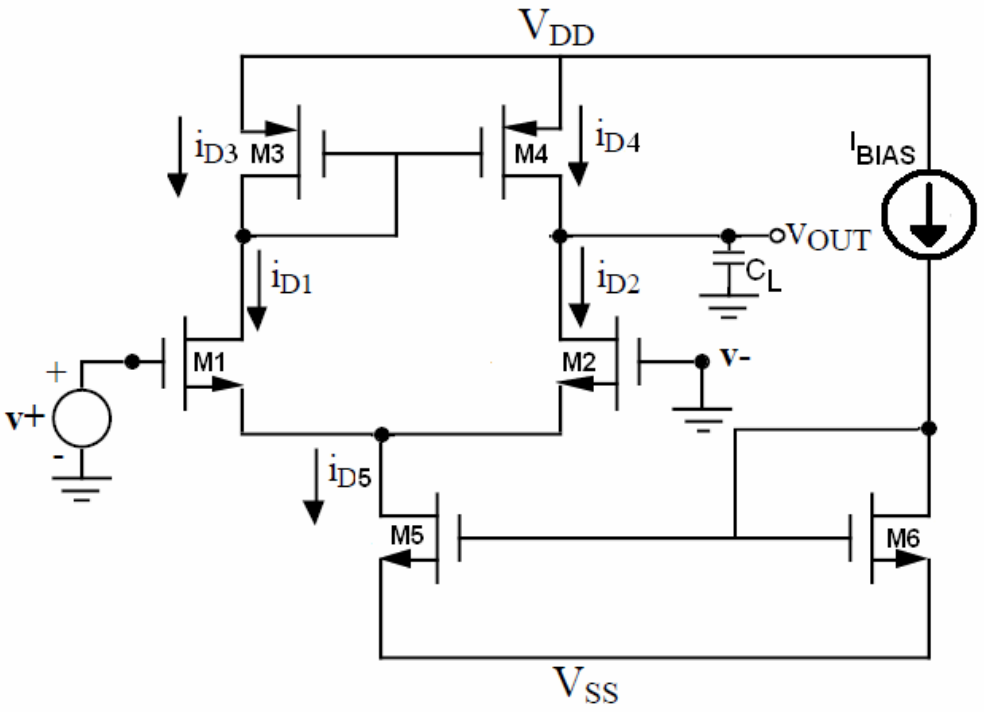

Fig. 4.8 Differential Amplifier with Current Mirror Load (Allen and Holberg, 2002)

- *Common Mode Rejection Ratio (CMRR)*: Fig. 4.9(a) depicts a schematic model for a differential amplifier (this symbol will also be used for op amp).

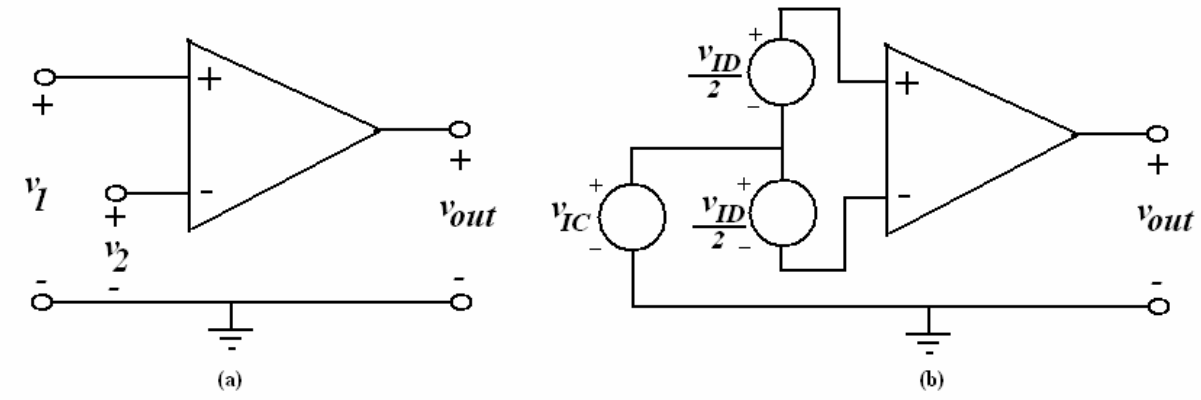

Fig. 4.9(a) Symbol for a differential amplifier (b) Illustration of the differential mode, v_{ID} and common mode, v_{IC}, input voltages (Allen and Holberg, 2002)

Voltages v_1, v_2 and v_{out} are called single ended voltages which mean that they are defined with respect to the ground. The differential-mode input voltage v_{ID} of the differential amplifier is defined as the difference between v_1 and v_2 (4.6). This voltage is defined between two terminals, neither of which is ground. The common-mode voltage v_{IC} is defined as the average value of v_1 and v_2 (4.7).

$$v_{ID} = v_1 - v_2 \tag{4.6}$$

$$v_{IC} = \frac{v_1 + v_2}{2} \tag{4.7}$$

Single ended voltages v_1 and v_2 can be expressed as given in (4.8) and (4.9) respectively.

$$v_1 = v_{IC} + \frac{v_{ID}}{2} \tag{4.8}$$

$$v_2 = v_{IC} - \frac{v_{ID}}{2} \tag{4.9}$$

The output voltage of the differential amplifier is expressed in terms of its differential-mode and common-mode input voltages as in (4.10).

$$v_{out} = A_{VD} v_{ID} \pm A_{VC} v_{IC} = A_{VD}(v_1 - v_2) \pm A_{VC}\left(\frac{v_1 + v_2}{2}\right) \tag{4.10}$$

where A_{VD} is the differential-mode gain and A_{VC} is the common-mode voltage gain. The ± sign preceding the common-mode voltage gain implies that the polarity of this voltage gain is not known beforehand.

The objective of a differential amplifier is to multiply the difference between two different potentials regardless of the common-mode value. Thus a differential amplifier can be characterized by its *common-mode rejection ratio* (CMRR) which is the ratio of the magnitude of the differential gain to the common-mode gain. An ideal differential amplifier will have a zero value of A_{VC} and therefore an infinite CMRR.

- *Input Offset Voltage (V_{OS}):* Another characteristic affecting the performance of the differential amplifier is the offset voltage. Ideally when input terminals of a differential amplifier are connected together, the output voltage is at a desired quiescent point. In a real differential amplifier, the output offset voltage is the difference between the actual output voltage and the ideal output voltage when the input terminals are connected together. If this offset voltage is divided by the differential voltage gain of the differential amplifier then it is

called the *input offset voltage* (V_{OS}). Typically, the input offset voltage of a CMOS differential amplifier is 5-20 mV.

- *Slew Rate (SR)* : The slew rate performance of a CMOS differential amplifier depends on the value of I_{SS} (I_{SS}=I_{D5} from Fig. 4.8) and the capacitance from the output node to the AC ground. *SR* is determined by the amount of current that can be sourced or sunk into the output capacitor and its expression is given in (4.11)

$$SR = I_{SS} / C_L \tag{4.11}$$

- *Power Dissipation (P_{diss})*: The quiescent power dissipation of the differential amplifier is described in (4.12).

$$P_{diss} = (V_{DD} + |V_{SS}|)(I_{bias} + I_{SS}) \tag{4.12}$$

- *Small Signal Characteristics (A_v&ω_{-3dB})*: The small signal analysis of the differential amplifier of Fig.4.8 (without M6 and I_{bias}) can be accomplished with the assistance of the model (ignoring body effect and intrinsic capacitances) shown in Fig. 4.10 (a). This model can be simplified to that shown in Fig. 4.10 (b) and is only appropriate for differential analysis when both sides of the amplifier are assumed to be perfectly matched. Despite the fact that the current mirror causes this assumption to be invalid because the drain loads of M1 and M2 are not matched this assumption is still valid in (Allen and Holberg, 2002).

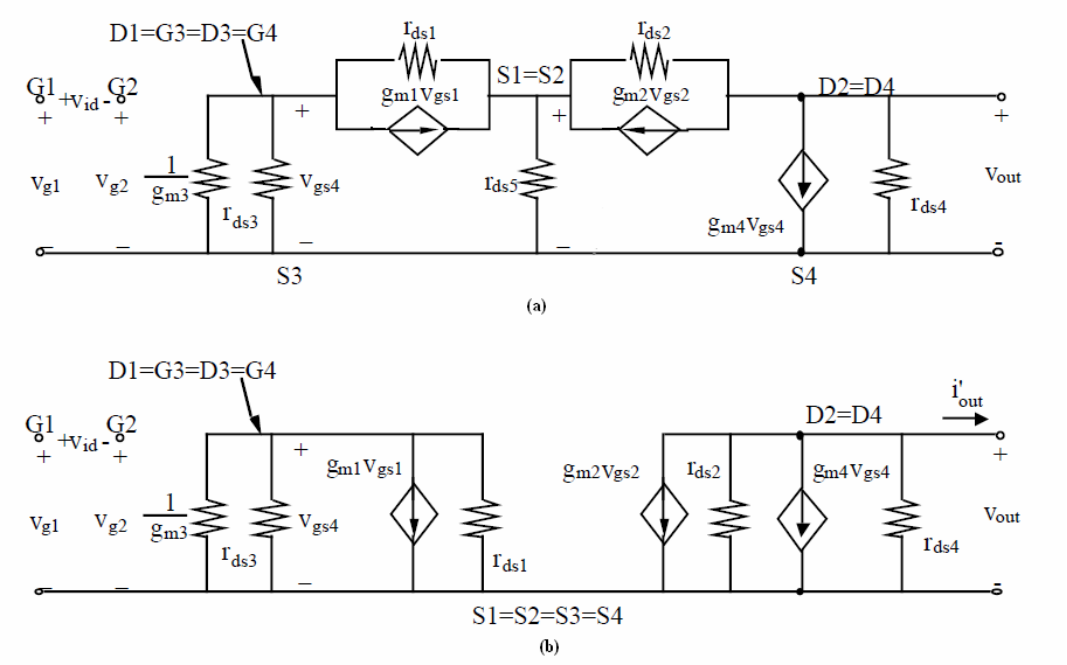

Fig. 4.10 Small signal model for the CMOS differential amplifier (a) Exact model (b) Simplified equivalent model.

The point where two sources of M1 and M2 are connected can be considered to be at AC ground. If the differential stage is assumed to be unloaded, then with the output is shorted to AC ground, the differential-transconductance gain can be expressed as in (4.13)

$$i_{out}^{'} = -g_{m4}v_{gs4} - g_{m2}v_{gs2} = \frac{g_{m1}g_{m4}(r_{ds1} \parallel r_{ds3})}{1 + g_{m3}(r_{ds1} \parallel r_{ds3})}v_{gs1} - g_{m2}v_{gs2} \tag{4.13}$$

If $g_{m3}(r_{ds1} \parallel r_{ds3}) >> 1$, $g_{m3} = g_{m4}$, and $g_{m1} = g_{m2} = g_{md}$, then (4.14) or (4.15) will be valid.

$$i_{out}^{'} \approx g_{m1}v_{gs1} - g_{m2}v_{gs2} = g_{md}(v_{gs1} - v_{gs2}) = g_{md}v_{id} \tag{4.14}$$

$$i_{out}^{'} \approx g_{md}v_{id} = \sqrt{\frac{K^{'}WI_{ss}}{L}}v_{id} \quad , K_{n,p}^{'} = \mu_{n,p}C_{ox} \tag{4.15}$$

The unloaded differential voltage gain can be determined by finding the small-signal output resistance of the differential amplifier. From Fig. 4.10 (b) r_{out} can be obtained as in (4.16)

$$r_{out} = \frac{1}{g_{ds2} + g_{ds4}} \tag{4.16}$$

Therefore, the voltage gain is given as the product of g_{md} and r_{out} as in (4.17)

$$A_v = \frac{v_{out}}{v_{id}} = \frac{g_{md}}{g_{ds2} + g_{ds4}} = \frac{(K_1^{'}I_{ss}W_1 / L_1)^{1/2}}{(\lambda_2 + \lambda_4)(I_{ss}/2)} = \frac{2}{(\lambda_2 + \lambda_4)}\left(\frac{K_1^{'}W_1}{I_{ss}L_1}\right)^{1/2} \tag{4.17}$$

Cut-off frequency is the frequency at which the gain drops 3 dB below the open loop gain and expressed in terms of r_{out} and C_L as given in (4.18)

$$\omega_{-3dB} = \frac{1}{r_{out}C_L} \tag{4.18}$$

Unity gain bandwidth (ω_t)is defined as the frequency at which $|H(j\omega_t)|=1$ where H represents the transfer function. The phase margin (PM) is defined in terms of the phase of the transfer function at the unity-gain bandwidth as in (4.19)

$$PM = \pi - \angle H(j\omega_t) \tag{4.19}$$

A phase margin constraint specifies a lower bound on the phase margin typically between 30^o and 60^o.

- *Input Common Mode Range (ICMR)*: *ICMR* specifies over what range of common-mode voltages the differential amplifier continues to sense amplify the difference signal with the same gain. The way that the ICMR is found is to set V_{ID} to zero and vary V_{IC} until one of the transistors in the differential amplifier is no longer saturated. The highest common-mode range can be obtained by following two paths from V_{IC} to V_{DD}. The path with a smaller value of $V_{IC\ (max)}$ will be selected from a worst-case viewpoint and is given in (4.20). The lowest input voltage at the gate of M1 or M2 is found as in (4.21)

$$V_{IC\ (max)} = V_{DD} - V_{SG3} + V_{TN1} \tag{4.20}$$

$$V_{IC\ (min)} = V_{SS} + V_{DS5\ (sat)} + V_{GS1} = V_{SS} + V_{DS5\ (sat)} + V_{GS2} \tag{4.21}$$

- *Power Supply Rejection Ratio (PSRR)*: *PSRR* is a measure of how well a circuit rejects ripple coming from the input power supply at various frequencies and is very critical in many RF and wireless applications. For amplifiers with both positive and negative power supplies (with respect to earth), the PSRR for each supply voltage may be separately specified (sometimes written: PSRR+ and PSRR-), but normally the PSRR is tested with opposite polarity signals applied to both supply rails at the same time [6].

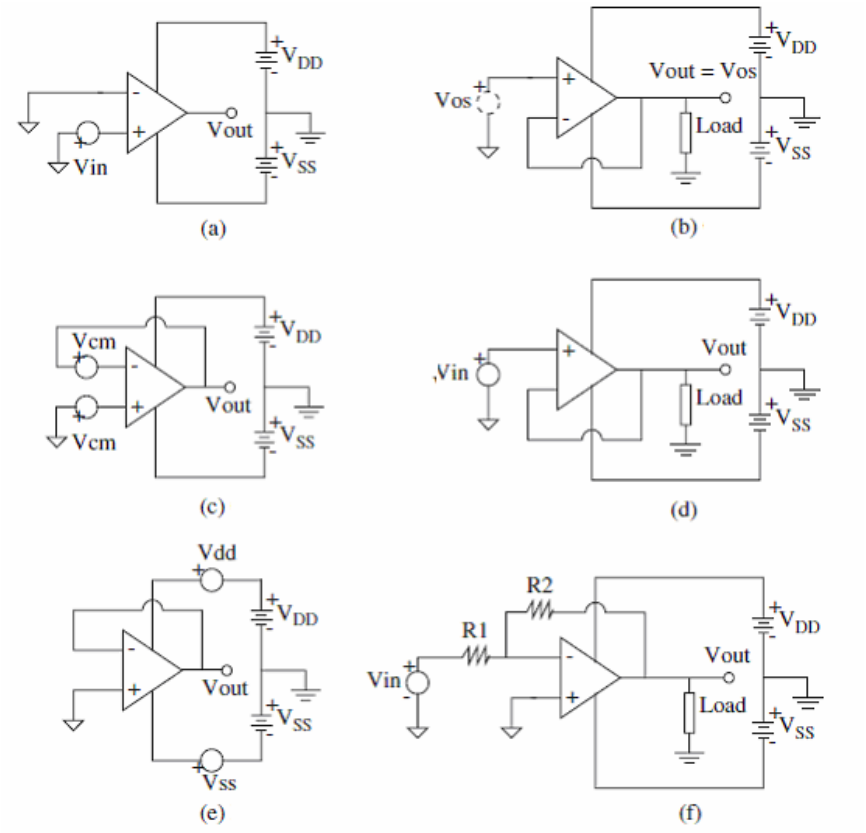

Fig. 4.11 Configurations for simulating design specifications for (a) Gain and Phase Margin (b) Offset voltage (c) CMRR (d) ICMR (e) PSRR (f) Output voltage swing

4.2.1.2 Two Stage Operational Amplifier Structure

The specific two stage CMOS op-amp considered in this study is given in Fig.4.12. The circuit consists of an input differential stage with active load followed by a common-source stage also with active load. An output buffer is not used; this amplifier is assumed to be part of a very large scale integration (VLSI) system and is only required to drive a fixed on-chip capacitive load of a few picofarads (Hershenson et al, 2001; Allen and Holberg, 2002). This op-amp architecture has many advantages: high open-loop voltage gain, rail-to-rail output swing, large common-mode input range, only one frequency compensation capacitance, and a small number of transistors. However its main drawback is the nondominant pole formed by the load capacitance and the output impedance of the second stage which reduces the achievable bandwidth (Hershenson et al, 2001).

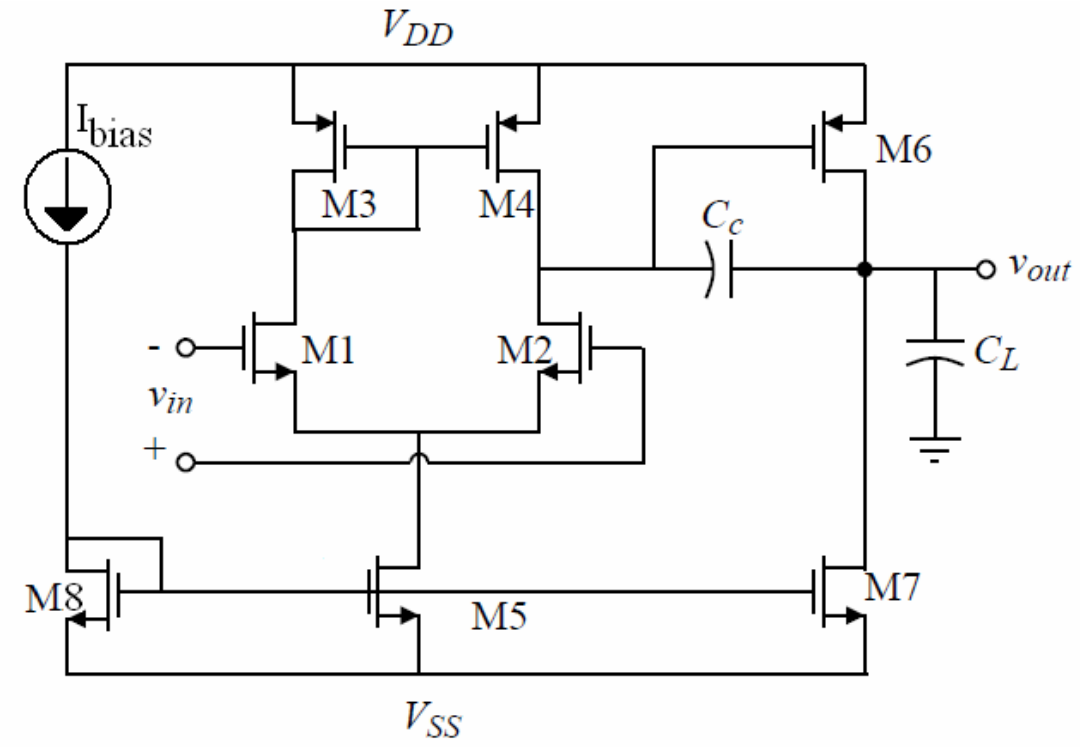

Fig.4.12 Two-stage Operational Amplifier (Allen and Holberg, 2002)

Operational amplifiers are generally used in a negative-feedback configuration. In this way, the relatively high, inaccurate forward gain can be used with feedback to achieve a very high accurate transfer function that is a function of the feedback elements only. Ideally, the phase characteristic of an amplifier's frequency response would be constant; however, device limitations make this goal physically unattainable. More particularly, capacitances within the gain stages of an amplifier cause the output signal to lag behind the input signal by 90° for

each pole they create. If the sum of these phase lags reaches 360°, the output signal will be in phase with the input signal. Feeding back any portion of this output signal to the input when the gain of the amplifier is sufficient will cause the amplifier to oscillate. This is because the feedback signal will reinforce the input signal. That is, the feedback is then positive rather than negative. Frequency compensation is implemented to avoid this result. In Fig.4.12, Miller compensation technique is utilized by connecting a capacitance (C_c) from the output to the input of the second transconductance stage.

4.2.2 Design Procedure for Analog Integrated Circuits

In this subsection, design steps and design equations utilized for cost functions of Particle Swarm Optimization based analog integrated circuit design are explained. These design equations will be considered for obtaining MOS dimensions (as in Fig.4.8 and Fig.4.12) and moreover minimizing the total MOS transistor area.

4.2.2.1 Design Procedure for Differential Amplifier

Design constraints of small-signal differential voltage gain (A_v), cut-off frequency (f_{-3dB}), maximum and minimum input common mode voltages ($V_{IC(max)}$, $V_{IC(min)}$), slew rate (SR) power dissipation (P_{diss}) and design parameters of output capacitance (C_L) and MOS device sizes are provided within limits. Design procedure is given below:

- Determine range of I_{d5} (I_{ss}) to satisfy the slew rate (SR) and power dissipation (P_{diss}).

$$I_{dmin} < I_{d5} < I_{dmax} \tag{4.22}$$

where

$$I_{dmax} = P_{diss} / (V_{DD} + |V_{SS}|); \tag{4.23}$$

$$I_{d\min} = \max\left(SRxC_L, \left(2 / \left(\left(\lambda_n + \lambda_p \right) \; x\left(1 / \left(2\pi f_{-3dB} C_L \right) \right) \right) \right) \right) \tag{4.24}$$

- Design W_1/L_1 (W_2/L_2) to satisfy A_v from (4.17)
- Design W_3/L_3 (W_4/L_4) to satisfy the upper ICMR as from (4.20)
- Design W_5/L_5 (W_6/L_6) to satisfy the lower ICMR as from (4.21)
- Obtain exact values of design parameters and iterate if necessary

4.2.2.2 Design Procedure for Two-Stage Operational Amplifier

Here, design constraints of small-signal differential voltage gain (A_v), unity gain bandwidth (f_t), maximum and minimum input common mode voltages ($V_{IC(max)}$, $V_{IC(min)}$), slew rate (SR) power dissipation (P_{diss}) and design parameters of output capacitance (C_L) and MOS device sizes are provided within limits. Design procedure is given below:

- Choose minimum value for the compensation capacitor Cc. Placing the loading pole p_2 2.2 times higher than the f_t, permitted a 60° phase margin, assuming that the right half plane (RHP) zero $z1$ is placed at or beyond ten times f_t (Allen and Holberg,2002).

$$C_c > (2.2/10)C_L \tag{4.25}$$

- Determine I_{d5} (I_{ss}) to satisfy the slew rate (SR)

$$I_{d5}=C_c*SR \tag{4.26}$$

- Design W_1/L_1 (W_2/L_2) using the transconductance of the differential input stage

$$(W_1 / L_1) = (W_2 / L_2) = \frac{g_{m1}^2}{2K_1^{'}I_1} \tag{4.27}$$

where

$$g_{m1} = 2\pi f_t C_c \tag{4.28}$$

- Design W_3/L_3 (W_4/L_4) to satisfy the upper ICMR as from (4.20)
- Design W_5/L_5 (W_8/L_8) to satisfy the lower ICMR as from (4.21)
- Design W_6/L_6 assuming balanced conditions

$$(W_6 / L_6) = (W_4 / L_4)\frac{g_{m6}}{g_{m4}} \tag{4.29}$$

where

$$g_{m6} \geq 10 g_{m1} \text{ (assuming zero } z1 \text{ is placed beyond ten times } f_t \text{ (Allen and Holberg, 2002))} \tag{4.30}$$

$$g_{m4} = \sqrt{2K_4^{'}(W_4 / L_4)I_{d4}} \tag{4.31}$$

- Calculate I_{d6} which will most likely determine the majority of the power dissipation.

$$I_{d6} = \frac{g_{m6}^2}{2K_6^{'}(W_6 / L_6)} \tag{4.32}$$

- Design W_7/L_7 to achieve the desired current ratios between I_{d5} and I_{d6}

$$(W_7 / L_7) = (W_5 / L_5)\frac{I_6}{I_5} \tag{4.33}$$

- Check gain and power dissipation specifications

$$A_v = \frac{2g_{m2}g_{m6}}{I_{d5}(\lambda_2 + \lambda_3)(\lambda_6 + \lambda_7)} \tag{4.34}$$

$$P_{diss} = (I_{bias} + I_{d5} + I_{d6})(V_{DD} + |V_{SS}|) \tag{4.35}$$

- Obtain exact values of design parameters and iterate if necessary

4.2.3 Simulation Results of PSO Based Analog Integrated Circuit Design

In order to investigate the usage of PSO in analog integrated circuit design, optimal design of two basic analog circuit structures studies are carried out. The aim of both cases is to minimize total IC area while satisfying design criteria and design parameter constraints. In each case study, by establishing design criteria and design parameters to PSO, the optimal circuit structure was aimed to be designed by the algorithm. Design problem has been introduced to PSO algorithm by composing an equation consists of input variables and design parameters as a cost function (CF). In the beginning of the algorithm a certain range was determined for both design criteria and design parameters by human designer. Input variables were also determined by the human designer and dependent to preferential technology parameters. PSO should minimize the given CF and obtain design criteria and design parameter values for the given range which gives minimum CF value.

4.2.3.1 Simulation Results of PSO based Differential Amplifier Design

The starting point of design consists of two types of information. First type of information such as the technology and the power supply is set by the designer. The other type of information is the design criteria. The range of each criteria and design parameter, power

supply values and technology information is set as an input to PSO algorithm (Table 4.12) and PSO algorithm should obtain the solution set that consists the exact values of design parameters (C_L, $(W/L)_{1..6}$) and design criteria (f_{-3dB}, $V_{IC(max)}$, $V_{IC(min)}$, SR, P_{diss}, A_v) for given ranges. The design is implemented with the relationships that describe design specifications to solve for DC currents and W/L values of all MOS transistors. The appropriate relationships were provided in the previous subsections.

Table 4.12 Inputs and Outputs for PSO based Differential Amplifier Design

Components in CF	Information	Input/Output for PSO
V_{DD}, V_{SS} V_{tn}, V_{tp} $\mu_n C_{ox}$, $\mu_p C_{ox}$ λ_n, λ_p	➢ Set by human designer ➢ Fabrication technology dependent	INPUT
C_L $(W/L)_{1..6}$ f_{-3dB} $V_{IC(max)}$, $V_{IC(min)}$ SR P_{diss} A_v	➢ PSO would find exact results for the given ranges	OUTPUT

PSO is utilized for a differential amplifier with current mirror load having design specifications of $SR \geq 10V/\mu s$ ($C_L=5pF$), $f_{-3dB} \geq 100$ kHz ($C_L=5pF$), $A_v > 100$ V/V, $P_{diss} \leq 1mW$, $-1.5V \leq ICMR \leq 2V$ with inputs of $V_{DD}= -V_{SS}=2.5V$, $V_{tn}= 0.7V$, $V_{tp}= -0.7V$, $K_n'= 110\mu A/V^2$, $K'_p= 50\mu A/V^2$, $\lambda_n=0.04\ V^{-1}$, $\lambda_p=0.05\ V^{-1}$ (Allen and Holberg, 2002). Constraints for design parameters are set as $C_L>=5pF$, $100 \geq (W/L)_{1..6} \geq 3$. In order to minimize the channel modulation effect, MOSFET length values are chosen as $L_{1..4}=3.5\ \mu m$ and $L_{5..6}=1.4\ \mu m$.

Initial population matrix size was 10x7 where row number of 10 indicates the number of particles in the population and column number of 7 is the dimension of particle vector which is expressed as in (4.36) where SR is the slew rate (V/µm), C_L is the output capacitance (pF), A_v is the gain (V/V), f_{-3dB} is the cut-off frequency (kHz), P_{diss} is the power dissipation (µW), V_{icmin} and V_{icmax} (V) are the lower and upper limits of ICMR, respectively

$$x = [SR, P_{diss}, C_L, f_{-3dB}, A_v, V_{icmin}, V_{icmax}] \tag{4.36}$$

Velocity update parameters c_1, c_2 and w were 1.7, 1.7 and 0.99, respectively. The algorithm runs for upper limit of 750 iterations. CF is defined as the total area that MOS transistors occupy and given in (4.37)

$$CF = \sum_{k=1}^{T} \left(W_{(k)} x L_{(k)} \right) \tag{4.37}$$

where T is the total number of MOS transistors in differential amplifier design.

PSO based differential amplifier with current mirror load design results are provided in Fig.4.13. Exact values of design parameters obtained from PSO based design are given below:

- $(W1/L1)=(W2/L2)=6.7294(\mu m/\mu m)$
- $(W3/L3)=(W4/L4)=1.6492(\mu m/\mu m)$
- $(W5/L5)=(W6/L6)=1.7327(\mu m/\mu m)$
- $C_L=5\ .23pF$
- $I_{bias}=73\ \mu A$

Target value of CF is aimed to be smaller 65 μm^2. A total MOS transistor area of 63.5016 μm^2 with each (W/L) ratio and I_{bias} is obtained within 629 epochs. Those results are compared with classical method (Allen and Holberg, 2002) in Table 4.13.

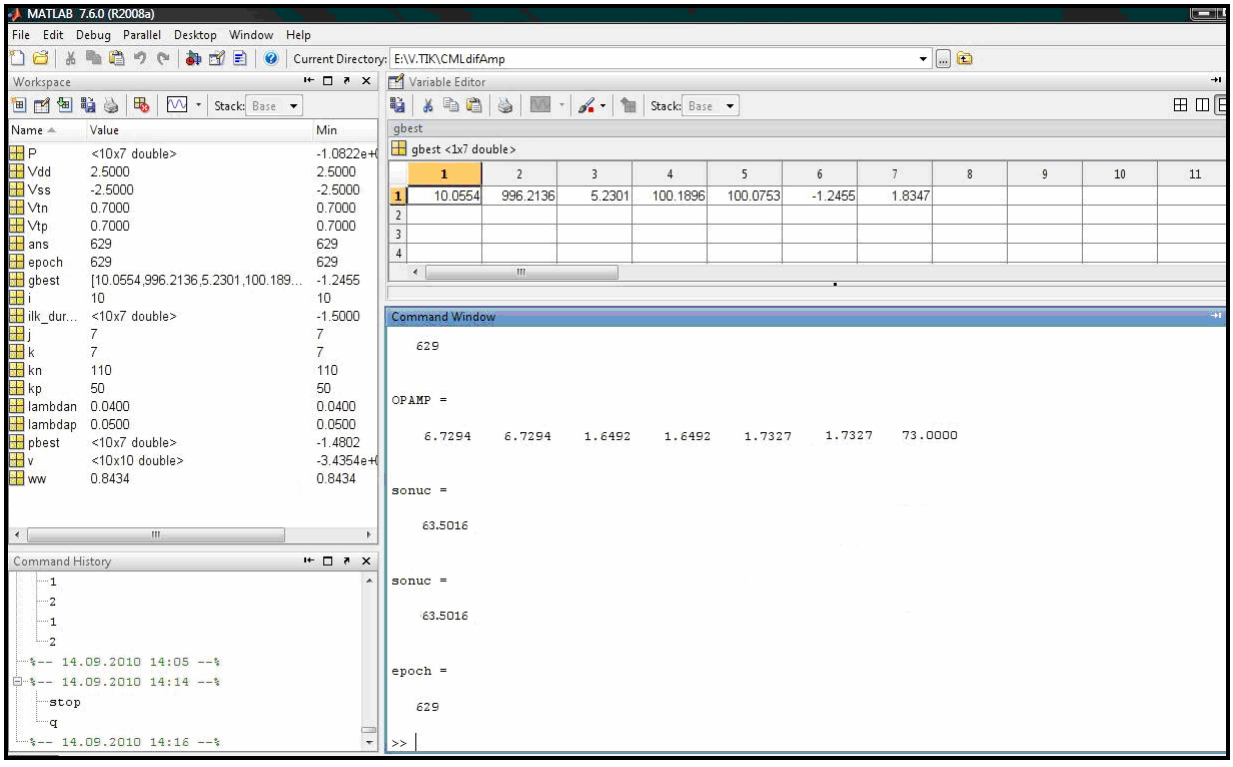

Fig. 4.13 PSO Based Differential Amplifier with Current Mirror Load Design Results-I

Table 4.13 Comparison of classical method and PSO based method by means of design specifications

Differential Amplifier with Current Mirror Load Design Criteria	**Spec**	**Classical design**	**PSO based design**
Output Capacitance (pF)	≥5	5	5.23
Slew Rate (V/µs)	≥ 10	10	10.0554
Power Dissipation (µW)	≤ 1000	1000	996
Cut-off Frequency (kHz)	≥100	100	100.1896
Gain (dB)	> 40	43.39	41
Vic_{min} (V)	≥-1.5	-1.5	-1.2455
Vic_{max} (V)	≤2	2	1.8347
Total Area (m^2)	**<6.5x10^{-11}**	**-**	**6.35x10^{-11}**

Table 4.14 Comparison of classical method and PSO based method by means of design parameters

Differential Amplifier with Current Mirror Load Design Parameters	**Classical design**	**PSO based design**
I_{bias} *(µA)*	100	73
W1/L1 (µm/µm)	40	6.7294
W2/L2 (µm/µm)	40	6.7294
W3/L3 (µm/µm)	8	1.6492
W4/L4 (µm/µm)	8	1.6492
W5/L5 (µm/µm)	82	1.7327
W6/L6 (µm/µm)	82	1.7327
C_L *(pF)*	5	5.23

PSO based design minimizes the total MOS transistor area when compared with classical design results providing that design constraints are satisfied. Since the technology parameters are taken from (Allen and Holberg, 2002) for a true comparison; it is not clear whether the design method will work using manufactured technology parameters yet. Therefore same circuit analog structure is redesigned using TSMC 0.35 µm model parameters [5]. Design results are shown in Fig.4.14.

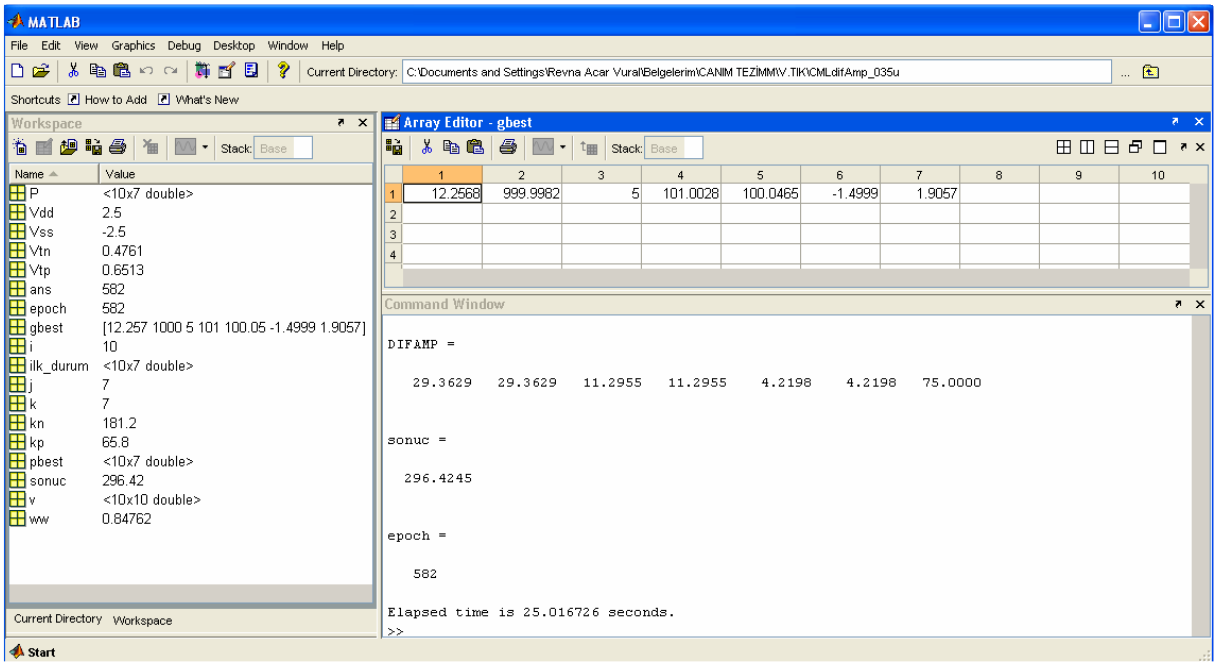

Fig. 4.14 PSO Based Differential Amplifier with Current Mirror Load Design Results-II

Target value of CF is aimed to be smaller than 300 µm^2. A total MOS transistor area of 296 µm^2 with each MOSFET width ($W_{n,p}$), I_{bias} and C_L is obtained within 582 epochs in 25.02 s. Exact design criteria values are obtained as global best (g_{best}) particle vector given below.

$$g_{best}=[12.257\ 1000\ 5\ 101\ 100.05\ -1.49\ 1.91] \quad (4.38)$$

where vector elements are SR (V/µs), P_{diss} (µW), C_L (pF), f_{-3dB} (kHz), A_v (V/V) $V_{ic(min)}$ (V) and $V_{ic(max)}$ (V) values obtained by PSO-based method respectively.

Channel lengths were set as $L_{1..4}=3.5\mu m$ and $L_{5..6}=1.4\mu m$. Exact values of design parameters obtained from PSO based design are given below:

- $W1=W2=29.3629\mu m$
- $W3=W4=11.2955\mu m$
- $W5=W6=4.2198\mu m$
- $C_L=5\ pF$
- $I_{bias}=75\mu A$

Differential amplifier with current mirror load is redesigned using the resulting design parameters in SPICE simulator in order to validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are not coherent with PSO based design results depending on the difference of circuit equation sets utilized in SPICE and theoretical calculations. Current driving capability of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor cannot be expected to have the same maximum charge/discharge current as a long-channel transistor

with the same (W/L) ratio (Kang and Leblebici, 2005). Therefore, in order to make PSO-based design more feasible, I_{bias} is increased from 75 µA to 125 µA while saving MOS sizes obtained with PSO-based design. Resulting SPICE simulation graphs are provided from Fig. 4.15 to Fig. 4.22.

Design specifications for differential amplifier with current mirror load, DARWIN synthesizer (Kruiskamp and Leenaerts, 1995) results and SPICE simulations results of PSO based design with bias current improvement are tabulated in first, second and third column of Table 4.14, respectively.

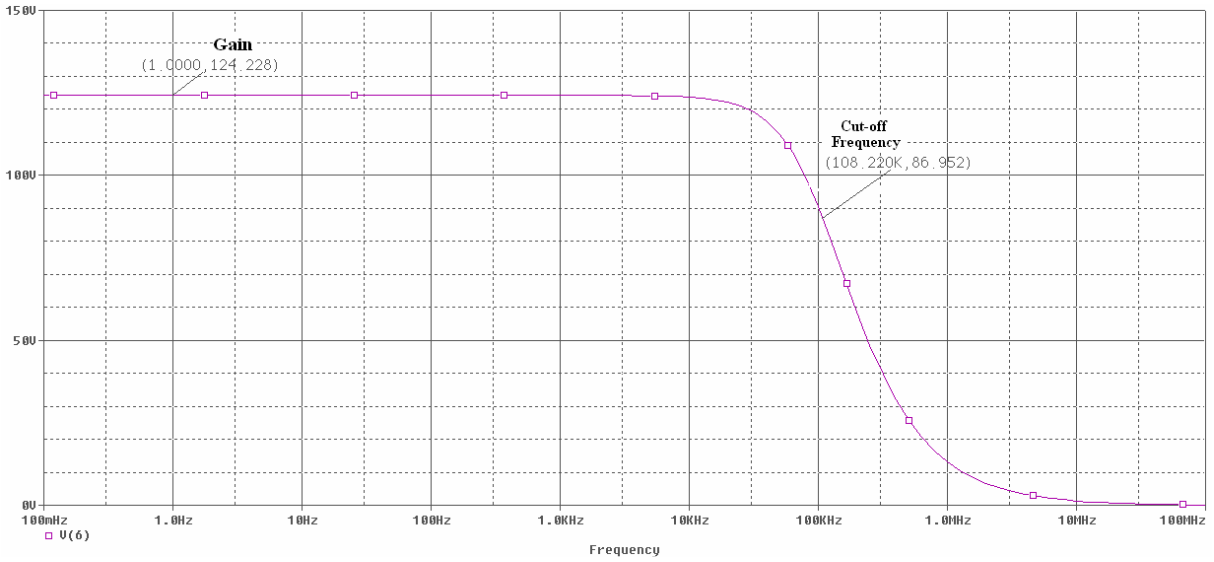

Fig. 4.15 Frequency response of PSO based Differential Amplifier with Current Mirror Load

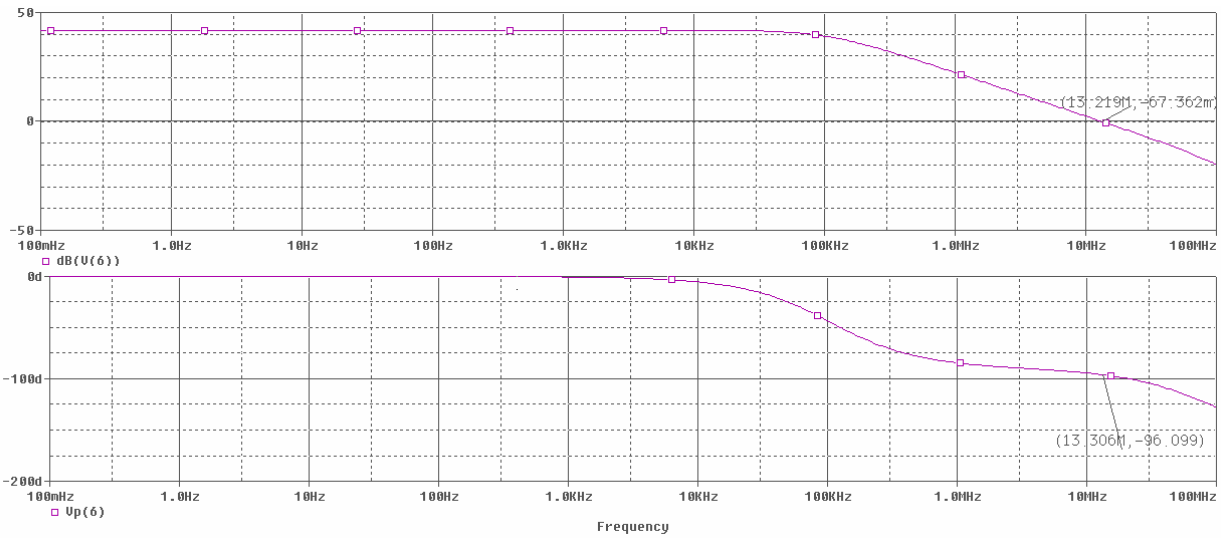

Fig. 4.16 Phase margin of PSO based Differential Amplifier with Current Mirror Load

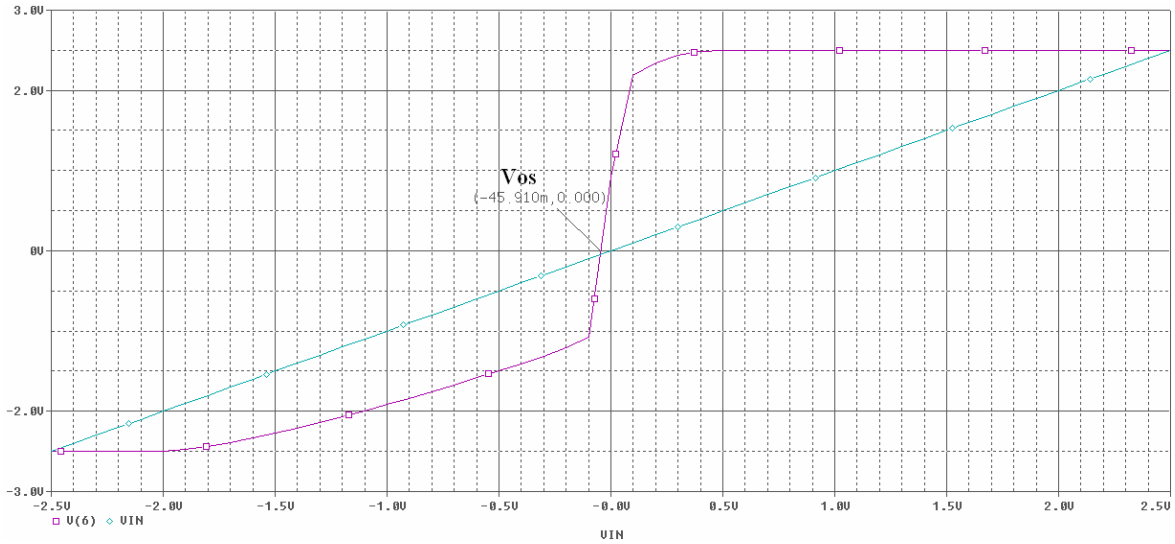

Fig. 4.17 Offset voltage of PSO based Differential Amplifier with Current Mirror Load

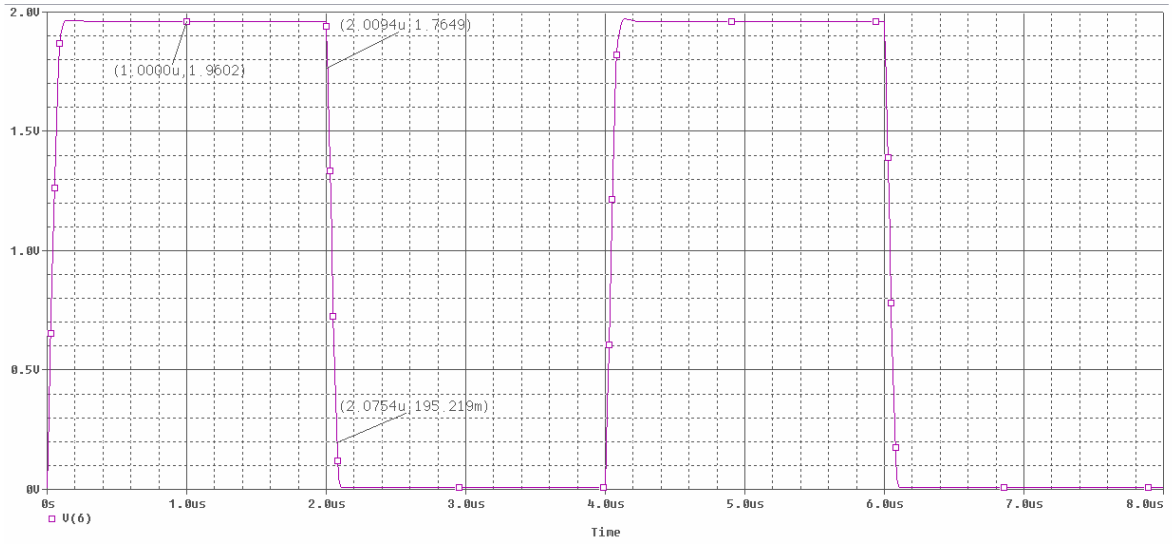

Fig. 4.18 Slew rate of PSO based Differential Amplifier with Current Mirror Load

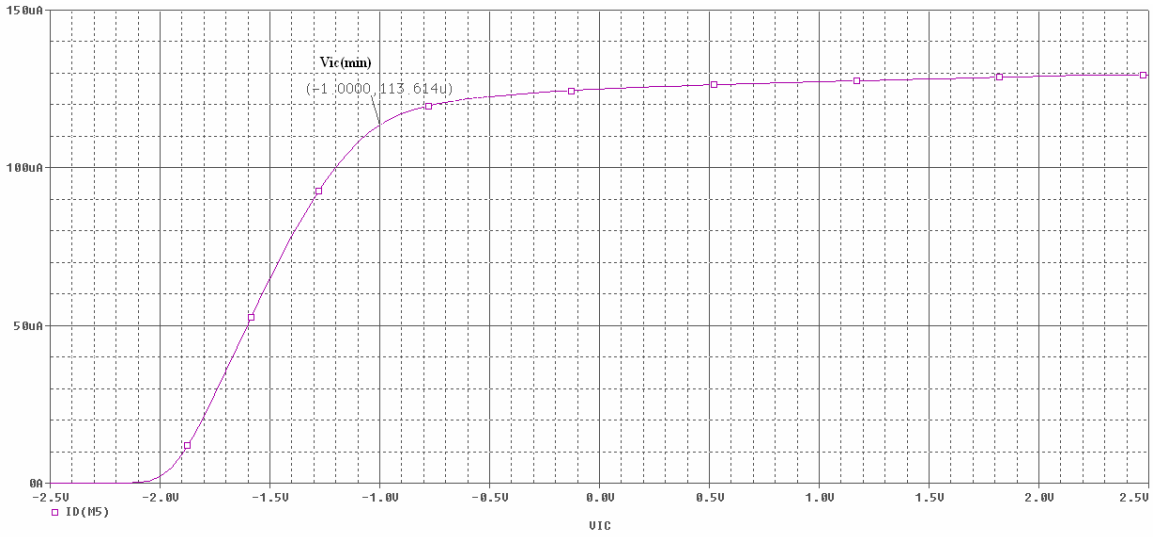

Fig. 4.19 ICMR-*Vic(min*) of PSO based Differential Amplifier with Current Mirror Load

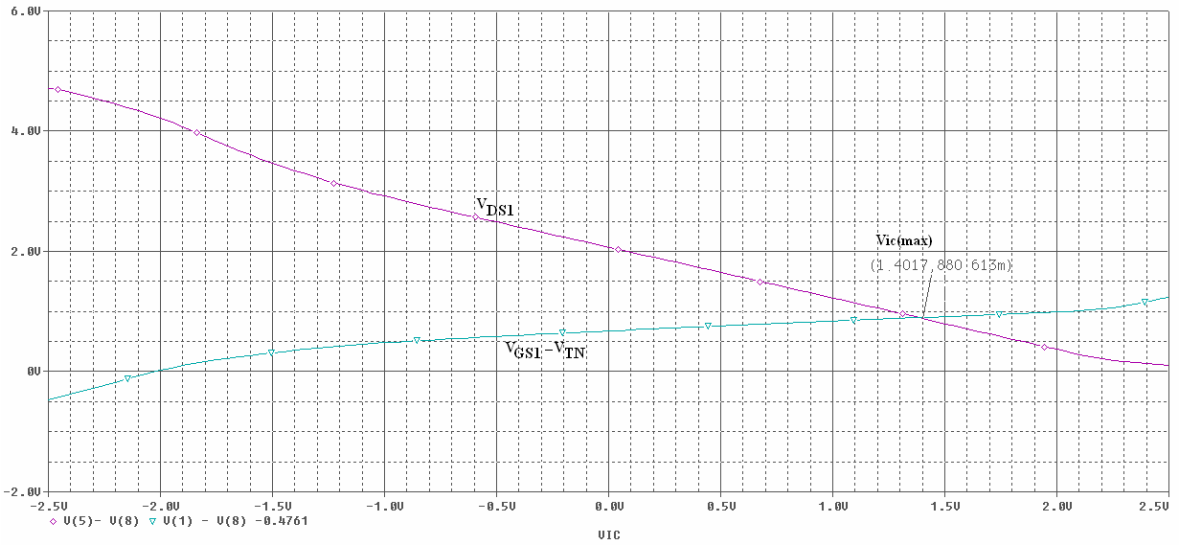

Fig. 4.20 ICMR-V_{ic}*(max)* of PSO based Differential Amplifier with Current Mirror Load

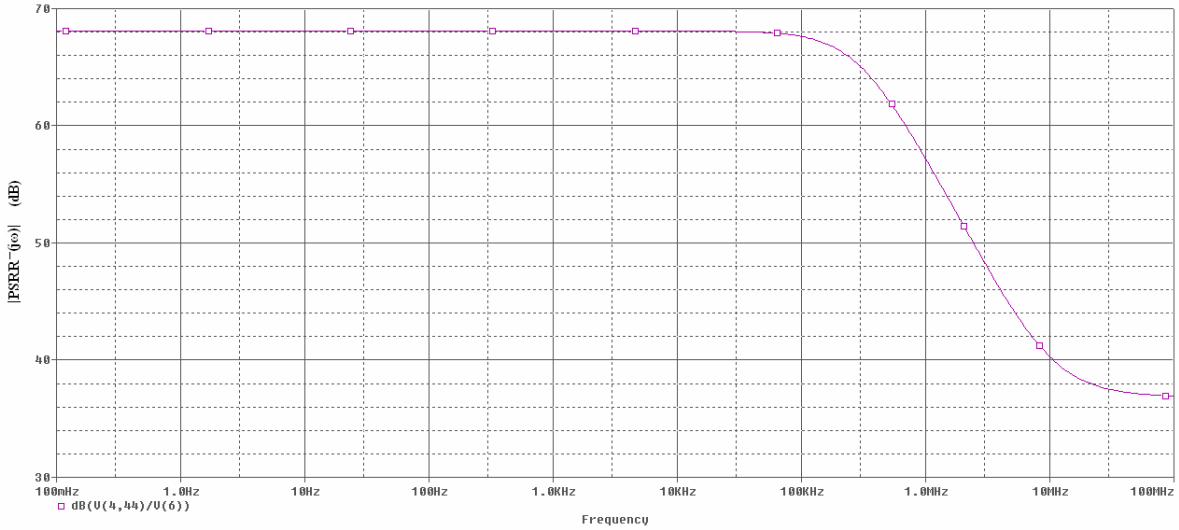

Fig. 4.21 PSRR$^-$ of PSO based Differential Amplifier with Current Mirror Load

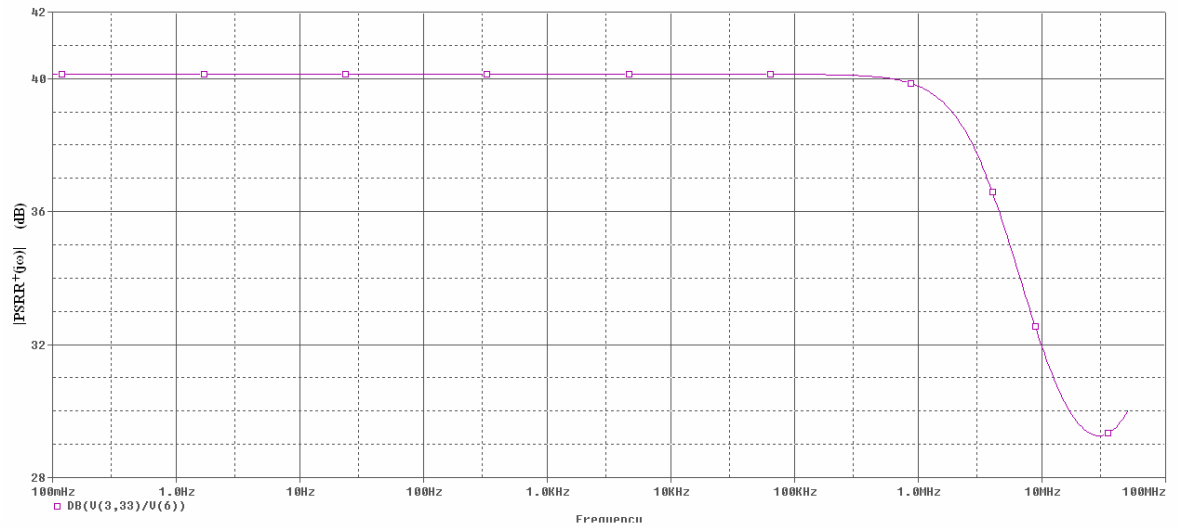

Fig.4.22 PSRR$^+$ of PSO based Differential Amplifier with Current Mirror Load

Table 4.15 depicts the design parameters obtained within each method. Consequently, the synthesized differential amplifier with current mirror load meets all specifications according to the estimations of PSO-based design with bias current improvement. PSO-based design also minimizes the total MOS area with respect to the DARWIN tool (Kruiskamp and Leenaerts, 1995).

Table 4.15 Comparison of DARWIN and PSO with bias current improvement by means of design specs

Differential Amplifier with Current Mirror Load Design Criteria	**Spec**	**DARWIN**	**PSO with bias current improvement (SPICE)**
Output Capacitance (pF)	≥5	2	5
Slew Rate (V/μs)	≥ 10	3.2	23.76
Power Dissipation (μW)	≤ 2000	31	1260
Phase Margin (^{o})	>45	72	84
Cut-off Frequency (kHz)	≥100	7.17	108.22
Gain (dB)	> 40	60	42
Vic_{min} (V)	≥-1.5	-1.3	-1
Vic_{max} (V)	≤2	1.9	1.4
CMRR (dB)	> 40	100	84
$PSRR^+$ (dB)	>40	78	41
$PSRR^-$ (dB)	>40	136	68
V_{os} (mV)	<50	41	45.91
Output Resistance (kΩ)	>200	10000	334
Total Area (m^2)	**<3x10^{-10}**	**65x10^{-10}**	**2.96x10^{-10}**

Table 4.16 Comparison of DARWIN and PSO with bias current improvement by means of design parameters

Differential Amplifier with Current Mirror Load Design Parameters	**DARWIN**	**PSO with bias current improvement**
I_{bias} *(μA)*	2	125
W1/L1, W2/L2 (μ/μ)	240/13.2	29.3629/3.5
W3/L3, W4/L4 (μ/μ)	7.3/7.7	11.2955/3.5
W5/L5 (μ/μ)	4.6/2.4	4.2198/1.4
W6/L6 (μ/μ)	2.4/2.4	4.2198/1.4
C_L *(pF)*	2	5

4.2.3.2 Simulation Results of PSO based Two-Stage Operational Amplifier Design

The starting point of design consists of two types of information. First type of information such as the technology and the power supply is set by the designer. The other type of information is the design criteria. The range of each criteria and design parameter, power supply values and technology information is set as an input to PSO algorithm (Table 4.17) and PSO algorithm should obtain the solution set that consists the exact values of design parameters (C_c,, C_L, $(W/L)_{1..8}$) and design criteria (f_t, $V_{IC(max)}$, $V_{IC(min)}$, SR, P_{diss}, A_v) for given ranges. The design is implemented with the relationships that describe design specifications to solve for DC currents and W/L values of all MOS transistors. The appropriate relationships were provided in the previous subsections. Simulations are performed with TSMC 0.35 µm technology parameters.

Table 4.17 PSO Inputs and Outputs for PSO based Two-stage Operational Amplifier

Components in CF	Information	Input/Output for PSO
V_{DD}, V_{SS} V_{tn}, V_{tp} $\mu_n C_{ox}$, $\mu_p C_{ox}$ λ_n, λ_p	➢ Set by human designer ➢ Fabrication technology dependent	INPUT
C_L,,C_c $(W/L)_{1..8}$ f_t $V_{IC(max)}$, $V_{IC(min)}$ SR P_{diss} A_v	➢ PSO would find exact results for the given ranges	OUTPUT

PSO is utilized for a two-stage operational amplifier having design specifications of SR≥10V/µs (C_L=10pF), f_t≥3MHz (C_L=10pF), A_v>1000 V/V, -1.5V≤ ICMR≤2V, P_{diss}≤2.5mW with PSO inputs of V_{DD}= -V_{SS}=2.5V, V_{tn}= 0.4761V, V_{tp}= -0.6513V, K'_n= 181.2µA/V^2, K'_p= 65.8µA/V^2, λ_n=0.04 V^{-1}, λ_p=0.05 V^{-1}. Constraints for design parameters are set as C_L>=10 pF, 100≥$(W/L)_{1..8}$≥2. In order to minimize the channel modulation effect, MOSFET length values are chosen as $L_{1..8}$=2 µm.

Initial population matrix size was 10x7 where row number of 10 indicates the number of particles in the population and column number of 7 is the dimension of particle vector. Particle vector structure is expressed in (4.39) where SR is the slew rate (V/µm), C_L is the output capacitance (pF), A_v is the gain (V/V), f_t is the unity gain bandwidth (MHz), P_{diss} is the

power dissipation (µW), V_{icmin} (V) and V_{icmax} (V) are the lower and upper limits of ICMR, respectively.

$$x = [SR, C_L, A_v, f_t, V_{icmin}, V_{icmax} P_{diss}] \quad (4.39)$$

Velocity update parameters c_1, c_2 and w were 1.7, 1.7 and 0.99, respectively. The algorithm runs for upper limit of 100 iterations. CF is defined as the total area that MOS transistors occupy and given in (4.37).

PSO based two-stage operational amplifier design results are provided in Fig. 4.23. Target value of CF is aimed to be smaller than 300 µm^2. PSO-based design method resulted in a total MOS transistor area of 235.8373 µm^2 along with exact values of design specification and design parameters (($W_{n,p}$), I_{bias}, C_c, C_L). Design process concluded after 100 epochs with a total execution time of 8.6 s.

$$g_{best}=[13.4653\ 10\ 39682\ 5.531\ -0.726\ 1.699\ 2318] \quad (4.40)$$

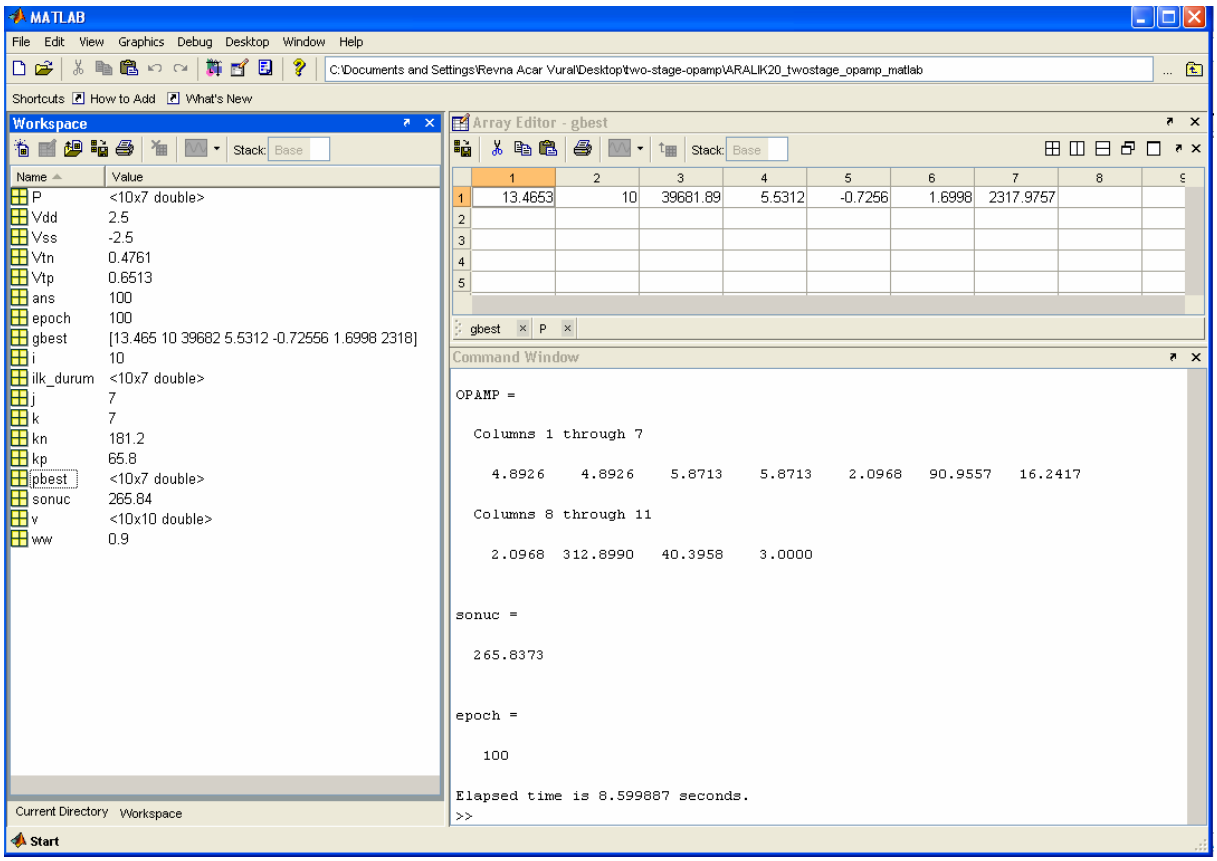

Fig. 4.23 PSO Based Two-stage Operational Amplifier Design Results

Channel lengths were set as $L_{1..8}=2\ \mu m$. Exact values of design parameters obtained from PSO based design are given below:

- $W1=W2=4.8926\ \mu m$
- $W3=W4=5.8713\ \mu m$
- $W5=W8=2.0968\ \mu m$
- $W6=90.9557\ \mu m$
- $W7=16.2417\ \mu m$
- $C_L=10\ pF$
- $C_c=3\ pF$
- $I_{bias}=40.39\ \mu A$

Two-stage operational amplifier is redesigned using the resulting design parameters in SPICE simulator in order to validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are very coherent with PSO based design results except gain value. This is mostly due to the more complex channel modulation effect equations in SPICE simulations which have been discarded in theoretical calculations. Nevertheless, PSO-based design is satisfying all design specifications, which is demonstrated from Fig. 4.24 to Fig. 4.31.

Design specifications for two-stage operational amplifier, convex optimization (Hershenson, 2001) results, theoretical calculation and SPICE simulations results of PSO based design are tabulated in first, second, third and fourth column of Table 4.18, respectively.

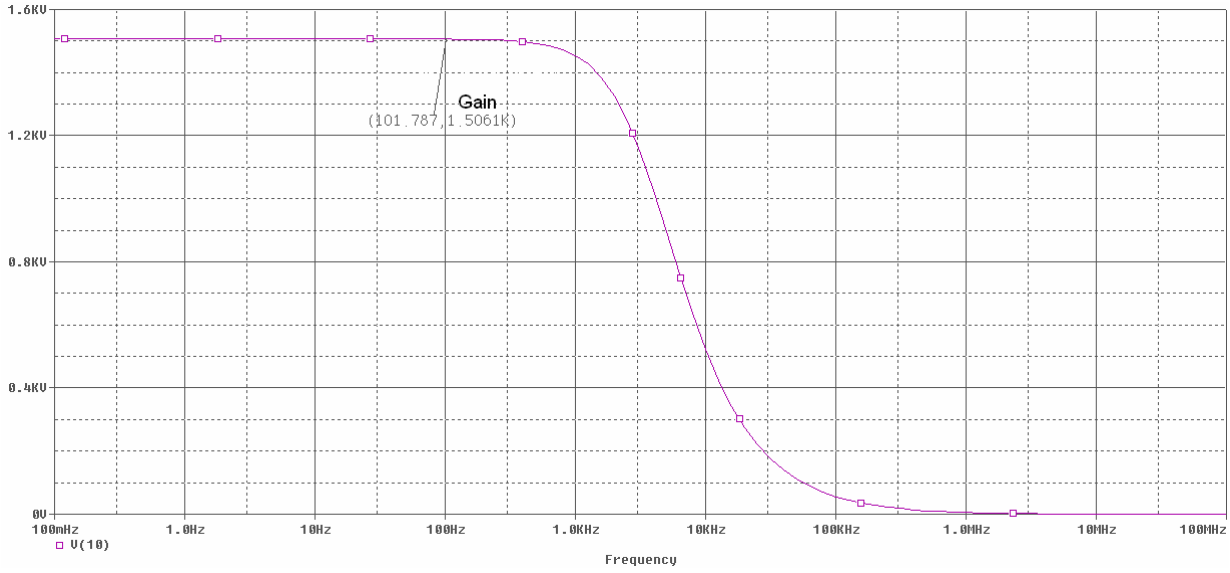

Fig.4.24 Frequency response of PSO based Two-stage Operational Amplifier

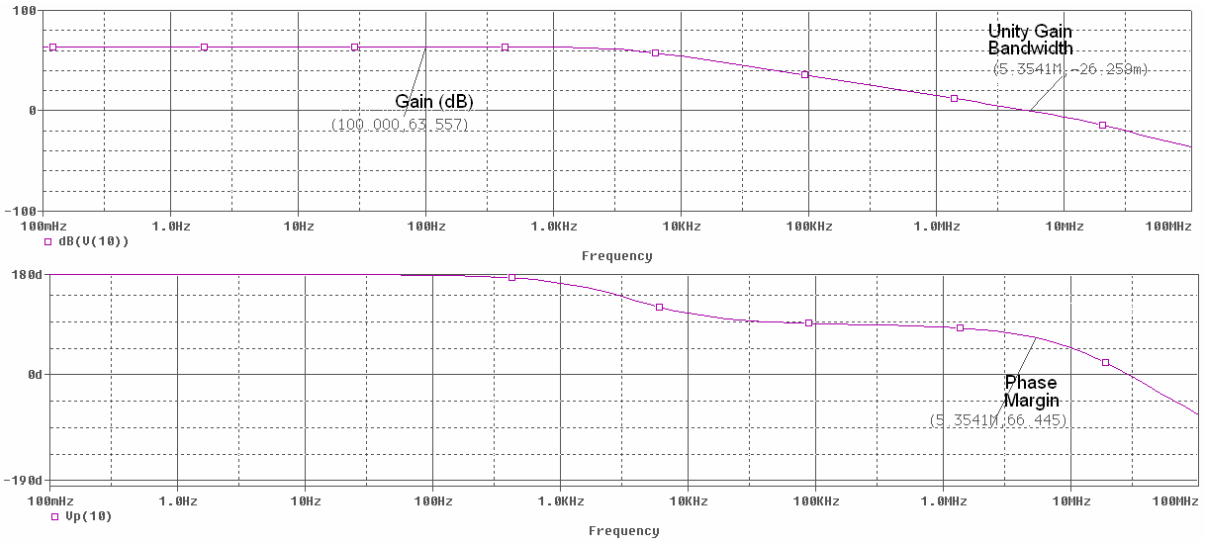

Fig. 4.25 Phase margin of PSO based Two-stage Operational Amplifier

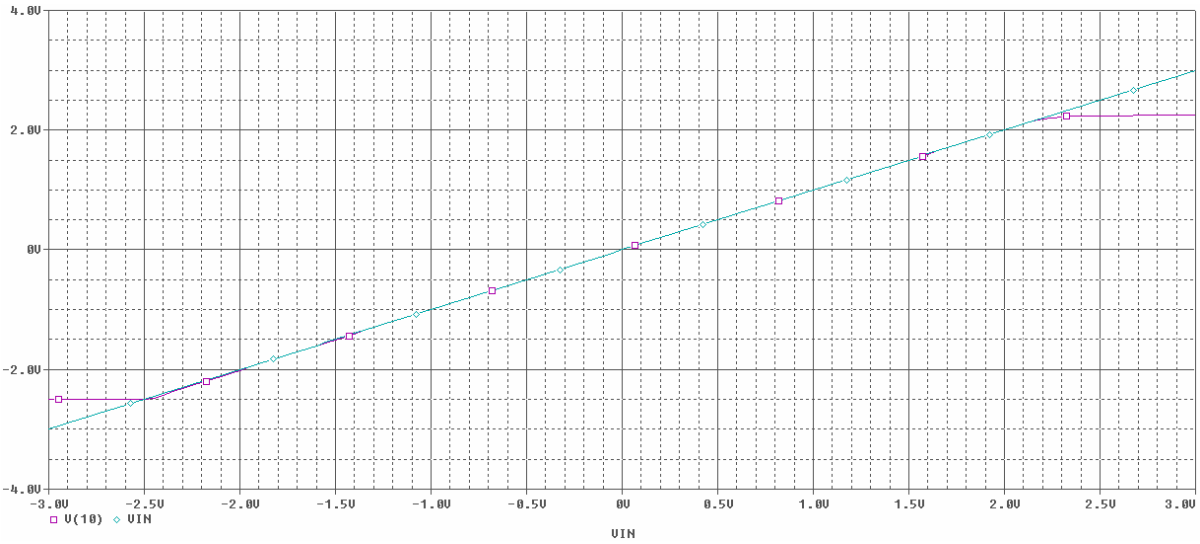

Fig. 4.26 Output voltage range of PSO based Two-stage Operational Amplifier

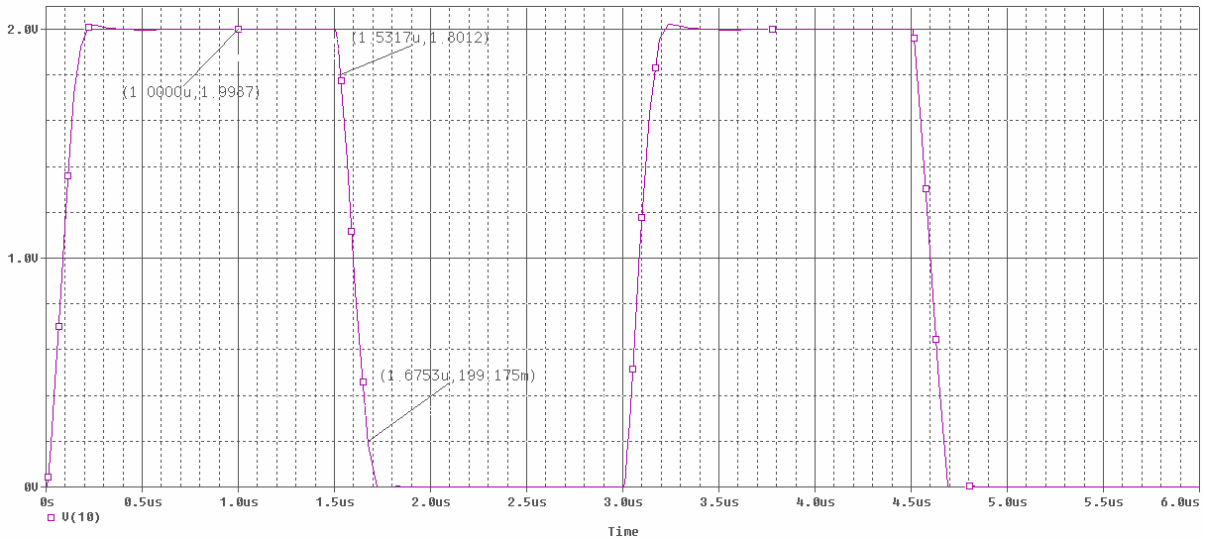

Fig. 4.27 Slew rate of PSO based Two-stage Operational Amplifier

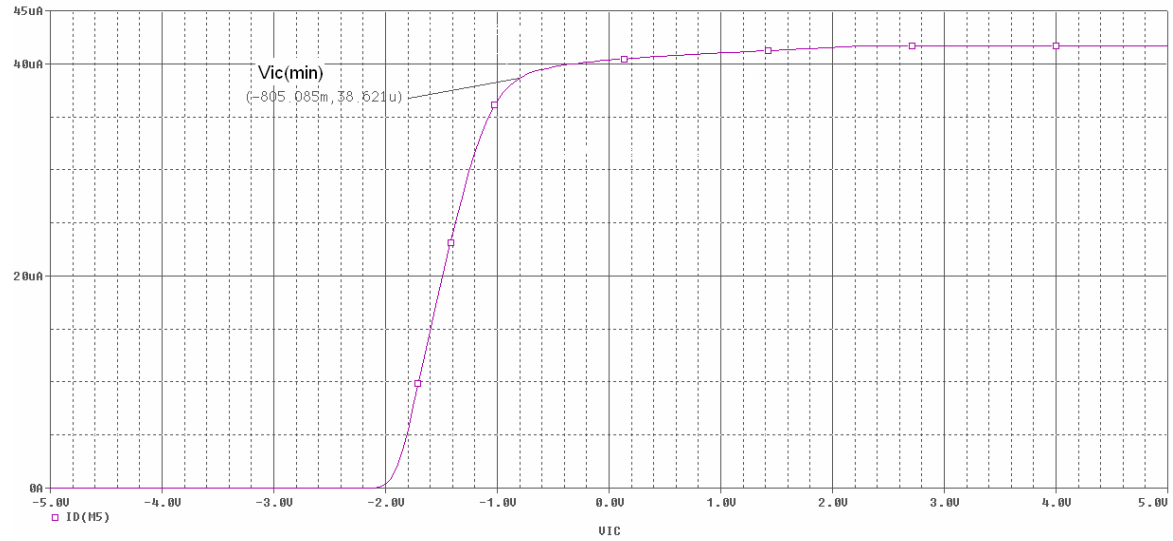

Fig.4.28 ICMR -V_{ic}*(min)* of PSO based Two-stage Operational Amplifier

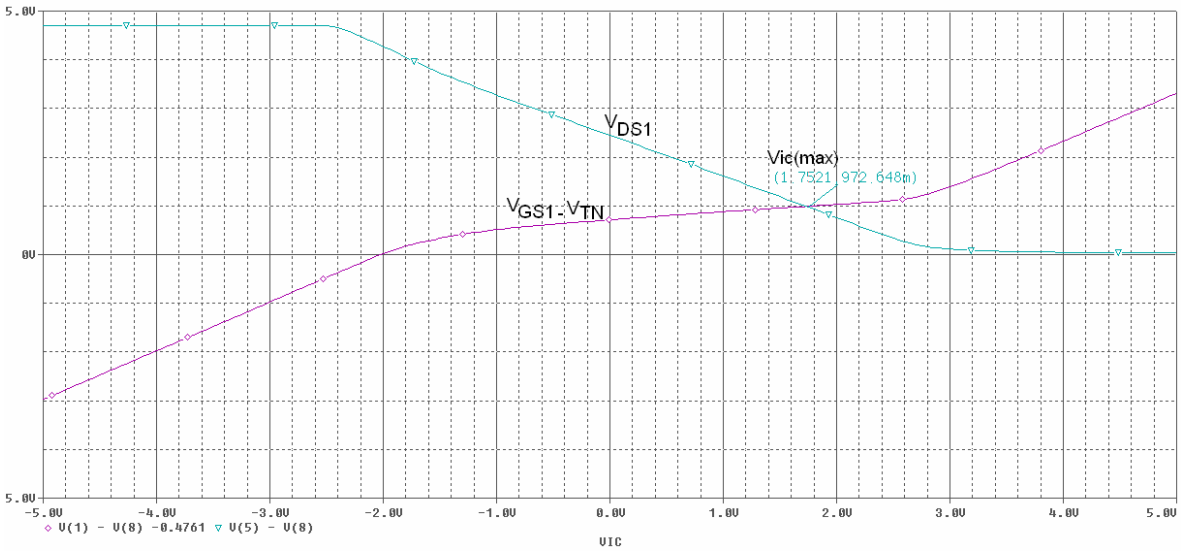

Fig.4.29 ICMR - V_{ic}*(max)* of PSO based Two-stage Operational Amplifier

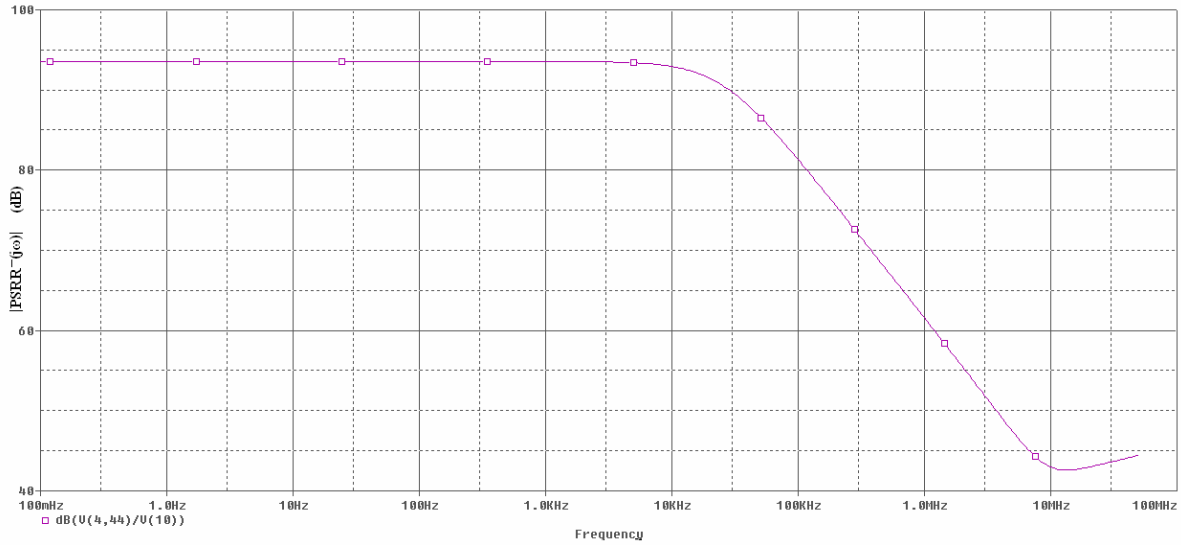

Fig. 4.30 PSRR$^-$ of PSO based Two-stage Operational Amplifier

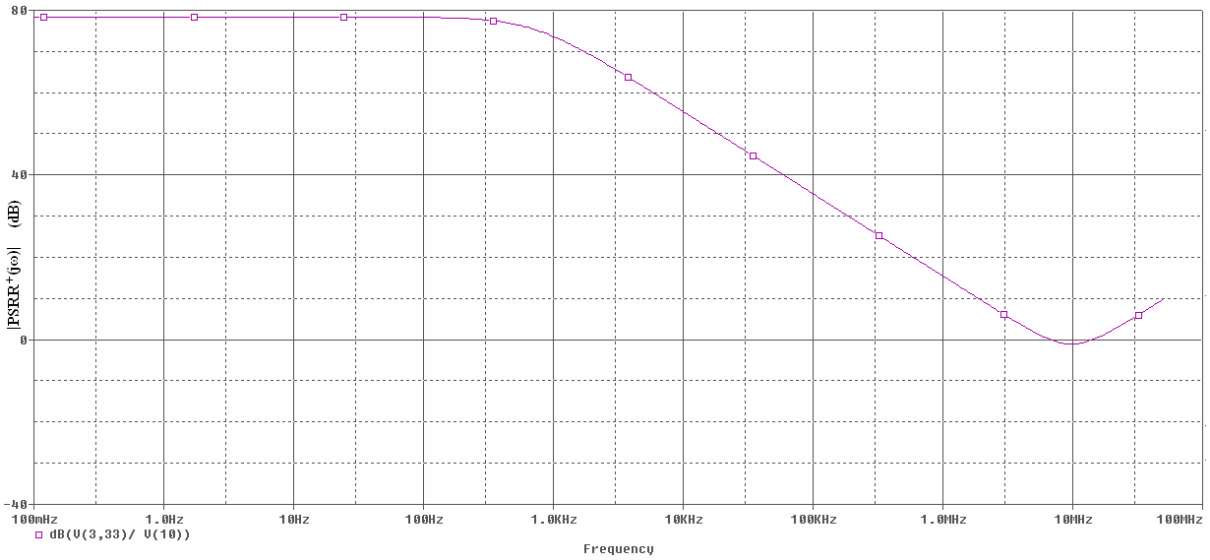

Fig.4.31 PSRR^{+} of PSO based Two-stage Operational Amplifier

Table 4.18 Comparison of Convex Optimization and PSO based design method by means of design specifications

Two-stage Operational Amplifier Design Criteria	**Spec**	**Convex Optimization**	**PSO (Theoretical)**	**PSO (SPICE)**
Output Capacitance (pF)	≥10	3	10	10
Slew Rate (V/μs)	≥ 10	88	13.465	11.15
Power Dissipation (μW)	≤ 2500	5000	2320	2370
Phase Margin (^{o})	>45	60	-	66.45
Unity Gain Bandwidth (MHz)	≥3	86	5.531	5.3541
Gain (dB)	> 60	89.2	91	63.5
Vic_{min} (V)	≥-1.5	-	-0.726	-0.8
Vic_{max} (V)	≤2	-	1.699	1.75
CMRR (dB)	> 60	92.5	-	83.74
PSRR^{+} (dB)	>70	116	-	78.36
PSRR^{-} (dB)	>70	98.4	-	93.56
Output Resistance (kΩ)	>200	-	-	751
Total Area (m^2)	**<3x10^{-10}**	**82x10^{-10}**	**2.65x10^{-10}**	**2.65x10^{-10}**

Table 4.19 depicts the design parameters obtained within each method. Consequently, the synthesized operational amplifier with current mirror load meets all specifications according to the estimations of PSO-based design. PSO-based design also minimizes the total MOS area with respect to the convex optimization.

Table 4.19 Comparison of Convex Optimization and PSO based design method by means of design parameters

Two-Stage Operational Amplifier Design Parameters	**Convex optimization**	**PSO-based method**
$I_{bias}\,(\mu A)$	10	40.39
W1/L1, W2/L2 (µm/µm)	232.8/0.8	4.8926/2
W3/L3, W4/L4 (µm/µm)	143.6/0.8	5.8713/2
W5/L5 (µm/µm)	64.6/0.8	2.0968/2
W6/L6 (µm/µm)	588.8/0.8	90.9557/2
W7/L7 (µm/µm)	132.6/0.8	16.2417/2
W8/L8 (µm/µm)	2/0.8	2.0968/2
$C_L\,(pF)$	3	10
$C_c\,(pF)$	3.5	3

4.3 Summary

Particle swarm optimization has been utilized for both digital and analog integrated circuit design cases. In order to investigate PSO performance for digital circuit design, switching characterization of an inverter is considered and three different design cases concerning transient characteristics were carried out. First, PSO algorithm was configured in order to estimate output voltage fall time depending on the design parameters and fabrication technology parameters. 10 different ranges of design parameters and design criterion were specified for PSO algorithm. Synthesized results were compatible with theoretical calculations. Second case deals with an inverter design having a symmetrical output voltage. For this purpose, error between fall time and rise time of output voltage was minimized by PSO algorithm for 7 inverter design having different ranges for design parameters and design criteria. Differences of fall and rise times of nine of the examples were below specified target error. For the all specified ranges, PSO algorithm found a minimum error value smaller than the specified error. The last case was the most complicated among the others. A symmetrical output voltage was also of concern along with the minimization of propagation delay times. CF, namely error, was defined by the sum of difference between output voltage delay times and propagation delay times. Synthesized results showed that PSO obtained design parameters and design criterion values in compatible with specified ranges. CF of nine of the design examples was below the target error and specified ranges of the remaining example did not allow PSO to find a minimum CF smaller than the target error. This example was investigated on purpose, in order to state the importance of range selection.

For the first design case, it was assumed that PMOS and NMOS dimension ratios were equal. Considering following examples, different dimension ratios of PMOS and NMOS has been obtained by PSO and synthesized results showed that symmetrical delay times could not be obtained with a dimension ratio of PMOS to NMOS smaller than five. Decreasing PMOS dimension leads to a smaller area on chip layout but worse symmetry at the output voltage.

For comparison, using PSO results of output capacitance value and MOSFET dimensions, inverters were redesigned in SPICE environment for each case, considering PSO results as SPICE inputs. The difference between SPICE results and PSO-based design results arise from the fact that SPICE computes using more complex circuit equation sets than used in theoretical calculations. PSO uses delay expressions which were derived using simple current-voltage relationships originally developed for long-channel transistors. These current expressions based on the gradual channel approximation can still be used for sub-micron MOS transistors with proper parameter adjustments; therefore delay analysis used here remains largely valid for small-geometry devices as well. Yet it should be noted that the current driving capability of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor cannot be expected to have the same maximum charge/discharge current as a long-channel transistor with the same (W/L) ratio. Thus, SPICE results in greater delay times compared to PSO-based inverter design.

Regarding analog ICs, design scheme is perceived as less systematic and more heuristic and knowledge-intensive in nature than digital IC design. The variety of circuit schematics and the number of conflicting requirements and corresponding diversity of device sizes are also much larger. Analog IC design mainly consists of topology choice, sizing task and the generation of layout. The problem considered here is the optimal selection of transistor dimensions, which is only a part of a complete analog circuit CAD tool. Actually, analog sizing is a constructive procedure that aims at mapping the circuit specifications where the performance metrics of the circuit, such as gain, power dissipation, occupied area, etc. have to be formulated in terms of the design parameters. Following, these design parameters such as device sizes and bias currents should be adjusted under multiple design objectives and constraints. The many degrees of freedom in parameter space as well as the need for repeated circuit performance evaluation made this a lengthy and tedious process. Here, particular specifications for specified topologies of a differential amplifier with current mirror load and a two-stage operational amplifier are aimed to be met by adjusting design parameters such as device sizes and bias currents with PSO algorithm.

Design equations of each analog circuit are utilized for multi-objective cost function of PSO algorithm, since numerous design specs are of concern. Following, SPICE simulations are carried out in order to validate the feasibility of synthesized circuits. Considering differential amplifier with current mirror load, design performance of PSO-based method is first compared with classical method (Allen and Holberg, 2002). Having satisfactory results, PSO-based design is utilized for TSMC 0.35 µm technology parameters and design process is concluded in 25 s. Using the resulting design parameters, synthesized circuit is simulated with SPICE in order to validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are not coherent with PSO based design results depending on the difference of circuit equation sets utilized in SPICE and theoretical calculations. Current driving capability of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor can not be expected to have the same maximum charge/discharge current as a long-channel transistor with the same (W/L) ratio. Therefore, in order to make PSO-based design more feasible bias current is increased while saving MOS sizes obtained with PSO-based design. This improvement not only satisfied all the constraints but also minimized the total MOS area with respect to DARWIN (Kruiskamp and Leenaerts, 1995) tool.

Two-stage operational amplifier design involves more design parameter adjusting than the former design task while each analog design task case has the same dimension of particle vector size. PSO-based design for two-stage amplifier is concluded in 8.6 seconds after 100 iterations. Resulting design parameters are utilized for redesign in SPICE simulator in order to validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are very coherent with PSO based design results except gain value. This is mostly due to the more complex channel modulation effect equations in SPICE simulations which have been discarded in theoretical calculations. Nevertheless, PSO-based two-stage operational amplifier design satisfied all the design specifications as well as minimized total design area with respect to convex optimization (Hershenson et al., 2001).

Considering both design cases, PSO proved its efficiency on analog IC design with high accuracy and short computation time. Analog IC design problem can be considered as a multi-objective constrained problem and as a global optimization tool, PSO is very successful with handling those conflicting objectives as long as design specification equations are well defined and introduced to PSO.

5. ANN FOR TECHNOLOGY INDEPENDENT INTEGRATED CIRCUIT DESIGN

This section introduces a technology independent Artificial Neural Network (ANN) modeling technique for current steering CMOS Digital-to-Analog Converter (DAC) circuit. Previous sections deals with EA based design of basic circuit structures with various design constraints and particular technology parameters. Here, the aim is to predict the transistor sizes of DAC circuit that meets the design constraints, with minimum user effort and design knowledge for a newer semiconductor technology, using ANN co-simulated with Cadence Spectre Analog Environment. In contrast to other modeling researches, the transistor sizes of DAC circuitry are predicted for new technology designs. The design constraints are specified and numerous simulations are carried out for different channel length and channel width of all transistors that meet the constraints using Cadence. The simulations are performed using three different technology BSIM3v3 SPICE parameters that are ON SEMI 1.5µm, ON SEMI 0.5µm and TSMC 0.35µm [5] technology parameters. Eventually, a large database is developed consisting of transistor sizes (*W, L*) of DAC designs for three different technologies and static characteristics of DAC designs as obtained from simulation results. The key point is that the ANN was trained with the dataset including the simulation results of ON-SEMI 1.5µm and 0.5µm technology parameters and the test data is constituted with only the simulation results of TSMC 0.35µm technology parameters which has not been introduced to the ANN structure for training beforehand. The ANN provides the channel lengths and widths of all transistors for a newer technology when the designer sets the numeric values of DAC static output specifications as Differential Nonlinearity (DNL) error, Integral Nonlinearity (INL) error, monotonicity and gain error as the inputs of the network.

5.1 Current Steering Type DAC Structure

DAC is one of the key components in mixed-signal systems that converts a digital code into an analog signal. The output is proportional to the value of the digital code provided to its inputs. The output of a typical DAC is an analog signal, which is usually voltage or current (Wang et al, 2001; Zumbahlen, 2008). Current-mode signal processing circuits have demonstrated many advantages over voltage-mode circuits, including increased bandwidth, higher dynamic range and easy implementation. In addition, current-mode processing often leads to smaller circuit size and less power consumption (Wen and Lee, 2001). The voltage output is an advantage of voltage mode, as is the constant output impedance, which eases the stabilization of any amplifier connected to the output node (Kester, 2005).

A wide variety of DAC architecture exists, each with unique characteristics and different limitations. In this study, designed 4 bit DAC circuitry is based on the familiar R-2R resistive current divider (Hoeschele, 1994). PMOS transistors are used in place of polysilicon resistors to save chip area (Kier et al, 2004). The same functionality has been achieved using PMOS-only implementation shown in Fig. 5.1.

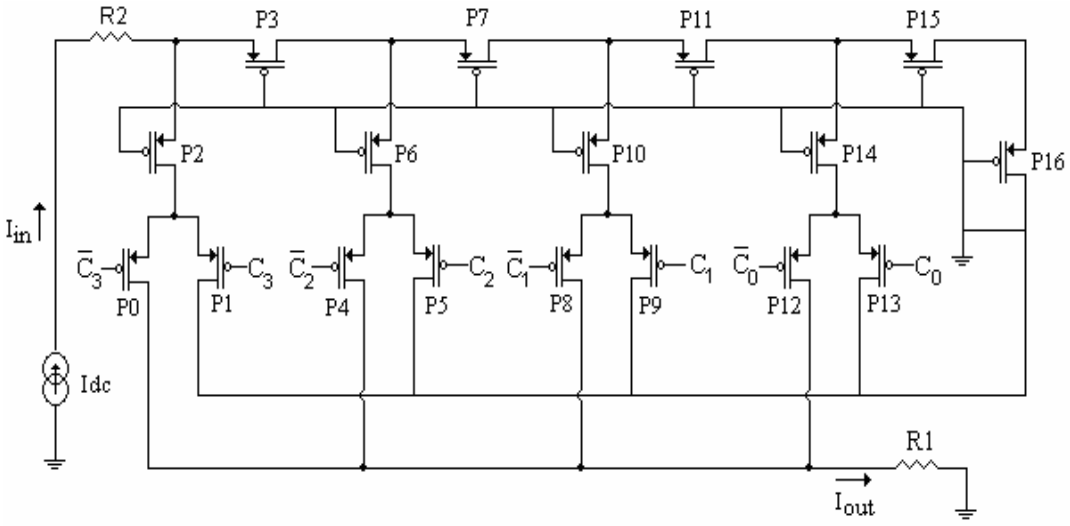

Fig. 5.1 Current steering 4-bit CMOS DAC circuitry (Hoeschele, 1994)

The pseudo-resistance of each MOS transistor (Vittoz and Arreguit, 1993) which is determined only by its width-to-length ratio is denoted as R in this structure. P3, P7, P11, P15 transistors act as the R part of designed R-2R ladder DAC circuitry. Series combination of the switching transistors (P0,P1,P4,P5,P8,P9,P12,P13) and the pass transistor in each brunch (P2,P6,P10,P14) perform as 2R portion of resistive counterpart. The switching transistors are driven with complementary pulse signals ($\overline{C}, C$). The incoming current, I_{in}, is divided into brunches based on the current division principles. In this work, the input current source I_{dc} is set to 100μA. The value of input current source determines the full scale output current of the circuitry. The current in each branch is switched to ground or onto the output node. The output of the circuit is a current, I_{out}, which is some fraction of reference current (Baker et al., 1997) as given in (5.1). *D* refers to digital input word with *N* bits wide.

$$I_{out} = \frac{D}{2^N} I_{in} \tag{5.1}$$

5.2 Static Characterization of DAC

Non ideal circuit elements of DAC cause the analog increments to differ from ideal values. Outputs of DAC become distorted and noise to be added to the signal because of nonlinearities. A number of different measurements characterize DAC and its performance. The transfer characteristics of DAC can be analyzed through its static and dynamic parameters. In general, static specification parameters (SSP) include resolution, offset error, gain error, monotonicity, integral nonlinearity (*INL*), and differential nonlinearity (*DNL*). Total harmonic distortion, signal to noise ratio, gain band-width and slew rate are usually measured for examining the dynamic parameters. These SSPs must be measured and compared against requirements during the performance evaluation of DAC circuits. The importance of a particular SSP and its limit depend upon the application. In some applications only a few SSPs are needed to guarantee performance (Hoeschele,1994). In this study, SSPs including *DNL*, *INL*, gain error, monotonicity were measured for performance evaluation of the designed 4-bit DAC. However, for simplicity, information about static characterization is explained considering a 3-bit DAC.

- *Differential Nonlinearity (DNL)*: The step size in the non-ideal data converter deviates from the ideal size and this error is called the *differential nonlinearity* (*DNL*) (Hoeschele, 1994). *DNL* is a measure of the worst variation in analog quantizing step size from the ideal 1-LSB step over the full range of the DAC. The Least Significant Bit (LSB) defines the smallest possible change in the analog output voltage. Deviations are usually specified in terms of fractional part of 1 LSB. *DNL* error for 3-bit DAC is illustrated in Fig. 5.2.

$$DNL = [(V_{D+1} - V_D)/V_{LSB\text{-}IDEAL} - 1] \tag{5.2}$$

where $0 < D < 2N - 2$ and V_D represents the actual output value of DAC corresponded to D^{th} digital input code.

The *DNL* is usually expressed in units of LSBs rather than absolute units. If the *DNL* for a DAC is less than -1 LSBs, then the DAC is said to be nonmonotonic, which means that the analog output voltage does not always increase as the digital input code is incremented (Baker et al., 1997).

- *Integral Nonlinearity (INL)*: Integral Nonlinearity (INL) is a measure of the largest deviation of the DACs transfer function from a straight line. There are various methods for

determining INL. It is often specified with respect to a straight line drawn between the actual 0 (first value) and full-scale endpoint (last value) (Hoeschele, 1994). The other method compares the output values to the ideal reference line, regardless of the first and last output values. This measurement of error includes not only linearity errors but also the offset and gain error. Here, deviations of the output values to the ideal reference line are measured in order to determine INL. Usually, gain error and offset are determined as separate specifications (Baker et al, 1997).

$$INL = [(V_D - V_{Dideal})/V_{LSB\text{-}IDEAL}] \tag{5.3}$$

where $0 < D < 2N - 1$ and V_{Dideal} is the ideal output value of DAC corresponded to D^{th} digital input code.

INL is represented in units of LSBs as similar with *DNL*. Generally, a DAC with N-bit resolution will have less than ±1/2 LSB of *DNL* and *INL*. *INL* error for 3-bit DAC is illustrated in concurrence with *DNL* error in Fig. 5.2.

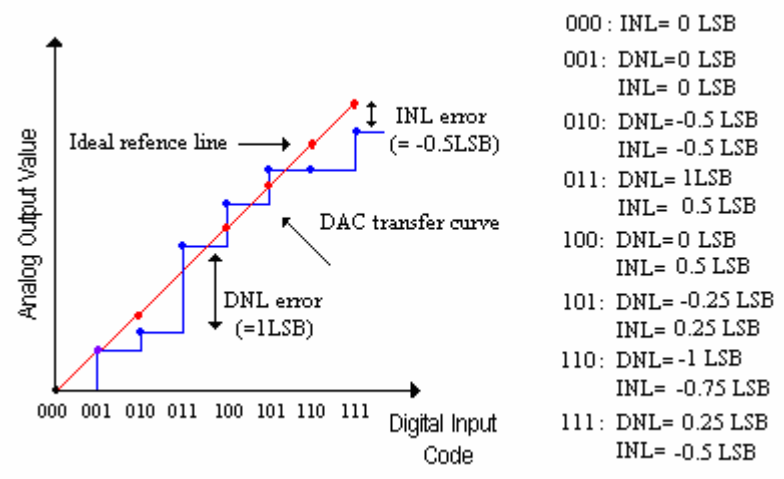

Fig. 5.2 DNL and INL errors for 3-bit DAC

- Gain Error: Gain error is the error in slope of the DAC transfer function. A gain error exists if the actual slope line through the transfer curve is different from the slope of the best-fit line for ideal case. The ideal transfer function has a slope defined by drawing a straight line through the two end points. In non ideal DACs, this slope can differ from ideal resulting in a gain error- which is usually expressed as a percentage because it affects each code by the same percentage. If there is no offset error, gain error is easily determined by applying the all "1"s code to the DAC and measuring its output, designed as V_{111} for 3-bit DAC (Kester, 2005). Gain error for 3-bit DAC is illustrated in Fig.5.3.

$$GainError = \frac{V_{111}}{V_{FS} - 1LSB} - 1 \tag{5.4}$$

where V_{FS} is the full scale value of ideal DAC characteristic and V_{111} refers to endpoint analog value of actual DAC transfer curve. Gain error for 3-bit DAC is illustrated in Fig.5.3.

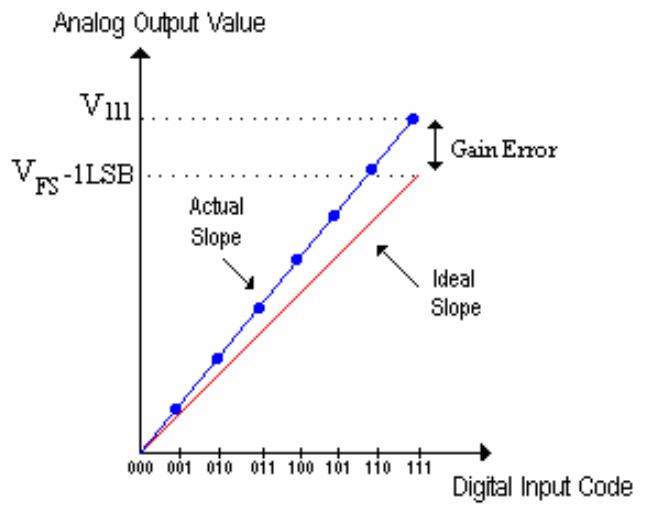

Fig.5.3. Gain error for 3-bit DAC

- *Monotonicity*: Monotonicity is defined by the output always incrementing by a non-negative amount as the input code increases (Razavi, 1999). DAC transfer function is defined as monotonic under the condition that as the digital input is increased in value, the output should also increase or may approach a zero change in the increasing direction, but never decreases in value (Hoeschele, 1994). Fig.5.4 depicts a nonmonotonic transition.

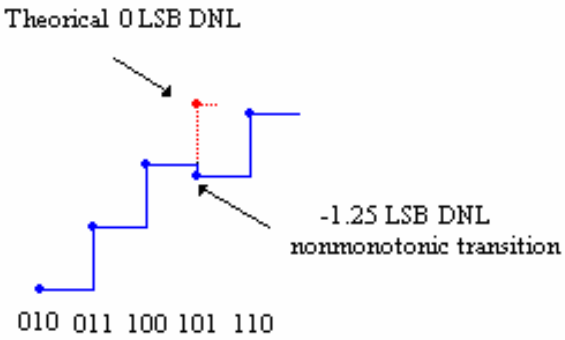

Fig.5.4. Illustration of nonmonotonic transition for 3-bit DAC

5.3 ANN Based Design Methodology

MOS transistors of submicron integrated circuit process technologies are modeled by complex nonlinear equations with dimensional and process-parameter dependencies. BSIM models were developed to solve the problems existing in semi-empirical models. They have extensively built in dependencies of important dimensional and process parameters such as channel length, channel width, gate oxide thickness, junction depth, doping concentration, etc. BSIM3v3 has became an industry standard for modeling submicron MOS technologies. In standard integrated circuit design, the design of the circuit schematic and its layout affect the circuit operation. MOSFET channel width (W) and length (L) parameters directly affect the current driving capability of the corresponding transistor. In submicron technologies, it is difficult to choose the most suitable W and L values since the MOSFETs are modeled by complex nonlinear equations (Abebe and Tyree, 2000).

Nonlinear modeling technique based on ANN models is increasingly being used for complex system models. Here, the aim is to estimate transistor dimensions of a DAC structure for given SSP values when newer technology model parameters are chosen for redesigning the circuit structure that was previously designed using older technology parameters. General technology independent design methodology is given in Fig. 5.5.

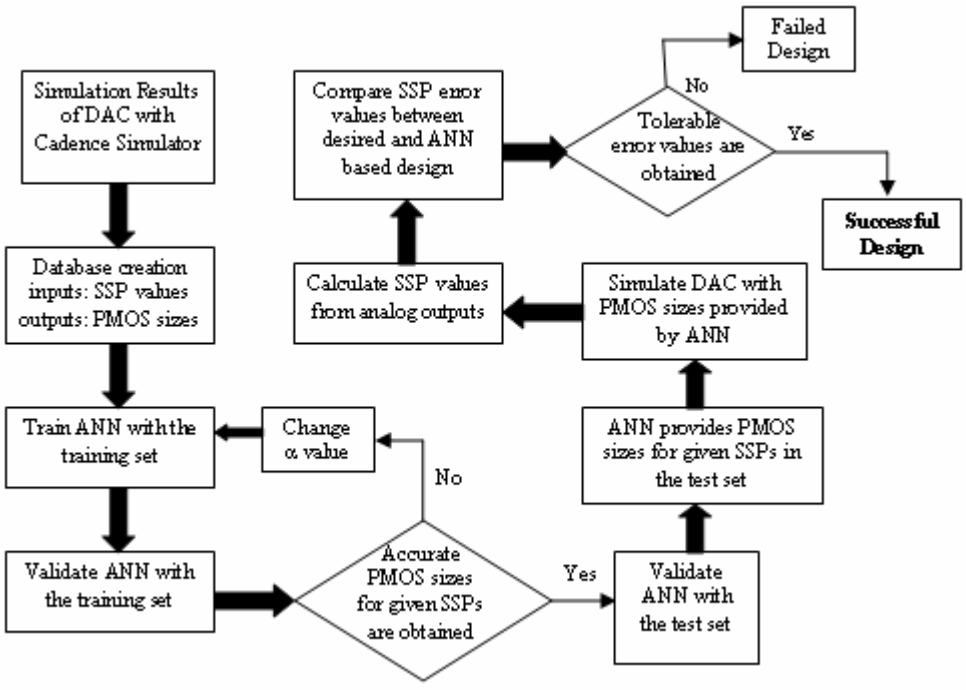

Fig.5.5 General Technology Independent Design Methodology for DAC

First, DAC structure is designed using three different technology parameters with possible combinations of transistor sizes in order to obtain the database for ANN (Serin, 2008). Secondly, an appropriate ANN architecture is selected which can process a large sized database with high accuracy. Finally, SSP values of the proposed model are compared with actual values provided in the test set. Details are provided in the following.

5.3.1 Measurement of SSP Values and Creation of DAC Database

One of the most crucial steps in ANN methodologies is undoubtedly the creation of the circuit database which is composed of the training set and the test set. In order to generate these sets, first of all, DAC structure given in Fig.5.1 was designed using ON SEMI 1.5 µm technology parameters with PMOS transistor sizes tabulated in Table 5.1 and simulated with Cadence Analog Environment. Resulting transient response is given in Fig. 5.6 (Serin, 2008).

Table 5.1 PMOS width and length values for DAC design with ON SEMI 1.5 µm technology

	P0	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**
W(µm)	96	96	24	96	48	48	16	48	24
L (µm)	1.6	1.6	1.6	1.6	1.6	1.6	1.6	1.6	1.6
	P11	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	
W(µm)	24	16	24	12	12	16	12	30	
L (µm)	1.6	1.6	1.6	1.6	1.6	1.6	1.6	1.6	

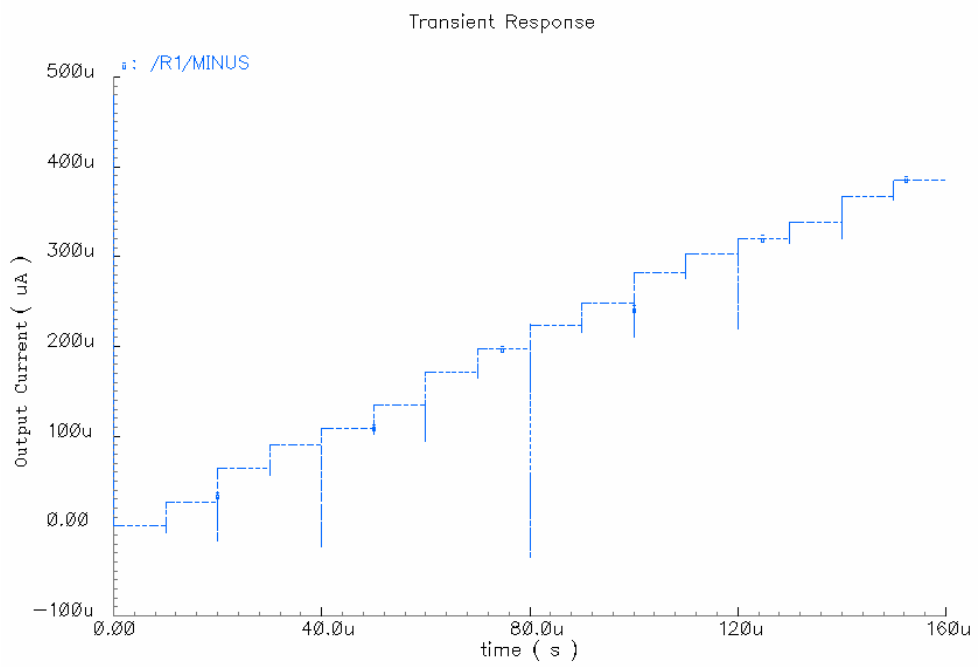

Fig. 5.6 Transient response of DAC design with ON SEMI 1.5 µm (Serin, 2008)

Static characterization equations were provided assuming that outputs were in terms of voltages. Since selected DAC structure is current mode, the equations are utilized in terms of currents. Ideal output current values are calculated using (5.1) and tabulated in Table 5.2. *DNL* error values are calculated as in (5.2) and given in Table 5.3. Actual output current values are provided from transient simulation of DAC as in Fig.5.5 and $I_{LSB\text{-}IDEAL}$ is equal to 6.25 µm. INL error values are obtained using (5.3) and tabulated in Table 5.4.

Table 5.2 Ideal output current values for each digital input word

	I_{Dideal} (µA)		I_{Dideal} (µA)
D=0	I_{Dideal}=100x0=**0**	D=8	I_{Dideal} =100x8/2^4= **50**
D=1	I_{Dideal} =100x1/2^4= **6.25**	D=9	I_{Dideal} =100x9/2^4= **56.25**
D=2	I_{Dideal} =100x2/2^4= **12.5**	D=10	I_{Dideal} =100x10/2^4= **62.5**
D=3	I_{Dideal} =100x3/2^4= **18.75**	D=11	I_{Dideal} =100x11/2^4= **68.75**
D=4	I_{Dideal} =100x4/2^4= **25**	D=12	I_{Dideal} =100x12/2^4= **75**
D=5	I_{Dideal} =100x5/2^4= **31.25**	D=13	I_{Dideal} =100x13/2^4= **81.25**
D=6	I_{Dideal} =100x6/2^4= **37.5**	D=14	I_{Dideal} =100x14/2^4= **87.5**
D=7	I_{Dideal} =100x7/2^4= **43.75**	D=15	I_{Dideal} =100x15/2^4= **93.75**

Table 5.3 DNL error values for each digital input word

	DNL Error		***DNL Error***
D=0	***DNL0***=(5.69-0)/6.25-1= **-0.09**	D=8	***DNL8*** =(55.24-49.7)/6.25-1 = **-0.11**
D=1	***DNL1***=(13.74-5.69)/6.25-1 = **0.29**	D=9	***DNL9***=(63.31-55.24)/6.25-1 = **0.29**
D=2	***DNL2***=(19.38-13.74)/6.25-1= **-0.1**	D=10	***DNL10***=(68.94-63.31)/6.25-1= **-0.1**
D=3	***DNL3***=(23.63-19.38)/6.25-1 = **-0.32**	D=11	***DNL11***=(73.34-68.94)/6.25-1= **-0.3**
D=4	***DNL4*** =(29.23-23.63)/6.25-1= **-0.1**	D=12	***DNL12***=(78.94-73.34)/6.25-1= **-0.1**
D=5	***DNL5***=(37.26-29.23)/6.25-1 = **0.28**	D=13	***DNL13***=(87.21-78.94)/6.25-1 = **0.32**
D=6	***DNL6*** =(42.88-37.26)/6.25-1= **-0.1**	D=14	***DNL14***=(92.97-87.21)/6.25-1= **-0.08**
D=7	***DNL7*** =(49.7-42.88)/6.25-1 = **0.09**		

Table 5.4 INL error values for each digital input word

	INL Error		***INL Error***
D=0	***INL0***=(0-0)/6.25 = **0**	D=8	***INL8*** =(49.7-50)/6.25 = **-0.05**
D=1	***INL1***=(5.69-6.25)/6.25 = **-0.09**	D=9	***INL9***=(55.24-56.25)/6.25 = **-0.16**
D=2	***INL2***=(13.74-12.5)/6.25 = **0.2**	D=10	***INL10***=(63.31-62.5)/6.25= **0.13**
D=3	***INL3***=(19.38-18.75)/6.25 = **0.1**	D=11	***INL11***=(68.94-68.75)/6.25= **0.03**
D=4	***INL4*** =(23.63-25)/6.25 = **-0.22**	D=12	***INL12***=(73.34-75)/6.25= **-0.27**
D=5	***INL5***=(29.23-31.25)/6.25 = **-0.32**	D=13	***INL13***=(78.94-81.25)/6.25 = **-0.37**
D=6	***INL6*** =(37.26-37.5)/6.25 = **-0.04**	D=14	***INL14***=(87.21-87.5)/6.25= **-0.05**
D=7	***INL7*** =(42.88-43.75)/6.25 = **-0.14**	D=15	***INL15***=(92.97-93.75)/6.25= **-0.12**

Gain error is calculated as -0.008 using (5.4) and from Fig.5.5, it can be seen that transient response is monotonic. Resulting error values are small enough to ensure that designed DAC is operating properly and actual output values are very close to ideal ones.

Following, both channel width and length values of each PMOS transistor in DAC design with ON SEMI 1.5 µm are variated and 138 different DAC structure is designed and simulated (Serin, 2008). Depending on the resulting transient response of each design DNL errors (DNL0:DNL14), INL errors (INL0:INL15) and gain errors are calculated. DAC design having a monotonic or non-monotonic response is expressed with 0.9 or 0.1, respectively.

Abovementioned tasks are repeated during DAC design with both ON SEMI 0.5 µm and TSMC 0.35 µm technology parameters. PMOS sizes of DAC design having very small static error values are tabulated in Table 5.5 for ON SEMI 0.5 µm technology and in Table 5.6 for TSMC 0.35 µm technology (Serin, 2008). Consequently, 414 different DAC design is realized where each design has 30 error value and a monotonicity marker.

Table 5.5 PMOS width and length values for DAC design with ON SEMI 0.5 µm technology

	P0	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**
W(µm)	40	40	10	40	20	20	6.6	20	10
L (µm)	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6
	P11	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	
W(µm)	10	6.6	10	5	5	6.6	6.6	12.5	
L (µm)	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	

Table 5.6 PMOS width and length values for DAC design with TSMC 0.35 µm technology

	P0	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**
W(µm)	24	24	6	24	12	12	4	12	6
L (µm)	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
	P11	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	
W(µm)	6	4	6	3	3	4	3	10	
L (µm)	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	

Since the aim was to predict the transistor sizes of DAC circuit that meets the design constraints, with minimum user effort and design knowledge for a newer semiconductor technology, simulations were carried out using ON SEMI 1.5 µm and 0.5 µm technology parameters for training set and TSMC 0.35 µm technology parameters for the test set. By using the specified technology parameters and changing width (W) and length (L) values of PMOS transistors, transient simulations were performed in order to obtain analog outputs for different transistor dimensions and technology parameters, since the possible combinations of transistors' sizes directly affect linearity of DAC circuitry. Transistor dimensions (W0:W16, L0:L16) constitute the outputs of ANN, while the inputs are the static specifications (*INL0:INL15*, *DNL0:DNL14*, monotonicity, gain error) which were calculated with the use of DAC outputs. The training set is composed of 276 samples, while the test set includes 138 samples as illustrated in Fig. 5.7. Here, the acronym of MONO stands for the monotonicity.

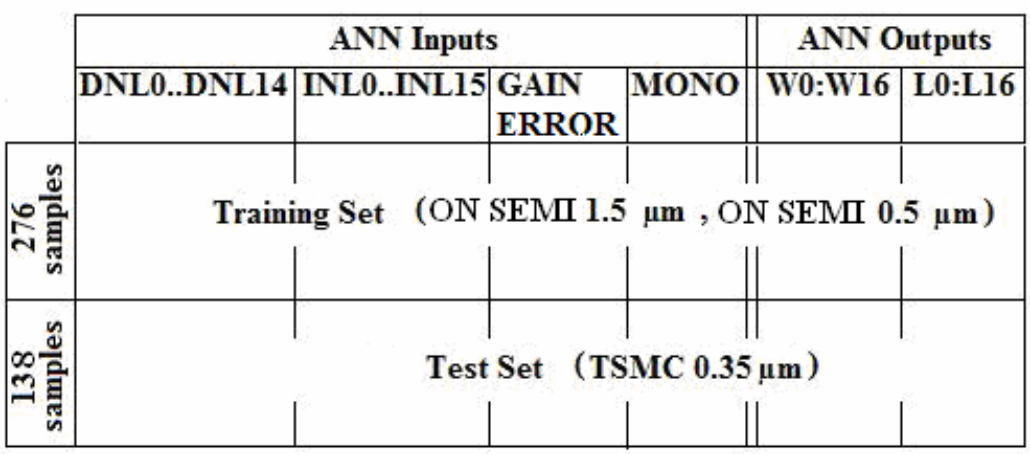

Fig. 5.7 DAC Database Structure

5.3.2 Selection of Appropriate ANN Structure

Due to the complexity of DAC database, selection of an appropriate ANN structure is very important. In this study, MLP, RBF and GRNN structures are considered. MLP is trained with three different methods; BP, LM and PSO. Except PSO-MLP, all of the ANN structures are constituted using MATLAB Neural Network Toolbox. Performances of each method for DAC design dataset with 33 inputs and 34 outputs are provided in the following.

First, MLP is trained with BP algorithm. Neuron number in the hidden layer is selected as 25, 40, and 55 for each trial. However, execution of BP-MLP (with 25 hidden neurons) for 13 iterations lasted 33 minutes and after 13 iterations, training was terminated due to the validation check. Following, MLP was trained with LM with a greater execution time of 46

minutes for 10 iterations due to the derivation procedures and training was again terminated due to the validation check. Finally MLP was trained with PSO. The duration of training lasted is 0.7 minutes; however training error values cannot be converged to the desired values as given in Table 5.7. Here, neuron number in hidden layer is tuned with varying number of particle vectors. *D* represents the dimension of each particle with respect to number of hidden neurons. Maximum number of iteration is set as 100. RBF was successful with the training phase by adjusting spread values but it failed during testing which demonstrates that it has not learned but it has memorized.

Table 5.7 Training error values depending to PSO-MLP parameters

MSE		**# of Hidden Neurons (HN)**					
		HN=36 (D=2376)	*HN=50 (D=3300)*	*HN=100 (D=6600)*	*HN=150 (D=9900)*	*HN=200 (D=13200)*	*HN=300 (D=19800)*
# of Particles (P)	*P=50*	8.58×10^{5}	$\mathbf{2.7 \times 10^{5}}$	5.6×10^{6}	6.14×10^{6}	6.47×10^{6}	3.9×10^{7}
	P=40	3.1×10^{5}	1.43×10^{6}	9.53×10^{6}	9.05×10^{6}	4.1×10^{7}	8.56×10^{7}
	P=30	1.04×10^{6}	1.05×10^{6}	2.35×10^{7}	2.35×10^{7}	3.3×10^{7}	8.86×10^{7}

Abovementioned ANN methods were failed to process DAC database with 33 inputs and 34 outputs. The remaining method, GRNN, not only obtained satisfying training results but also improves the convergence rates and concludes training in 5.6 minutes. The main advantage of GRNN over other ANN methods is that its internal structure is not problem dependent. The number of hidden layer nodes is equal to the number of training samples and cannot, nor need it, be modified. This eliminates much time spent designing the network as required for MLP. In order to validate whether trained GRNN was able to generalize, its performance is evaluated with the test set. Unlike RBF, GRNN has achieved a satisfying test performance. Performance of GRNN based design methodology is evaluated in the next subsection.

5.3.3 GRNN Based Design Methodology

GRNN structure was constituted with 33 neurons in the input layer and 34 neurons in the output layer using MATLAB®. All training samples were stored in the pattern layer. In order to improve the performance, all values in database were scaled between 0.1 and 0.9. During training phase, σ was chosen as 0.1. After training has been completed in 5.6 min, GRNN was validated with the training set in order to evaluate the learning ability for given sigma value.

As a result, for all given SSP values in the training set, GRNN provides correct PMOS sizes compatible with ON SEMI 1.5µm and 0.5µm technology parameters, respectively. Since successful results were obtained, GRNN was validated with the test set as the next step. The success of the validation with the test set indicates that GRNN based method can redesign the DAC with a new technology using the knowledge of previously designed DAC results using older technology parameters.

Estimated *W* and *L* values of 16 PMOS transistors were used for DAC design in Cadence Analog Environment, as to obtain actual output current values of GRNN-based design. Error between actual and desired output current values demonstrates how approximately static error values can be achieved with GRNN based DAC design methodology. Tolerable SSP errors would mean that the specifications set by the user has been provided by GRNN based design method with acceptable deviations. Design methodology is summarized in Fig. 5.8.

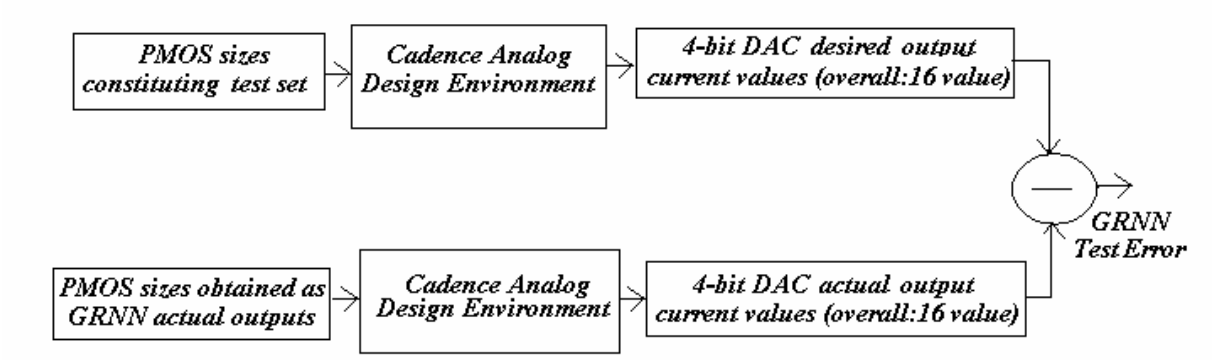

Fig.5.8 Validation of Test Results with GRNN Based Design Methodology

In the first step of GRNN based methodology, DAC is designed in Cadence Analog Design Environment with PMOS sizes that constituted the test set and resulting output current values are used for calculating SSP values. Obtained SSP values are introduced to the GRNN as test inputs. During test process, GRNN provides *W* and *L* values of each PMOS transistor for a newer technology that GRNN was not introduced during training. Resulting *W* and *L* values of each PMOS transistor is then utilized to redesign DAC structure. Following, redesigned DAC is simulated in Cadence Design Environment and output current values are corresponding to the actual outputs of the GRNN based design methodology.

5.4 Simulation Results

Validation of test samples resulted that ANN based design successfully provides PMOS width and length values in newer technology for given static specifications parameters. The relative error between desired and actual outputs of a test sample is shown in Table.5.8. First column represents the digital word number; second and third columns are the desired and actual output values for the considered test sample, respectively. Relative error for each digital word is calculated by dividing the difference between desired and actual output value to the desired value. Average of total 16 relative error values is the test error of GRNN based design methodology for the considered sample which is given in the fifth column.

Table 5.8 Performance of GRNN based design methodology for test sample–1

	Desired (µA)	**GRNN (µA)**	**Difference (µA)**	**Relative Error (%)**	**Average Relative Error Percent of GRNN for *test sample–1* = % 0.21**
0	0,00	**0,00**	0,00	**0,00**	
1	5,54	**5,49**	0,05	**0,90**	
2	13,64	**13,60**	0,04	**0,29**	
3	19,14	**19,05**	0,09	**0,47**	
4	23,68	**23,66**	0,02	**0,08**	
5	29,14	**29,10**	0,04	**0,14**	
6	37,23	**37,16**	0,07	**0,19**	
7	42,71	**42,60**	0,11	**0,26**	
8	49,46	**49,45**	0,01	**0,02**	
9	54,87	**54,85**	0,02	**0,04**	
10	62,98	**62,93**	0,05	**0,08**	
11	68,46	**68,37**	0,09	**0,13**	
12	73,12	**73,07**	0,05	**0,07**	
13	78,58	**78,49**	0,09	**0,11**	
14	86,86	**86,65**	0,21	**0,24**	
15	92,44	**92,15**	0,29	**0,31**	

The whole database contains not only proper DAC design samples where *DNL* and *INL* values are between limits of ± 0.5 LSBs but also nonmonotonic design samples. Simulation results show that GRNN based design method improves monotonicity, *DNL* and *INL* for numerous nonmonotonic samples. Performance of GRNN based design methodology for 72 proper test samples is illustrated in Fig. 5.9. Here, test error is smaller than 3% for half of the proper test samples.

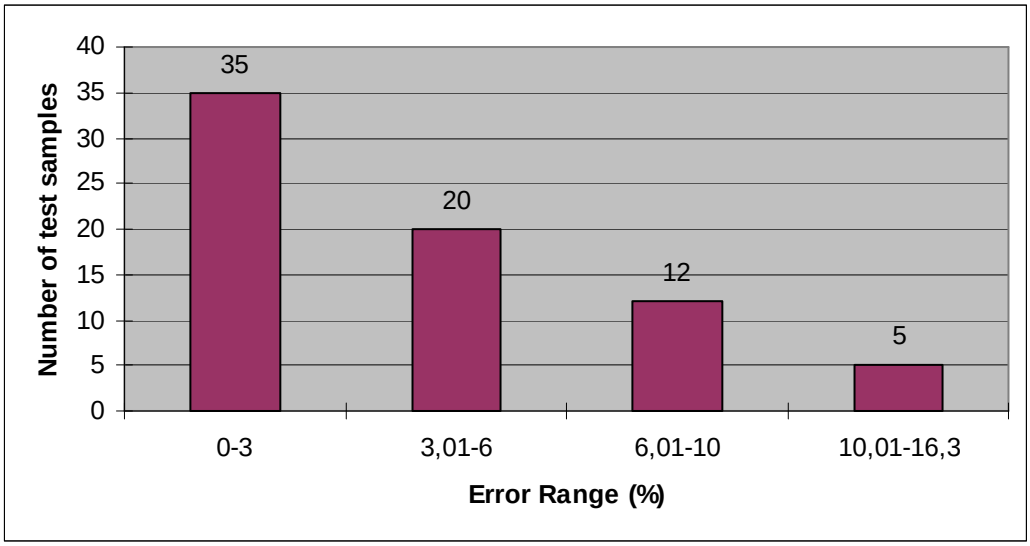

Fig.5.9. Test error of GRNN Based Design Methodology for proper samples

Considering proper DAC design samples, transient simulations with the PMOS sizes obtained from the proposed method results in compatible SSP values with the actual test inputs. Table 5.9 shows the GRNN based estimation performance of *DNL* and *INL* errors for whole test set. The term *"within limits"* represent the samples having a resolution less than ±1/2 LSB of *DNL* or *INL* whereas *"beyond limits"* represent the samples having a resolution more than ±1/2 LSB of *DNL* or *INL*. *"Improvement"* of *DNL* and *INL* values means that the *DNL* and *INL* errors beyond limits of samples in the test set have been reduced after GRNN based design. However, *"deteriorated" DNL* and *INL* values shows that the *DNL* and *INL* errors within the limits of the samples in the test set are increased after GRNN based design. For some test samples, results of GRNN based design remain the similar *DNL* and *INL* values by means of beyond limits or within limits compared with the original test set. In other words, for some test samples having any number of *DNL* or *INL* values beyond limits or within limits, GRNN based design method also resulted in same number of *DNL* or *INL* values when compared with the test set.

Table 5.9 DNL and INL results of GRNN based method for whole test set

	Improvement		Deterioration		Beyond limits (Unchanged)		Within limits (Unchanged)	
	in *DNL*	in *INL*	in *DNL*	in *INL*	in *DNL*	in *INL*	in *DNL*	in *INL*
# of test samples	70	71	8	23	14	14	46	30

In the database, inputs of each sample are gain error value, monotonicity marker, 15 *DNL* and 16 *INL* values for estimating transistor sizes of 4-bit DAC. As mentioned before, *DNL* and *INL* values must be between limits of ± 0.5 LSBs for appropriate DAC design. After GRNN based design, *DNL* and *INL* results of each test samples are evaluated for these characterizations; improvement, deterioration and unchanged values by means of beyond and within limits. Table 5.10 shows the number of *DNL* and *INL* errors that are beyond the limits for each selected test sample as an exemplification. Here, (−) indicates that all *DNL* or *INL* values for selected test sample are within limits.

Table 5.10 Number of DNL and INL errors beyond limits for each selected test sample

	DNL (Original)	*DNL* (After GRNN)	*INL* (Original)	*INL* (After GRNN)
Improvement (57^{th} sample)	3	-	8	-
Deterioration (129^{th} sample)	–	1	–	1
Unchanged-Beyond Limits (114^{th} sample)	4	4	2	2
Unchanged-Within Limits (132^{th} sample)	–	–	–	–

Monotonicity of test samples after GRNN based design method was also investigated. Amongst 138 test samples, there were 107 monotonic DAC design samples as well as 31 nonmonotonic samples. The proposed method affects the monotonicity for none of the monotonic samples. Furthermore, GRNN based design method improves monotonicity for 18 samples which were nonmonotonic in the test set. Fig.5.10 shows an example of how an actual output of a nonmonotonic test sample was improved with GRNN based method.

Same sample of the test set had also 3 *DNL* errors and 8 *INL* errors beyond limits before being introduced to the GRNN based method. The proposed method not only improved monotonicity but also reduced all *DNL* and *INL* error values within the limits as shown in Fig. 5.11 and Fig. 5.12.

Gain error remained same for the fifty seventh sample after GRNN based design. However, considering the whole test set, for fifty three samples gain error was increased slightly with a maximum error of 1% and for fifty six samples, gain error remained same with a maximum tolerance of 0.1% as compared to the test set. GRNN based design has reduced the gain error of twenty nine samples. In Fig. 5.13, the reduction of gain error for the sixty fourth sample after GRNN based design method is shown as an example.

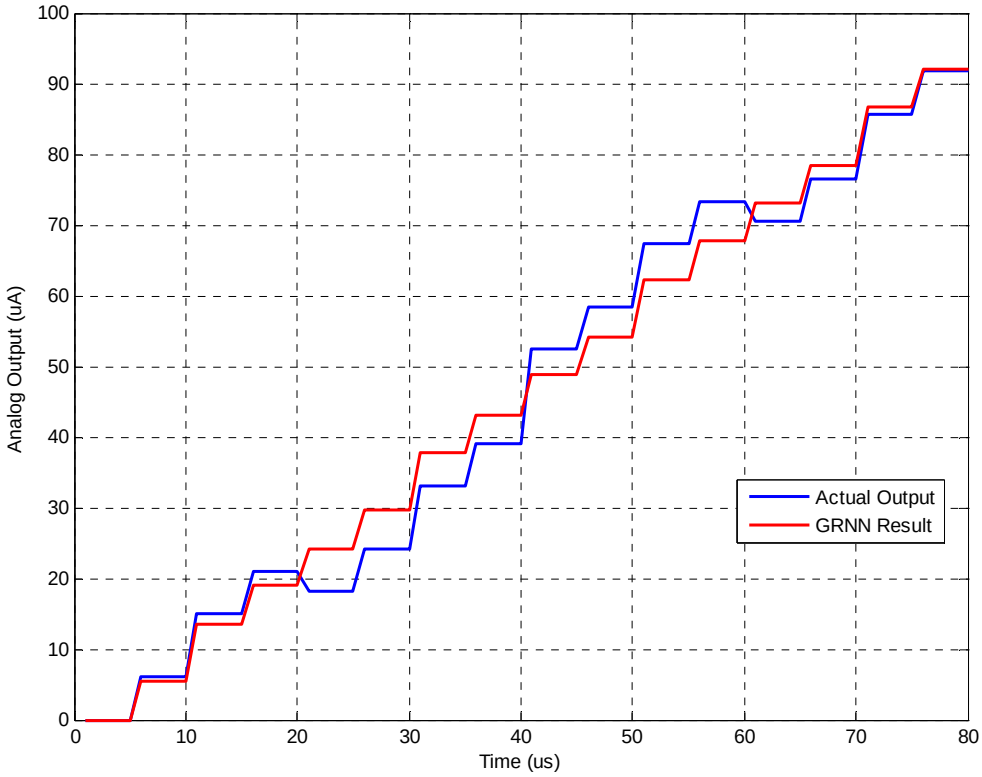

Fig. 5.10 Improvement of monotonicity for 57th test sample by GRNN Based Methodology

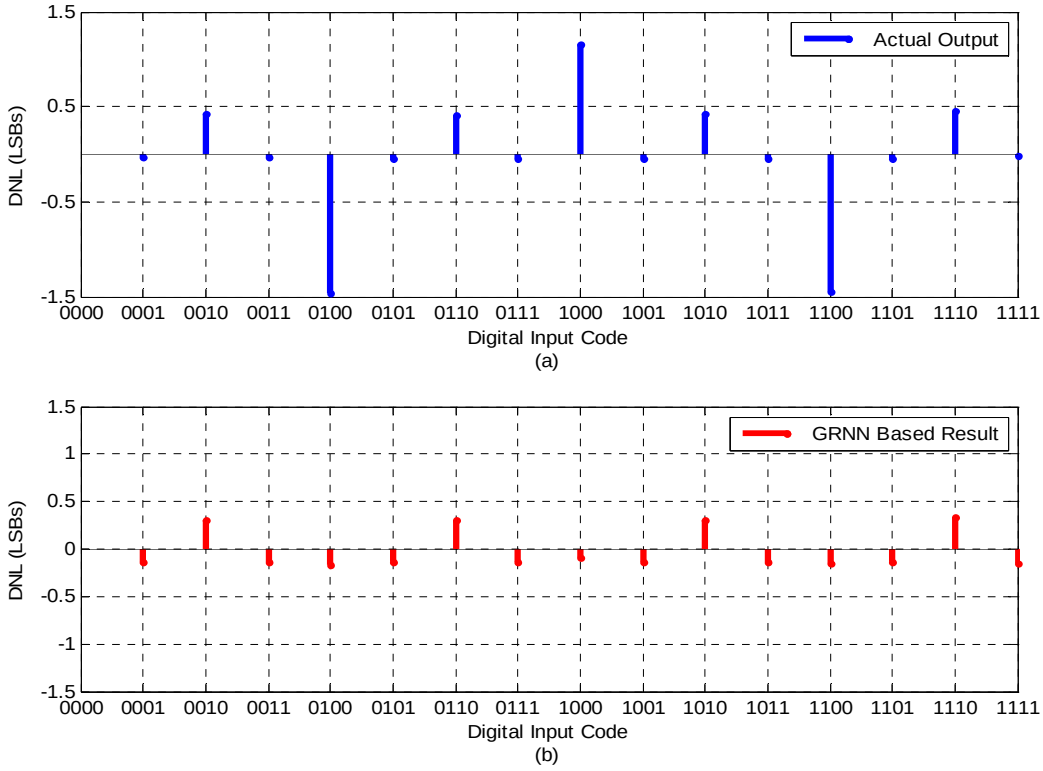

Fig. 5.11 DNL Improvement of GRNN based methodology (b) over actual output (a) for 57th test sample

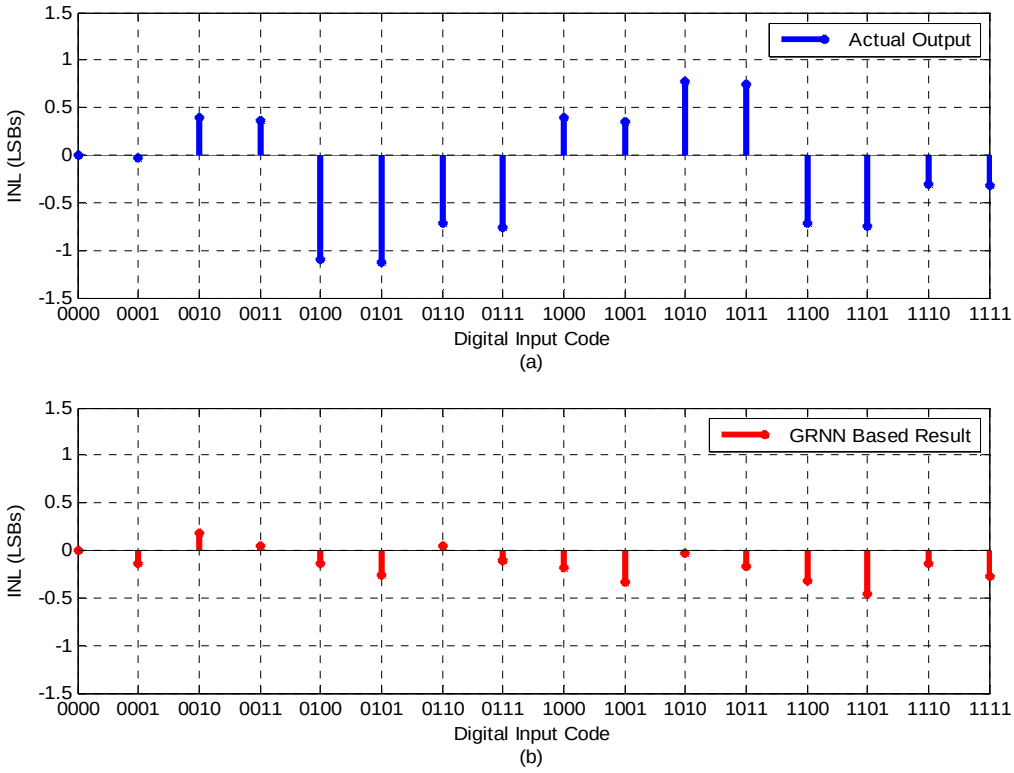

Fig. 5.12 INL Improvement of GRNN based methodology (b) over actual output (a) for 57th test sample

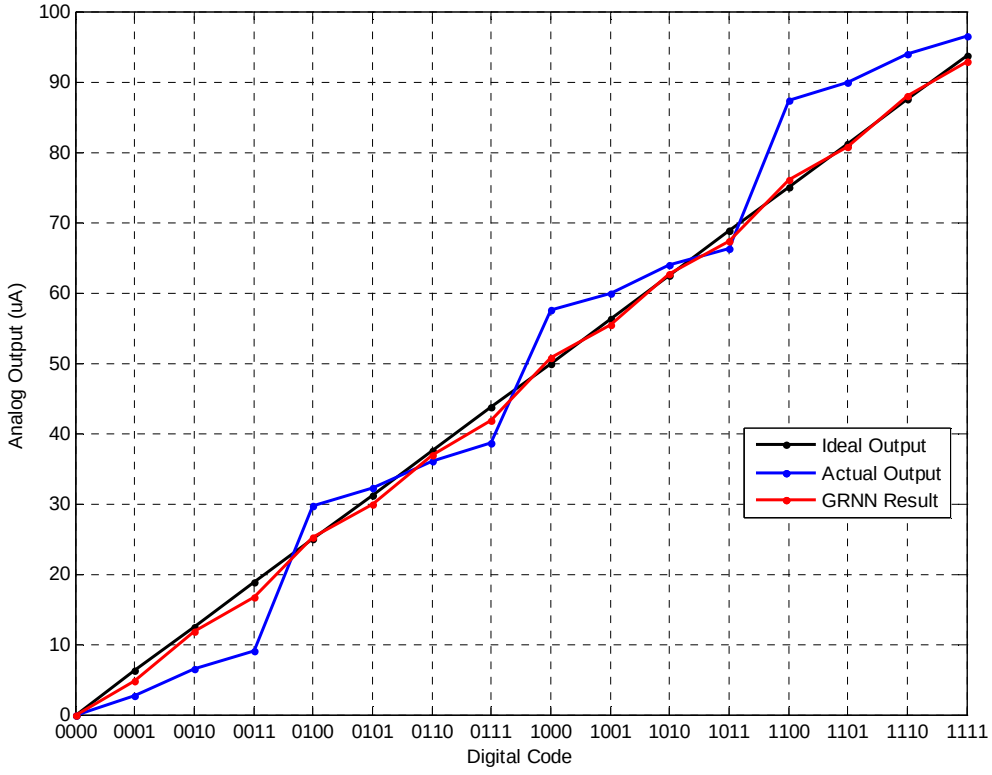

Fig.5.13 Reduction of gain error for 64th sample by GRNN based methodology

5.5 Summary

Dimension prediction of circuits in new technologies with the usage of neural networks can provide great convenience on the design of VLSI circuits. (Kahraman, 2008) utilized this approach for fundamental analog circuit (current mirror, differential amplifier, operational amplifier) and digital gate (NOR, NAND, XOR, INV) design. As a different and more complex structure, a digital to analog converter (DAC) is aimed to be designed using the same approach. Here, transistor sizes for current steering DAC circuit for newer technologies, has been successfully predicted using ANN based design methodology. Possible combinations of transistor sizes, the channel widths and lengths, were changed in order to construct the data set. Transient simulations of DAC were performed for various channel widths of all transistors that provide different output curves using Cadence with three different BSIM3v3 MOSFET technology parameters. The DAC circuit structures with fixed topology but different set of MOS transistor dimensions, those used for constructing the dataset for GRNN, do not always have a good output curve. Some of them have worse linearity, some of them are nonmonotonic and some of them have more gain error. Eventually, a large database is developed consisting of transistor sizes of DAC designs for and static characteristics of DAC designs as obtained from simulation results. Following, the appropriate ANN was selected and trained with the dataset including the simulation results of ON SEMI 1.5µm and 0.5µm technologies and the test data was constituted with the simulation results of TSMC 0.35µm technology which were not applied to the ANN for training beforehand.

During selection of appropriate ANN structure for this problem, BP-MLP, LM-MLP, PSO-MLP, RBF and GRNN are constructed. BP and LM failed to train the MLP architecture due to the heavy computation effort including derivative operations. To our knowledge, PSO-MLP has not been applied to such high-dimensional database before. However, due to the large number of input neurons, particle vector that includes weights between hidden layer-input layer and output layer-hidden layer has increased tremendously. Therefore, iterative nature of MLP is not suitable for this very-high dimensional data notwithstanding that a very efficient and a fast optimization algorithm is utilized for training. RBF concluded training successfully however failed during testing which indicates that RBF has memorized the data and therefore were not able to generalize. Among various ANN structures, GRNN method proved its effectiveness on processing this high-dimensional problem with high accuracy in short computation time. GRNN based design provides the channel lengths and widths of all

transistors which satisfy the design constraints (*DNL*, *INL*, monotonicity and gain error) that are also the inputs of the neural network.

The success of the validation with TSMC 0.35 μm test set resulted that the proposed method can redesign the DAC with a new technology using the knowledge of previously designed DAC simulation results for older technology. Moreover, the proposed method not only improved monotonicity but also reduced *DNL*, *INL* and gain error values within some limits. As a result, it was proved that the ANN based design approach provides the channel widths and lengths of all transistors in a newer technology when a user determines the static specification parameters, with minimum effort and design knowledge.

6. CONCLUSION

In this dissertation, the applicability of PSO in electronic circuit design automation has been investigated comprehensively. For this purpose, PSO is utilized as an optimization tool for both discrete and integrated circuit design.

Optimal passive component selection for analog active filter is considered as the discrete circuit design problem. GA, ABC and PSO algorithms were used for both 4^{th} order Butterworth low-pass analog filter and 2^{nd} order SVF design and were investigated for the selection of passive components from different manufactured series by means of accuracy and execution time. Selection of the optimal own parameters is very crucial on minimizing total design error value thus effecting filter performance. Among the evolutionary approaches used in this work, GA requires more fine-tuning of the own parameters and increasing number of chromosomes decreases total error values at the cost of execution time. Considering ABC, increasing search limit value facilitates obtaining better CF values when bee population is small. However, when bee population is crowded than selection of bigger search limits decelerates the algorithm and worsens the performance. Increasing particle number in PSO improves the performance unless the acceleration factors are selected as 2. Following, 20 runs were performed with optimal own parameters obtained previously. The resulting CF values were used to produce box and whisker plots which show that CF values obtained with GA method varies the most among the other methods. The iteration number required to achieve the quality restrictions are slightly different in each run for each method which can be seen in the plots of CF values versus iteration number for 5 different runs. Considering Butterworth filter design with components selected from E12 series for a true comparison with (Jiang et al., 2007); PSO achieved the smallest design error with respect to previous methods and other EA methods. Moreover, less design error is obtained with GA than the previously used one in (Jiang et al., 2007).

Components of SVF are selected from two different manufactured series in order to investigate whether performance of EA methods will increase or not when same topology is designed with different series. GA algorithm achieved the smallest design error but the longest execution time when selecting components from E24 series. This is mainly due to the fact that PSO and ABC has fewer primitive mathematical operators than in GA (e.g reproduction, mutation and crossover). Those mathematical operations require more fine-tuning of own parameters. However when tolerances of components became tighter in E96 series, ABC algorithm obtained the most successful results by means of both accuracy and

execution time. Choosing optimal search limit value avoids local minimum trapping and increasing number of bee population improves the probability of converging to global minimum. Therefore, selecting optimal own parameters for ABC improves the accuracy as well as the convergence rate. ABC outperforms other methods by means of execution time for all design cases; however, when considering design accuracy each method has achieved the smallest design error depending on the design case. Moreover, the performance of PSO was not affected significantly due to the usage of different manufactured series for same filter topology as other EA methods were. From Spice simulation results, it is clearly observed that the conventional method does not provide a maximally flat response in the pass band and results in bigger cut-off frequency deviation. SVF design with conventional method also screens stop band ripples during frequency analysis unlike with EA techniques.

Next, PSO is utilized for both digital and analog IC design cases. In order to investigate PSO performance for digital circuit design, switching characterization of an inverter is considered. Switching performance is one of the quality metrics when considering a digital design. From a system designer's perspective, the performance of a digital circuit expresses its computational ability. The switching characteristics of digital integrated circuits and in particular, of inverter circuits, essentially determine the overall operating speed of digital systems. The transient performance requirements of a digital system are usually among the most important design specifications that must be met by the circuit designer. Therefore, the switching speed of the circuit must be estimated and optimized very early in the design phase.

For this purpose a CMOS inverter circuit has been demonstrated and three different design cases concerning transient characteristics were carried out. First, PSO algorithm was configured in order to estimate output voltage fall time depending on the design parameters and fabrication technology parameters. 10 different ranges of design parameters and design criterion were specified for PSO algorithm. Synthesized results were compatible with theoretical calculations. Second case deals with an inverter design having a symmetrical output voltage. For this purpose, error between fall time and rise time of output voltage was minimized by PSO algorithm for 10 inverter design having different ranges for design parameters and design criteria. Differences of fall and rise times of nine of the examples were below than specified target error. For the specified ranges of the eighth example, PSO algorithm could not find a minimum error value smaller than the specified error. This example was investigated on purpose, in order to show the importance of range selection for the success of optimization algorithm. The last case was the most complicated one among the

others. A symmetrical output voltage was also of concern by including the minimization of propagation delay times. CF, namely error, was defined by the sum of difference between output voltage delay times and propagation delay times. As seen from the synthesized results, PSO obtained design parameters and design criterion values compatible with specified ranges. CF of nine of the design examples was below the target error and specified ranges of the remaining example did not allow PSO to find a minimum CF smaller than the target error. The PSO technique proved its effectiveness in finding minimum error for all design problem cases. Almost equal output voltage delay times and propagation delay times lead to a symmetrical output voltage which determines the overall operating speed of digital systems.

For the first design case, it was assumed that PMOS and NMOS dimension ratios were equal to each other. However for the following examples, different dimension ratios of PMOS and NMOS has been obtained by PSO algorithm and synthesized results showed that symmetrical delay times could not be obtained with a dimension ratio of PMOS to NMOS smaller than five. Decreasing PMOS dimension leads to a smaller area on chip layout but worse symmetry at the output voltage.

For comparison, using PSO results of output capacitance value and transistor dimensions, inverters were redesigned in SPICE environment for each case, considering PSO results as SPICE inputs and obtained. The difference between SPICE results and PSO-based design results arise from the fact that SPICE computes using more complex circuit equation sets than used in theoretical calculations. PSO uses delay expressions which were derived using simple current-voltage relationships originally developed for long-channel transistors. These current expressions based on the gradual channel approximation can still be used for sub-micron MOS transistors with proper parameter adjustments; therefore delay analysis used here remains largely valid for small-geometry devices as well. Yet it should be noted that the current driving capability of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor cannot be expected to have the same maximum charge/discharge current as a long-channel transistor with the same (W/L) ratio. Thus, SPICE results in greater delay times compared to PSO-based inverter design.

While the amount of digital design activity far outpaces that of analog design, most digital systems require analog modules for interfacing to the external world. Regarding analog ICs, design scheme is perceived as less systematic and more heuristic and knowledge-intensive in nature than digital IC design. The variety of circuit schematics and the number of conflicting requirements and corresponding diversity of device sizes are also much larger. Analog IC

design mainly consists of topology choice, sizing task and the generation of layout. The problem considered here is the optimal selection of transistor dimensions, which is only a part of a complete analog circuit CAD tool. Actually, analog sizing is a constructive procedure that aims at mapping the circuit specifications where the performance metrics of the circuit, such as gain, power dissipation, occupied area, etc. have to be formulated in terms of the design parameters. Following, these design parameters such as device sizes and bias currents should be adjusted under multiple design objectives and constraints. The many degrees of freedom in parameter space as well as the need for repeated circuit performance evaluation made this a lengthy and tedious process. Here, particular specifications for specified topologies of a differential amplifier with current mirror load and a two-stage operational amplifier are aimed to be met by adjusting design parameters such as device sizes and bias currents with PSO algorithm.

Design equations of each analog circuit are utilized for multi-objective cost function of PSO algorithm, since numerous design specs are of concern. Then, SPICE simulations are carried out in order to validate the feasibility of synthesized circuits. Considering differential amplifier with current mirror load, design performance of PSO-based method is first compared with classical method (Allen and Holberg, 2002). Having satisfactory results, PSO-based design is utilized for TSMC 0.35 µm technology parameters and design process is concluded in 25 s. Using the resulting design parameters, synthesized circuit is simulated with SPICE in order to validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are not coherent with PSO based design results depending on the difference of circuit equation sets utilized in SPICE and theoretical calculations. Current driving capability of sub-micron transistors is significantly reduced as a result of channel velocity saturation; a small-geometry transistor cannot be expected to have the same maximum charge/discharge current as a long-channel transistor with the same (W/L) ratio. Therefore, in order to make PSO-based design more feasible bias current is increased while saving MOS sizes obtained with PSO-based design. This improvement not only satisfied all the constraints but also minimized the total MOS area with respect to DARWIN (Kruiskamp and Leenaerts, 1995) tool.

Two-stage operational amplifier design involves more design parameter adjusting than the former design task while each analog design task case has the same dimension of particle vector size. PSO-based design for two-stage amplifier is concluded in 8.6 seconds after 100 iterations. Resulting design parameters are utilized for redesign in SPICE simulator in order to

validate the exact values of design specifications obtained with PSO. Spec results obtained after SPICE simulations are very coherent with PSO based design results except gain value. This is mostly due to the more complex channel modulation effect equations in SPICE simulations which have been discarded in theoretical calculations. Nevertheless, PSO-based two-stage operational amplifier design satisfied all the design specifications as well as minimized total design area with respect to convex optimization (Hershenson, 2001) tool.

Considering both design cases, PSO has proved its efficiency on analog IC design with high accuracy and short computation time. Analog IC design problem can be considered as a multi-objective constrained problem and as a global optimization tool, PSO is very successful with handling those conflicting objectives as long as design spec equations are well defined and introduced to PSO.

So far, considered design problems utilized a particular technology for MOS dimension estimation intended for an optimized design. Finally, MOS transistor sizes of 4-bit current steering DAC is predicted that meets the design constraints, with minimum user effort and design knowledge for a newer semiconductor technology, using ANN. Dimension prediction of circuits in new technologies with the usage of neural networks can provide great convenience on the design of VLSI circuits. In contrast to other modeling researches, the user can obtain the transistor sizes for corresponding design specifications, which are INL, DNL, monotonicity and gain error for a newer technology, using the ANN based design methodology trained with design knowledge of older technologies. For this purpose, first of all, possible combinations of transistor sizes, the channel widths and lengths, were changed in order to construct the data set. Transient simulations of DAC were performed for various channel widths of all transistors that provide different output curves using Cadence with three different BSIM3v3 MOSFET technology parameters. The DAC circuit structures with fixed topology but different set of MOS transistor dimensions, those used for constructing the dataset for GRNN, do not always have a good output curve. Some of them have worse linearity, some of them are nonmonotonic and some of them have more gain error. Eventually, a large database is developed consisting of transistor sizes of DAC designs for three different technologies and static characteristics of DAC designs as obtained from simulation results. Following the database creation, the appropriate neural network was trained with the dataset including the simulation results of ON SEMI 1.5µm and 0.5µm technologies and the test data was constituted with only the simulation results of TSMC 0.35µm technology which were not applied to the ANN for training beforehand.

During selection of appropriate ANN structure for this problem, BP-MLP, LM-MLP, PSO-MLP, RBF and GRNN are constructed. BP and LM failed to train the MLP architecture due to the heavy computation effort including derivative operations. To our knowledge, PSO-MLP has not been applied to such high-dimensional database before. However, due to the large number of input neurons, particle vector that includes weights between hidden layer-input layer and output layer-hidden layer has increased tremendously. Therefore, iterative nature of MLP is not suitable for this very-high dimensional data notwithstanding that a very efficient and a fast optimization algorithm is utilized for training. RBF concluded training successfully however failed during testing which indicates that RBF has memorized the data and therefore were not able to generalize. Among various ANN structures, GRNN method proved its effectiveness on processing this high-dimensional problem with high accuracy in short computation time. GRNN based design provides the channel lengths and widths of all transistors which satisfy the design constraints (*DNL*, *INL*, monotonicity and gain error) that are also the inputs of the neural network.

The success of the validation with TSMC 0.35μm test set resulted that the proposed method can redesign the DAC with a new technology using the knowledge of previously designed DAC simulation results for older technology. Moreover, the proposed method not only improved monotonicity but also reduced *DNL*, *INL* and gain error values within some limits. As a result, it was proved that the ANN based design approach proposed here provides the channel widths and lengths of all transistors in a newer technology when a user determines the static specification parameters, with minimum effort and design knowledge.

Consequently, as a global optimization tool, PSO is a very efficient method for optimal component selection and sizing task in electronic circuit design automation by means of both high accuracy and short computation time. Since once programmed, no human intervention is required (e.g. to provide an initial “good design” or to interactively guide the optimization process), the proposed method yields completely automated sizing of optimal circuits by means of both discrete and integrated design concepts.

Utilizing PSO as a training algorithm for ANN is another work in this dissertation. Accurate results are obtained for EXOR classification when MLP is trained with PSO with a particle vector dimension of 15x20. However when particle vector size is increased to thousands for technology independent design problem, PSO algorithm did not succeed as a training algorithm of MLP. Although training is completed unlike with BP and LM, it did not converge to the desired MSE values.

As a further work, performance of PSO could be enhanced by utilizing more accurate transistor model equations, such as EKV modeling for electronic design automation considering a particular manufactured technology. Moreover, technology independent design automation of integrated circuits is also a very open research area. Variants of PSO could be utilized as a training algorithm of ANN for average dimensional technology independent design automation problems. Considering very-high dimensional problems; apart from training task, PSO could be utilized in a different configuration for hybridization with ANN structures having iterative nature.

REFERENCES

Aarts, E. and Lenstra, K., (2003), Local Search in Combinatorial Optimization, Princeton University Press.

Abebe, H. and Tyree, V.C., (2000), "BSIM3v3.1 Model Parameters Extraction and Optimization", USC-ISI The MOSIS Service Technical Report, October.

Al-Kazemi, B. and Mohan, C.K., (2002) "Training Feedforward Neural Networks Using Multi-Phase Particle Swarm Optimization", Proceedings of the Ninth International Conference on Neural Information Processing, 5:2615-2619.

Al-Othman, A.K. and Abdelhamid, T.H., (2008), "Elimination of Harmonics in Multilevel Inverters with Non-equal DC Sources Using PSO", 13th Power Electronics and Motion Control Conference (EPE-PEMC 2008), 606-613.

Allen, P. and Holberg D., (2002), CMOS Analog Circuit Design, Oxford University Press, Second Edition, New York.

Amir, A. and Chuanyi J., (1997), "How Initial Conditions Affect Generalization Performance in Large Networks", IEEE Transactions on Neural Networks, 8(2):448-451.

Amoretti M., (2009), "A Framework for Evolutionary Peer-to-Peer Overlay Schemes", European Workshops on the Applications of Evolutionary Computation (EvoWorkshops 2009), April 2009, Tubingen, Germany, 61-70.

Baker, R.J., Li, H.W. and Boyce, D.E., (1997), CMOS Circuit Design Layout and Simulation, Wiley-IEEE Press, USA.

Basseur, M., Talbi, E.G., Nebro, A., and Alba, E., (2006), "Metaheuristics for Multiobjective Combinatorial Optimization Problems: Review and Recent Issues, National Institute of Research in Informatics and Control (INRIA), September 2006, report no:5978.

Bellman, R., (2003), Dynamic Programming, Princeton University Press, Dover Paperback Edition.

Bishop, J.M., (1989), Stochastic Searching Networks, Proceedings of the IEE International Conference on Artificial Neural Networks, 329-331.

Bonabeau, E., Dorigo, M. and Theraulaz, G., (1999), Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press.

Braendler, D. and Hendtlass, T., (2002), "The Suitability of Particle Swarm Optimization for Training Neural Hardware", 15th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, LNAI 2358:190–199.

Braik, M., Sheta, A. and Arieqat, A., (2008), "A Comparison Between GAs and PSO in Training ANN to Model the TE Chemical Process Reactor", Proceedings of the AISB 2008 Symposium on Swarm Intelligence Algorithms and Applications, 11:24-30.

Brayton, R.K., Hatchel, G.D., Sangiovanni-Vincentelli, A.L., (1981), A Survey of Optimization Techniques for Integrated-circuit Design, Proceedings of the IEEE, 69(10):1334-1362.

Chan, F.T.S. and Tiwari, M.K., (2007), Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, I-Tech Education and Publishing, Vienna.

Chang, S.J., Hou, H.S. and Su, Y.K., (2006), "Automated Passive Filter Synthesis Using a Novel Tree Representation and Genetic Programming," IEEE Transactions on Evolutionary Computation, 10(1):93–100.

Choi, K. and Allstot D., (2006), "Parasitic-aware Design and Optimization of a CMOS RF Power Amplifier", IEEE Transactions on Circuits and Systems I, 53(1):16-25.

Clerc, M., (2002), "The Particle Swarm - Explosion, Stability and Convergence in a Multidimensional Complex Space", IEEE Transactions on Evolutionary Computation, 58-73.

Coello, C.A., Luna, E.H. and Aguirre, A.H., (2003), "Use of Particle Swarm Optimization to Design Combinational Logic Circuits", Lecture Notes in Computer Science Springer Verlag 2606:398-409.

Cooren, Y., Fakhfakh M., Loulou, M., and Siarry, P., (2007), "Optimizing Second Generation Current Conveyors Using Particle Swarm Optimization", The IEEE 19th International Conference on Microelectronics (ICM'07), 2007, Cairo, Egypt.

Currit, N., (2002), "Inductive Regression: Overcoming OLS Limitations with the General Regression Neural Network", Computers Environment and Urban Systems, Elsevier Science Press, 26(4):335-353.

Das, A. and Vemuri, R., (2007), "An Automated Passive Analog Circuit Synthesis Framework Using Genetic Algorithms", IEEE Computer Society Annual Symposium on VLSI, 145-152.

DeMassa, T.A. and Ciccone, Z., (1996), Digital Integrated Circuits, John Wiley & Sons, NY.

Dohnal, J., Pivonka, P., (2003) "The Real-time Identification of Dynamic Systems by Using the Neural Net Approach", IEEE Int. Conference on Industrial Technology, 1:99-103.

Dorigo, M., DiCaro, G. and Gambardella, L.M., (1999), "Ant Algorithms for Discrete Optimization", Artificial Life Journal 5:137-172.

Dréo, J., Petrowski, A., Siarry, P. and Taillard, E., (2006), Metaheuristics for Hard Optimization: Methods and Case Studies, Springer Press, France.

Engelbrecht, A.P., (2007), Fundamentals of Computational Swarm Intelligence, John Wiley & Sons, England.

Fakhfakh, M., Loulou, M. and Masmoudi, N., (2009a), "A Novel Heuristic for Multi-objective Optimization of Analog Circuit Performances", Analog Integrated Circuits Signal Processing, 61:47-64.

Fakhfakh, M., Cooren Y., Loulou M., Siarry, P., (2009b), “A Particle Swarm Optimization Technique Used for the Improvement of Analogue Circuit Performances”, Particle Swarm Optimization, Aleksandar Lazinica, 169-182.

Fakhfakh, M., Cooren, Y., Sallem, A., Loulou, M. and Siarry, P., (2010), “Analog Circuit Design Optimization Through the Particle Swarm Optimization Technique”, Analog Integrated Circuits Signal Processing, 63:71-82.

Gielen, G.G.E., Rutenbar, R.A., (2000), “Computer-Aided Design of Analog and Mixed Signal Integrated Circuits”, Proceedings of IEEE, 88(12):1825-1852.

Glover, F., (1989), “Tabu search- part I”, ORSA Journal on Computing, 1(3):190-206.

Glover, F., (1990), “Tabu search- part II”, ORSA Journal on Computing, 2(1):4-32.

Goh, C. and Li, Y., (2001), “GA Automated Design and Synthesis of Analog Circuits with Practical Constraints”, IEEE Congress on Evolutionary Computation, 1(1):170–177.

Goldberg, D.E., (2005), Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley, USA.

Grimbleby, J.B., (2000), “Automatic Analogue Circuit Synthesis Using Genetic Algorithms”, Proceedings of the IEE Circuits, Devices and Systems, December 2000, 147(6):319-323.

Guerra-Gomez, I., Tlelo-Cuautle, E., McConaghy, T. and Gielen, G., (2009), “Decomposition-based multi-objective optimization of second generation current conveyors”, Proceedings of the IEEE Mid-West Symposium on Circuits and Systems, IMWSCS’09, Cancun, Mexico, 220-223.

Gudise, V.G., Venayagamoorthy, G.K., (2004), “FPGA Placement and Routing Using Particle Swarm Optimization”, IEEE Computer Society Ann. Symposium on VLSI, 307-308.

Harvey, J.P., Elmasry, M.I. and Leung, B., (1992), "STAIC: An Interactive Framework for Synthesizing CMOS and BiCMOS Analog Circuits", IEEE Trans. on CAD, 11:1402-1417.

Haykin, S., (1999), Neural Networks: A Comprehensive Foundation, Prentice-Hall, Second edition, New Jersey.

Hershenson, M.D., Boyd, S.P. and Lee, T.H., (2001), “Optimal Design of a CMOS Op-Amp via Geometric Programming”, IEEE Transactions Computer Aided Design Integrated Circuits and Systems, 20:1-21.

Hoeschele, D.F., (1994), Analog-to-Digital and Digital-to-Analog Conversion Techniques, John Wiley & Sons Press Second Edition, USA.

Holland, J., (1975), Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

Horrocks, D.H. and Spittle M.C., (1993),”Component Value Selection for Active Filter Using Genetic Algorithms”, Proceeding IEE/IEEE Workshop on Natural Algorithms in Signal Processing, 1(13):1-6.

Hu, X., Eberhart, R. and Shi, Y., (2003), “Engineering Optimization with Particle Swarm”, IEEE Swarm Intelligence Symposium, Indianapolis, USA, 53-57.

Hu, J., Zhong, X. and Goodman, E., (2005), “Open-Ended ,Robust Design of Analog Filters Using Genetic Programming”, Genetic and Evolutionary Computation Conf., 1619-1626.

Hush, D.R., Horne, B. and Salas, J.M., (1992), “Error Surfaces for Multilayer Perceptrons”, IEEE Transactions on Systems, Man and Cybernetics, 22(5):1152-1161.

Jabri, M.A., Coggins, R.J. and Flower. B.G., (1996), Adaptive Analog VLSI Neural Systems, Chapman & Hall, London.

Jiang, M., Yang, Z. and Gan, Z., (2007), “Optimal Components Selection for Analog Active Filters Using Clonel Selection Algorithm”, LNCS 4681, ICIC (I), 950-959.

Jianjun, X., Yagoub, M.C.E., Runtao, D., Zhang, Q.J., (2003), “Feedforward Dynamic Neural Network Technique for Modeling and Design of Nonlinear Telecommunication Circuits and Systems”, Neural Networks Proceedings of the International Joint Conference, 2:930 935.

Junyou, B., (2007), “Stock Price Forecasting Using PSO-Trained Neural Networks”, IEEE Congress on Evolutionary Computation, 2879-2885.

Kahraman, N., (2008), “Technology Independent Design of Submicron CMOS Circuits with Artificial Neural Networks”, Ph.D. Dissertation, Yildiz Technical University, Graduate School of Natural and Applied Sciences, Istanbul.

Kahraman, N. and Yıldırım, T., (2008a), “Technology Independent Circuit Sizing for Standard Cell Based Design Using Neural Networks”, Digital Signal Processing, 19:708-714.

Kahraman, N. and Yıldırım, T., (2008b) “Technology Independent Circuit Sizing for Fundamental Analog Circuits Using Artificial Neural Networks”, Proc. of PRIME08, 1-4.

Kalinli, A., (2004), “Optimal Circuit Design Using Immune Algorithm”, LNCS 3239, ICARIS 2004, 42-52.

Kalinli, A., (2006), “Component Value Selection for Active Filters Using Parallel Tabu Search Algorithm”, AEU International Journal of Electronics and Communications, 60:85-92.

Kang, S. and Leblebici, Y., (2005), CMOS Digital Integrated Circuits, McGraw Hill, NY.

Karaboga, D., (2005), An Idea Based On Honey Bee Swarm For Numerical Optimization, Technical Report-TR06, Erciyes University, Kayseri.

Karaboga, D. and Basturk, B., (2007), “A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm”, Journal of Global Optimization, Springer Netherlands, 39(3): 459-471.

Karaboga, N. (2009), “A New Design Method Based on Artificial Bee Colony Algorithm for Digital IIR Filters”, Journal of the Franklin Institute, 346: 328-348.

Kennedy, J. and Eberhart, R.C., (1995), “Particle Swarm Optimization”, Proceedings IEEE International Conference on Neural Networks, 1942-1948.

Kester, W., (2005), The Data Conversion Handbook, Analog Devices Inc., 1-17.

Kier, R.J., Harrison, R.R. and Beer, R.D., (2004), “An MDAC Synapse for Analog Neural Networks”, Proc. of ISCAS '04, 5:752-755.

Kiink, T., Vesterstroem, J.S. and Riget, J., (2002), “Particle Swam Optimization with Spatial Particle Extension”, Proc. of the IEEE Congress on Evolutiorian Computation, 1474-1479.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P., (1983), “Optimization by Simulated Annealing”, Journal of Science, 220:671-680.

Koh, H.Y., Sequin C.H. and Gray P.R., (1990), "OPASYN: A Compiler for CMOS Operational Amplifiers", IEEE Transactions Computer-Aided Design, 9(2):113-125.

Koza, J. R, Bennett III, F.H., Andre, D., Keane, M.A. and Dunlap, F., (1997), “Automated Synthesis of Analog Electrical Circuit by Means of Genetic Programming”, IEEE Transactions Evolutionary Computation, 1(2):109-128.

Kruiskamp, W. and Leenaerts, D., (1995), “DARWIN: CMOS Op-Amp Synthesis by Means of a Genetic Algorithm”, Proceedings of 32nd Annual Design Automation Conf., 433–438.

Kuo, C., (2007), “Particle Swarm Trained Neural Network for Fault Diagnosis of Transformers by Acoustic Emission”, ICIC (2) 2007, LNAI, 4682:992–1003.

Lacanette, K., (2010), “A Basic Introduction to Filters-Active, Passive and Switched-Capacitor”, National Semiconductor Application Note, 779.

Land, A.H. and Doig, A.G., (1960), “An Automatic Method for Solving Discrete Programming Problem”, Econometrica, 28:497-520.

Liu, B., Fernandez, F., Gielen, G., Lopez, R. and Roca, E., (2009), “A Memetic Approach to the Automatic Design of High-Performance Analog Integrated Circuits”, ACM Transactions on Design Automation of Electronic Systems, 14(3):1-24.

Luitel, B. and Venayagamoorthy, G.K., (2008), “Differential Evolution Particle Swarm Optimization for Digital Filter Design”, IEEE Conference on Evolutionary Computation, 3954-3961.

Mandal, S.K., Sural, S. and Patra, A., (2008), “ANN and PSO Based Synthesis of On-Chip Spiral Inductors for RF ICs”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 27(1):188-192.

Masara, R.E., (1991), Optimization Methods of Electronic Circuit Design, Longman Scientific&Technical, London.

Massier, T., Graeb, H. and Schlichtmann, U., (2008), “The Sizing Rules Method for CMOS and Bipolar Analog Integrated Circuit Synthesis”, IEEE Transactions Computer-Aided Design of Integrated Circuits Systems, 27: 2209-2222.

Maten, E.J.W., Heijmen, T.G.A, Lin C. and Guennouni, A.J., (2006), “Optimization of Electronic Circuits”, SIMAI-2006, Annual Meeting of Societá Italiana di Matematica Applicata e Industriale, Baia Samuele (Ragusa), Sicily.

Maulik, P.C., Carley L.R. and Rutenbar R.A., (1995), "Integer Programming Based Topology Selection of Cell-Level Analog Circuits," IEEE Trans. on CAD, 14(4):401-412.

Medeiro, F., Rodríguez-Macías, R., Fernández, F.V., Domínguez-astro, R., Huertas, J.L., Rodríguez-Vázquez, A., (1994), “A Global Design of Analog Cells Using Statistical Optimization Techniques”, Analog Integrated Circuits and Signal Processing, 6(3):179-195.

Mendes, R., Cortez, P., Rocha, M., and Neves, J., (2002), “Particle Swarms for Feedforward Neural Network Training”, Proc. of the Int. Joint Conf. on Neural Networks, 1895-1899.

Moo Moore, P. and Venayagamoorty, G.K., (2005), “Evolving Combinational Logic Circuits Using a Hybrid Quantum Evolution and Particle Swarm Inspired Algorithm”, NASA/DoD Conference on Evolvable Hardware, 97-102.

Moore, P. and Venayagamoorty, G.K., (2006), “Evolving Digital Circuits Using Particle Swarm, Optimization and Differential Evolution”, International Journal of Neural Systems, 16(3):163-177.

Nelder, J.A. and Mead, R., (1965), “A Simplex Method for Function Optimization”, Computer journal, 7:308-313.

Paarman, L.D., (2007), Design and Analysis of Analog Filters, Kluwer Academic Publishers.

Pratap, R.J., (2005), “Design and Optimization of Microwave Circuits and Systems Using Artificial Intelligence Techniques”, Ph.D dissertation, Georgia Institute of Technology.

Rabaey, J.M., Chandrakasan, A. and Nikolic, B., (2003), Digital Integrated Circuits: A Design Perspective, Prentice Hall, Second edition, New Jersey.

Razavi, B. (1999), “Principles of Data Conversion System Design”, IEEE Press, New York.

Salem, S.B., Fakhfakh, M., Masmoudi, D.S, Loulou, M., Loumeau, P. and Masmoudi, N., (2006), “A High Performances CMOS CCII and High Frequency Applications”, Analog Integrated Circuits and Signal Processing, 49(1):71-78.

Sapargaliyev, Y. and Kalganova, T., (2006), “Constrained and Unconstrained Evolution of LCR Lowpass Filters with Oscillating Length Representation”, IEEE Congress on Evolutionary Computation, Canada, 1529-1536.

Sapatnekar, S., Rao, V.B., Vaidya, P. and Kang, S.M., (1993), “An Exact Solution to the Transistor Sizing Problem for CMOS Circuits Using Convex Optimization,” IEEE Transactions Computer-Aided Design, 12:1621–1634.

Schalkoff, R.J., (1997), Artificial Neural Networks, McGraw-Hill, Singapore.

Schaumann, R. and Valkenburg, M.V., (2001), Design of Analog Filters, Oxford University Press, New York.

Sheta, A.F., (2010), “Analogue Filter Design Using Differential Evolution”, International Journal of Bio-Inspired Computation, 2(3/4):233-241.

Scniederjans, M.J., (1995), Goal Programming Methodology and Applications, Kluwer Publishers, Boston.

Serin, E., (2008), 4-bit Digital to Analog Converter Design, B.Sc Thesis, Yildiz Technical University, Department of Electronics and Communication Engineering, Istanbul.

Sherwani, N., (1999), Algorithms for VLSI Physical Design Automation, Kluwer Academic Press, Third Edition, Dordrecht.

Siarry, P., Berthiau, G., Durdin, F. and Haussy, J., (1997), “Enhanced Simulated Annealing for Globally Minimizing Functions of Many-Continuous Variables”, ACM Transactions on Mathematical Software, 23:209-228.

Siarry, P., Michalewicz, Z. (Eds), (2007), Advances in Metaheuristics for Hard Optimization, Springer, Berlin.

Specht, D.F., (1991a), “General Regression Neural Network”, IEEE Transactions Neural Networks, 2:568–576.

Specht D.F., (1991b), “Enhancements to Probabilistic Neural Network” Proceedings of International Joint Conference on Neural Network, 1:761–768.

Sripramong, T. and Toumazou, C. (2002), "The Invention of CMOS Amplifiers Using Genetic Programming and Current-Flow Analysis", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 21(11):1237-1252.

Talbi, E.G., (2002), “A Taxonomy of Hybrid Metaheuristics”, Journal of Heuristics, 541-564.

Tawdross, P. and König, A., (2005), “Investigation of Particle Swarm Optimization for Dynamic Reconfiguration of Field-Programmable Analog Circuits”, Proceedings of 5th Conference on Hybrid Intelligent Systems HIS´05, 259-264.

Tawdross, P. and König, A., (2006), “Particle Swarm Optimization for Reconfigurable Sensor Electronics – Case Study: 3 Bit Flash ADC”, 4th IEEE International Workshop on Intelligent Solutions in Embedded Systems (WISES '06), 117-126.

Thakker, R.A., Baghini, M.S. and Patil, M.B., (2009), "Low-Power Low-Voltage Analog Circuit Design Using Hierarchical Particle Swarm Optimization", IEEE 22nd International Conference on VLSI Design, 427-432.

Tlelo-Cuautle, E., Guerra-Gomez, I., Duarte-Villasenor M.A.,. Fraga L.G., Flores-Becerra, G., Reyes-Salgado, G., Reyes-Garcia, C.A. and Rodriquez-Gomez, G., (2010), “Applications of Evolutionary Algorithms in the Design Automation of Analog Integrated Circuits”, Journal of Applied Sciences, 10: 1859-1872.

Tulunay, G. and Balkir, S., (2004), “A Compact Optimization Methodology for Single Ended LNA”, Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS 2004, 273-276.

Tulunay, G. and Balkir, S., (2008), “A Synthesis Tool for CMOS RF Low Noise Amplifiers”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 21(11).

Ulker, S., (2008), “Particle Swarm Optimization Applications to Microwave Circuits”, Microwave and Optical Technology Letters, 50(5):1333-1336.

Venayagamoorthy, G.K., Smith S.C. and Singhal, G., (2007), “Particle Swarm Based Optimal Partitioning Algorithm for Combinational CMOS Circuits”, Engineering Applications of Artificial Intelligence, 177-184.

Vilovic, I., Burum, N. and Milic, D., (2009), “Using Particle Swarm Optimization in Training Neural Network for Indoor Field Strength Prediction”, 51st International Symposium ELMAR-2009, 275–278.

Vittoz, E.A. and Arreguit, X.i, (1993), “Linear Networks Based on Transistors”, Electronic Letters, 20:297–299.

Vural, R.A. and Yildirim, T., (2008), “Optimization of Integrated Circuits Using an Artificial Intelligence Algorithm”, Proceedings of PRIME08, 13-15.

Vural, R.A. and Yildirim, T., (2010a), “Component Value Selection for Analog Active Filter Using Particle Swarm Optimization”, The 2nd International Conference on Computer and Automation Engineering (ICCAE 2010), 1:25-28.

Vural, R.A., and Yildirim, T., (2010b), “State Variable Filter Design Using Particle Swarm Optimization”, XIth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD 2010).

Vural, R.A, Der, O., Yildirim, T., (2010c), "Particle Swarm Optimization Based Inverter Design Considering Transient Performance" Digital Signal Processing, Elsevier, 20(4):1215-1220.

Vural, R.A, Der, O., Yildirim, T., (2010d), "Investigation of Particle Swarm Optimization for Switching Characterization of Inverter Design" Expert Systems with Applications, Elsevier, 38(5): 5696-5703.

Vural, R.A., Yildirim, T., Kadioglu, T. and Basargan, A., (Article in Press), “Performance Evaluation of Evolutionary Algorithms for Optimal Filter Design” IEEE Transactions on Evolutionary Computation, doi:10.1109/TEVC.2011.2112664

Wang, L., Fukatsu, Y., Watanabe, K., (2001), “A CMOS R-2R Ladder Digital-to-Analog Converter and Its Characterization”, Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference IMTC, 2:1026–1031.

Wang, W., Lu, Y., Fu, J. and Xiong, Y., (2005), “Particle Swarm Optimization and Finite-Element Based Approach for Microwave Filter Design”, IEEE Transactions Magnetics, 41(5):1800–1803.

Wen, Y. and Lee, K., (2002), “A Current-Mode BIST Structure of DACs Measurement”, Elsevier Science Ltd., 31(3):147-163.

Wolfe, G. And Vemuri, R., (2003), “Extraction and Use of Neural Network Models in Automated Synthesis of Operational Amplifiers”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 22(2):198–212.

Xie, Y. and Jabri, M., (1992), “Analysis of Effects of Quantisation in Multilayer Neural Networks Using Statistical Models”, IEEE Transactions Neural Networks, 3(2):334-338.

Xu H. and Ding, Y., (2009), “Optimizing Method for Analog Circuit Design Using Adaptive Immune Genetic Algorithm”, International Conference on Frontier of Computer Science and Technology, 359-363.

Zebulum, R.S., Pacheco, M.A. and Vellasco, M., (1998), “Comparison of Different Evolutionary Methodologies Applied to Electronic Filter Design”, IEEE Conference on Evolutionary Computation, 434-439.

Zebulum, R.S., Pacheco, M.A. and Vellasco, M., (1999), “Artificial Evolution of Active Filters: A Case Study”, First NASA DoD Workshop on Evolvable Hardware, 66-75.

Zhang, C., Shao, H. and Li, Y., (2000), “Particle Swarm Optimization for Evolving Artificial Neural Network”, Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 2487-2490.

Zhao, F., Ren, Z., Yu, D. and Yang, Y., (2005), “Application of An Improved Particle Swarm Optimization Algorithm for Neural Network Training”, International Conference on Neural Networks and Brain, ICNN&B '05, 13-15 Oct. 2005, 3:1693-1698.

Zhongliang, P., (2004), "Neural Network Model for Testing Stuck-at and Delay Faults in Digital Circuit", Proceedings of 17th International Conference on VLSI Design, 499-504.

Zumbahlen, H., (2008), Linear Circuit Design Handbook Analog Devices Inc., Newnes Elsevier.

Zurada, J.M., (1995), Introduction to Artificial Neural Systems, PWS, Boston, MA.

Internet References

[1] http://www.learnartificialneuralnetworks.com

[2] http://www.dtreg.com/mlfn.htm

[3] http://www.swarmintelligence.org/tutorials.php

[4] http://mathworld.wolfram.com/XOR.html

[5] http://www.analog.com/static/imported-files/tutorials/MT-043.pdf

APPENDIXES

Appendix 1 TSMC 0.25μm Technology Model Parameters

MOSIS PARAMETRIC TEST RESULTS

RUN: N99Y VENDOR: TSMC
TECHNOLOGY: SCN025 FEATURE SIZE: 0.25 microns

COMMENTS: TSMC 025SPPM.

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	0.36/0.24			
Vth		0.56	-0.53	Volts
SHORT	20.0/0.24			
Idss		590	-263	uA/um
Vth		0.60	-0.59	Volts
Vpt		7.6	-7.2	Volts
WIDE	20.0/0.24			
Ids0		13.1	-1.7	pA/um
LARGE	20.0/20.0			
Vth		0.52	-0.63	Volts
Vjbkd		6.1	-7.0	Volts
Ijlk		-22.9	-7.5	pA
Gamma				V^0.5
K' (Uo*Cox/2)		109.7	-25.5	uA/V^2

COMMENTS: Poly bias varies with design technology. To account for mask and etch bias use the appropriate value for the parameter XL in your SPICE model card.

Design Technology	XL
SCN5M_DEEP (lambda=0.12)	0.03
thick oxide, NMOS	0.02
thick oxide, PMOS	-0.03
TSMC25	0.03
thick oxide, NMOS	0.03
thick oxide, PMOS	0.03
SCN3M_SUBM (lambda=0.15)	-0.03
thick oxide, NMOS	0.02
thick oxide, PMOS	-0.03

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>15.0	<-15.0	Volts

PROCESS PARAMETERS	N+ACTV	P+ACTV	POLY	MTL1	MTL2	MTL3	MTL4	UNITS
Sheet Resistance	4.7	3.4	4.1	0.08	0.07	0.07	0.07	ohms/sq
Width Variation (measured - drawn)	0.09	0.14	0.12	0.08	0.03	0.01	-0.03	microns
Contact Resistance	6.9	6.0	5.8		2.04	4.06	5.72	ohms
Gate Oxide Thickness	57	angstrom						

PROCESS PARAMETERS	MTL5	N_WELL	UNITS
Sheet Resistance	0.03	1214	ohms/sq
Width Variation (measured - drawn)	0.08		microns

```
 Contact Resistance           7.52            ohms
CAPACITANCE PARAMETERS  N+ACTV  P+ACTV  POLY MTL1 MTL2 MTL3 MTL4 MTL5
N_WELL UNITS
 Area (substrate)      1727    1888      97   37   19   13    8    8    63
aF/um^2
 Area (N+active)                       6042   50   20   14   11    9
aF/um^2
 Area (P+active)                       5796
aF/um^2
 Area (poly)                                61   18   10    7    6
aF/um^2
 Area (metal1)                                   39   15    9    7
aF/um^2
 Area (metal2)                                        37   15    9
aF/um^2
 Area (metal3)                                             38   15
aF/um^2
 Area (metal4)                                                  38
aF/um^2
 Fringe (substrate)     417     317           21   57   54   51   24
aF/um
 Fringe (poly)                                67   39   29   24   21
aF/um
 Fringe (metal1)                                 49   33   27   24
aF/um
 Fringe (metal2)                                      53   34   29
aF/um
 Fringe (metal3)                                           53   35
aF/um
 Fringe (metal4)                                                59
aF/um

CIRCUIT PARAMETERS                            UNITS
 Inverters                    K
  Vinv                      1.0        1.05  Volts
  Vinv                      1.5        1.13  Volts
  Vol (100 uA)              2.0        0.22  Volts
  Voh (100 uA)              2.0        2.07  Volts
  Vinv                      2.0        1.19  Volts
  Gain                      2.0      -16.66
 Ring Oscillator Freq.
  DIV1024_T (31-stage,2.5)             168.69  MHz
  DIV1024 (31-stage,2.5)               299.76  MHz
 Ring Oscillator Power
  DIV1024_T (31-stage,2.5)               0.06  uW/MHz/g
  DIV1024 (31-stage,2.5)                 0.06  uW/MHz/g

COMMENTS: DEEP_SUBMICRON
 N99Y SPICE BSIM3 VERSION 3.1 PARAMETERS
SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8

* DATE: Dec  6/99
* LOT: n99y                  WAF: 10
* Temperature_parameters=Default
.MODEL CMOSN NMOS (
+LEVEL   = 49             acm     = 3              hdif    = 0.35e-6
+VERSION = 3.1            TNOM    = 27             TOX     = 5.7E-9
+XJ      = 1E-7           NCH     = 2.3549E17      VTH0    = 0.4365497
+K1      = 0.3915623      K2      = 0.0175145      K3      = 1E-3
+K3B     = 2.6588343      W0      = 1E-7           NLX     = 1.111465E-7
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = -0.0408321     DVT1    = 0.0746768      DVT2    = 0.307109
```

```
+U0      = 407.1177485    UA      = 9.442714E-11   UB      = 1.092986E-18
+UC      = 1.63196E-11    VSAT    = 1.365087E5     A0      = 1.3189329
+AGS     = 0.2711719      B0      = 3.291713E-8    B1      = -1E-7
+KETA    = 4.645753E-3    A1      = 0              A2      = 1
+RDSW    = 439.9558234    PRWG    = 0.0345487      PRWB    = -0.0441065
+WR      = 1              WINT    = 1.645705E-9    LINT    = 1.116516E-9
+XL      = 3E-8           XW      = 0              DWG     = -1.494138E-9
+DWB     = 1.459097E-8    VOFF    = -0.1026054     NFACTOR = 0.1344887
+CIT     = 0              CDSC    = 1.527511E-3    CDSCD   = 0
+CDSCB   = 0              ETA0    = 1.930311E-3    ETAB    = 2.946158E-4
+DSUB    = 0.0214865      PCLM    = 1.3387947      PDIBLC1 = 0.480652
+PDIBLC2 = 9.034986E-3    PDIBLCB = -1E-3          DROUT   = 0.5593223
+PSCBE1  = 9.843289E9     PSCBE2  = 2.10878E-9     PVAG    = 1.0033136
+DELTA   = 0.01           MOBMOD  = 1              PRT     = 0
+UTE     = -1.5           KT1     = -0.11          KT1L    = 0
+KT2     = 0.022          UA1     = 4.31E-9        UB1     = -7.61E-18
+UC1     = -5.6E-11       AT      = 3.3E4          WL      = 0
+WLN     = 1              WW      = -1.22182E-16   WWN     = 1.2127
+WWL     = 0              LL      = 0              LLN     = 1
+LW      = 0              LWN     = 1              LWL     = 0
+CAPMOD  = 2              XPART   = 0.4            CGDO    = 3.11E-10
+CGSO    = 3.11E-10       CGBO    = 1E-11          CJ      = 1.758521E-3
+PB      = 0.99           MJ      = 0.457547       CJSW    = 4.085057E-10
+PBSW    = 0.8507757      MJSW    = 0.3374073      PVTH0   = 7.147521E-5
+PRDSW   = -67.2161633    PK2     = -1.344599E-3   WKETA   = 3.035972E-3
+LKETA   = -9.0406E-3     LAGS    = -0.3012         )
*
.MODEL CMOSP PMOS (
+LEVEL   = 49             acm     = 3              hdif    = 0.35e-6
+VERSION = 3.1            TNOM    = 27             TOX     = 5.7E-9
+XJ      = 1E-7           NCH     = 4.1589E17      VTH0    = -0.6586391
+K1      = 0.5199897      K2      = 0.0357513      K3      = 0
+K3B     = 15.5613889     W0      = 1E-6           NLX     = 1E-9
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 2.6100181      DVT1    = 0.4363142      DVT2    = -0.042436
+U0      = 196.024903     UA      = 2.767112E-9    UB      = 1.90709E-18
+UC      = 6.166867E-11   VSAT    = 1.975064E5     A0      = 0.2398712
+AGS     = 0.0943234      B0      = 3.21184E-6     B1      = 5E-6
+KETA    = 0.0312217      A1      = 0              A2      = 1
+RDSW    = 997.072701     PRWG    = -0.1916111     PRWB    = -0.495
+WR      = 1              WINT    = 2.527293E-9    LINT    = 1.254514E-8
+XL      = 3E-8           XW      = 0              DWG     = -3.253948E-8
+DWB     = 4.92072E-8     VOFF    = -0.15          NFACTOR = 1.5460516
+CIT     = 0              CDSC    = 1.413317E-4    CDSCD   = 0
+CDSCB   = 0              ETA0    = 0.7241245      ETAB    = -0.240523
+DSUB    = 1.0813613      PCLM    = 2.0772083      PDIBLC1 = 4.31459E-4
+PDIBLC2 = 0.0252121      PDIBLCB = -9.960722E-4   DROUT   = 0.0432774
+PSCBE1  = 3.191047E10    PSCBE2  = 1.323218E-8    PVAG    = 0.0420525
+DELTA   = 0.01           MOBMOD  = 1              PRT     = 0
+UTE     = -1.5           KT1     = -0.11          KT1L    = 0
+KT2     = 0.022          UA1     = 4.31E-9        UB1     = -7.61E-18
+UC1     = -5.6E-11       AT      = 3.3E4          WL      = 0
+WLN     = 1              WW      = 0              WWN     = 1
+WWL     = 0              LL      = 0              LLN     = 1
+LW      = 0              LWN     = 1              LWL     = 0
+CAPMOD  = 2              XPART   = 0.4            CGDO    = 2.68E-10
+CGSO    = 2.68E-10       CGBO    = 1E-11          CJ      = 1.902493E-3
+PB      = 0.9810285      MJ      = 0.4644362      CJSW    = 3.142741E-10
+PBSW    = 0.9048624      MJSW    = 0.3304452      PVTH0   = 4.952976E-3
+PRDSW   = 29.8169373     PK2     = 3.383373E-3    WKETA   = -7.913501E-3
+LKETA   = -0.0208318      )
```

Appendix 2 TSMC 0.35µm Technology Model Parameters

MOSIS WAFER ACCEPTANCE TESTS
RUN: T6BC (MM_NON-EPI) VENDOR: TSMC
TECHNOLOGY: SCN035 FEATURE SIZE: 0.35 microns

COMMENTS: TSMC 035

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	0.6/0.4			
Vth		0.56	-0.73	volts
SHORT	20.0/0.4			
Idss		518	-246	uA/um
Vth		0.59	-0.71	volts
Vpt		6.5	-6.2	volts
WIDE	20.0/0.4			
Ids0		< 2.5	< 2.5	pA/um
LARGE	50/50			
Vth		0.52	-0.71	volts
Vjbkd		8.7	-8.5	volts
Ijlk		<50.0	<50.0	pA
Gamma		0.60	0.37	V^0.5
K' (Uo*Cox/2)		90.6	-32.9	uA/V^2
Low-field Mobility		409.31	148.63	cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask bias use the appropriate value for the parameter XL in your SPICE model card.

Design Technology	XL (um)	XW (um)
SCMOS_SUBM (lambda=0.20)	-0.05	0.15
thick oxide	-0.10	0.15
SCMOS (lambda=0.25)	-0.15	0.15
thick oxide	-0.25	0.15

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>10.0	<-10.0	volts

PROCESS PARAMETERS	N+	P+	POLY	POLY2	M1	M2	M3	UNITS
Sheet Resistance	80.4	154.6	8.5	46.8	0.07	0.07	0.07	ohms/sq
Contact Resistance	66.9	128.2	6.9	37.6		1.27	1.09	ohms
Gate Oxide Thickness	78							angstrom

PROCESS PARAMETERS	M4	POLY2_ME	N\PLY	N_W	UNITS
Sheet Resistance	0.04		1045	1002	ohms/sq
Contact Resistance	1.11				ohms

COMMENTS: N\POLY is N-well under polysilicon.

CAPACITANCE PARAMETERS	N+	P+	POLY	POLY2	M1	M2	M3	M4	N_W	UNITS
Area (substrate)	929	1411	109		26	12	7	6	111	aF/um^2
Area (N+active)			4449		35	16	11	9		aF/um^2
Area (P+active)			4514							aF/um^2
Area (poly)				905	48	15	9	7		aF/um^2
Area (poly2)					46					aF/um^2
Area (metal1)						35	14	8		aF/um^2
Area (metal2)							35	13		aF/um^2
Area (metal3)								33		aF/um^2
Fringe (substrate)	306	337			44	32	23	13		aF/um
Fringe (poly)					62	37	28	23		aF/um
Fringe (metal1)						54	36	26		aF/um
Fringe (metal2)							53	34		aF/um
Fringe (metal3)								51		aF/um
Overlap (N+active)			311							aF/um
Overlap (P+active)			408							aF/um

CIRCUIT PARAMETERS	K		UNITS
Inverters			
Vinv	1.0	1.24	volts
Vinv	1.5	1.38	volts
Vol (100 uA)	2.0	0.21	volts
Voh (100 uA)	2.0	2.93	volts
Vinv	2.0	1.48	volts
Gain	2.0	-16.72	
Ring Oscillator Freq.			
DIV256 (31-stg,3.3V)		188.09	MHz
D256_THK (31-stg,5.0V)		121.01	MHz
Ring Oscillator Power			
DIV256 (31-stg,3.3V)		0.15	uW/MHz/gate
D256_THK (31-stg,5.0V)		0.31	uW/MHz/gate

COMMENTS: SUBMICRON
T6BC SPICE BSIM3 VERSION 3.1 PARAMETERS
SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8

```
* DATE: Jan 18/07
* LOT: T6BC                  WAF: 5004
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 7.8E-9
+XJ      = 1E-7           NCH     = 2.2E17         VTH0    = 0.4761483
+K1      = 0.6147836      K2      = 2.867694E-4    K3      = 100
+K3B     = -10            W0      = 2.600765E-5    NLX     = 2.25701E-7
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 4.2158797      DVT1    = 0.6077768      DVT2    = -0.048051
+U0      = 358.2644557    UA      = -7.65556E-10   UB      = 2.054672E-18
+UC      = 1.641937E-11   VSAT    = 1.374794E5     A0      = 1.2164483
+AGS     = 0.1809739      B0      = 8.878483E-7    B1      = 5E-6
+KETA    = 8.10016E-4     A1      = 3.993837E-4    A2      = 0.4923356
+RDSW    = 544.6941443    PRWG    = 0.1796386      PRWB    = -0.0911284
+WR      = 1              WINT    = 1.354508E-7    LINT    = 1.454983E-9
+XL      = -5E-8          XW      = 1.5E-7         DWG     = -7.29699E-10
+DWB     = 7.151736E-9    VOFF    = -0.0672255     NFACTOR = 1.2933668
+CIT     = 0              CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0              ETA0    = 0.1063985      ETAB    = -0.0141482
+DSUB    = 0.4931007      PCLM    = 2.6057248      PDIBLC1 = 1.271824E-3
+PDIBLC2 = 2.180112E-3    PDIBLCB = 0.1            DROUT   = 4.431661E-4
+PSCBE1  = 7.320236E8     PSCBE2  = 1E-3           PVAG    = 9.330174E-3
```

```
+DELTA   = 0.01           RSH     = 80.4           MOBMOD  = 1
+PRT     = 0              UTE     = -1.5           KT1     = -0.11
+KT1L    = 0              KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0              WLN     = 1              WW      = 0
+WWN     = 1              WWL     = 0              LL      = 0
+LLN     = 1              LW      = 0              LWN     = 1
+LWL     = 0              CAPMOD  = 2              XPART   = 0.5
+CGDO    = 3.11E-10       CGSO    = 3.11E-10       CGBO    = 1E-12
+CJ      = 9.16922E-4     PB      = 0.8            MJ      = 0.3478704
+CJSW    = 3.291571E-10   PBSW    = 0.8            MJSW    = 0.1145248
+CJSWG   = 1.82E-10       PBSWG   = 0.8            MJSWG   = 0.1145248
+CF      = 0              PVTH0   = -0.0139075     PRDSW   = -47.7936722
+PK2     = 2.359942E-3    WKETA   = 6.320378E-4    LKETA   = -4.325395E-3)

.MODEL CMOSP PMOS (                                LEVEL   = 49
+VERSION = 3.1           TNOM    = 27             TOX     = 7.8E-9
+XJ      = 1E-7          NCH     = 8.52E16        VTH0    = -0.6513775
+K1      = 0.4286847     K2      = -5.542793E-3   K3      = 97.1641761
+K3B     = -5            W0      = 7.054287E-6    NLX     = 2.516648E-7
+DVT0W   = 0             DVT1W   = 0              DVT2W   = 0
+DVT0    = 1.7876132     DVT1    = 0.5369794      DVT2    = 3.747654E-3
+U0      = 149.2330772   UA      = 1E-10          UB      = 1.535287E-18
+UC      = -2.41558E-11  VSAT    = 1.540022E5     A0      = 0.7990549
+AGS     = 0.2670643     B0      = 2.609264E-6    B1      = 5E-6
+KETA    = -3.612338E-3  A1      = 2.72138E-4     A2      = 0.9725641
+RDSW    = 3.156499E3    PRWG    = -0.1013163     PRWB    = 0.0728675
+WR      = 1             WINT    = 1.342653E-7    LINT    = 0
+XL      = -5E-8         XW      = 1.5E-7         DWG     = -9.772747E-9
+DWB     = 1.491485E-8   VOFF    = -0.122767      NFACTOR = 1.9970207
+CIT     = 0             CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0             ETA0    = 0.099556       ETAB    = 2.606693E-3
+DSUB    = 0.6275991     PCLM    = 5.5082963      PDIBLC1 = 9.987419E-4
+PDIBLC2 = 0.0117981     PDIBLCB = 0.1            DROUT   = 3.141427E-6
+PSCBE1  = 8E10          PSCBE2  = 5.027754E-10   PVAG    = 9.6519816
+DELTA   = 0.01          RSH     = 154.6          MOBMOD  = 1
+PRT     = 0             UTE     = -1.5           KT1     = -0.11
+KT1L    = 0             KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18     UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0             WLN     = 1              WW      = 0
+WWN     = 1             WWL     = 0              LL      = 0
+LLN     = 1             LW      = 0              LWN     = 1
+LWL     = 0             CAPMOD  = 2              XPART   = 0.5
+CGDO    = 4.08E-10      CGSO    = 4.08E-10       CGBO    = 1E-12
+CJ      = 1.413322E-3   PB      = 0.99           MJ      = 0.5609267
+CJSW    = 2.978864E-10  PBSW    = 0.99           MJSW    = 0.3817474
+CJSWG   = 4.42E-11      PBSWG   = 0.99           MJSWG   = 0.3817474
+CF      = 0             PVTH0   = 7.800191E-3    PRDSW   = 29.6696742
+PK2     = 2.25328E-3    WKETA   = -4.072642E-5   LKETA   = -6.119295E-3)
```

Appendix 3 ON SEMI 0.5µm Technology Model Parameters

MOSIS PARAMETRIC TEST RESULTS

RUN: T15M VENDOR: AMI
TECHNOLOGY: SCN05 EATURE SIZE: 0.5 microns

INTRODUCTION:
This report contains the lot average results obtained by MOSIS from measurements of MOSIS test structures on each wafer of this fabrication lot. SPICE parameters obtained from similar measurements on a selected wafer are also attached.

COMMENTS: American Microsystems, Inc. C5N

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	3.0/0.6			
Vth		0.81	-0.92	volts
SHORT	20.0/0.6			
Idss		465	-257	uA/um
Vth		0.70	-0.90	volts
Vpt		10.0	-10.0	volts
WIDE	20.0/0.6			
Ids0		< 2.5	< 2.5	pA/um
LARGE	50/50			
Vth		0.72	-0.94	volts
Vjbkd		11.6	-11.6	volts
Ijlk		<50.0	<50.0	pA
Gamma		0.47	0.57	V^0.5
K' (Uo*Cox/2)		58.6	-19.3	uA/V^2
Low-field Mobility		471.78	155.38	cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask and etch bias use the appropriate value for the parameter XL in your SPICE model card.

Design Technology	XL
SCN_SUBM (lambda=0.30)	0.00
AMI_C5	0.00
SCN (lambda=0.35)	-0.10

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>15.0	<-15.0	volts

PROCESS PARAMETERS	N+ACTV	P+ACTV	POLY	PLY2_HR	POLY2	MTL1	MTL2	UNITS
Sheet Resistance	80.1	104.1	21.6	1097	41.1	0.08	0.09	ohms/sq
Contact Resistance	60.9	143.2	15.8		27.3		0.79	ohms

Gate Oxide Thickness 139 angstrom

PROCESS PARAMETERS	MTL3	N\PLY	N_WELL	UNITS
Sheet Resistance	0.05	831	828	ohms/sq
Contact Resistance	0.76			ohms

COMMENTS: N\POLY is N-well under polysilicon.

CAPACITANCE PARAMETERS	N+ACTV	P+ACTV	POLY	POLY2	M1	M2	M3	N_WELL	UNITS
Area (substrate)	428	731	88		32	16	10	41	aF/um^2
Area (N+active)			2491		36	16	12		aF/um^2
Area (P+active)			2425						aF/um^2
Area (poly)				881	52	16	9		aF/um^2
Area (poly2)					47				aF/um^2
Area (metal1)						32	13		aF/um^2
Area (metal2)							36		aF/um^2
Fringe (substrate)	322	262			76	59	40		aF/um
Fringe (poly)					61	38	28		aF/um
Fringe (metal1)						53	34		aF/um
Fringe (metal2)							51		aF/um
Overlap (N+active)			207						aF/um
Overlap (P+active)			238						aF/um

CIRCUIT PARAMETERS			UNITS
Inverters	K		
Vinv	1.0	2.13	volts
Vinv	1.5	2.39	volts
Vol (100 uA)	2.0	0.23	volts
Voh (100 uA)	2.0	4.76	volts
Vinv	2.0	2.57	volts
Gain	2.0	-21.19	
Ring Oscillator Freq.			
DIV256 (31-stg,5.0V)		98.00	MHz
D256_WIDE (31-stg,5.0V)		151.14	MHz
Ring Oscillator Power			
DIV256 (31-stg,5.0V)		0.49	uW/MHz/gate
D256_WIDE (31-stg,5.0V)		1.02	uW/MHz/gate

COMMENTS: SUBMICRON
T15M SPICE BSIM3 VERSION 3.1 PARAMETERS

SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8

```
* DATE: Jul 20/01
* LOT: T15M                  WAF: 0206
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                LEVEL   = 49
+VERSION = 3.1           TNOM    = 27              TOX     = 1.39E-8
+XJ      = 1.5E-7        NCH     = 1.7E17          VTH0    = 0.6516076
+K1      = 0.8896025     K2      = -0.0979155      K3      = 23.8061513
+K3B     = -7.7691025    W0      = 1E-8            NLX     = 1E-9
+DVT0W   = 0             DVT1W   = 0               DVT2W   = 0
+DVT0    = 2.876542      DVT1    = 0.4218664       DVT2    = -0.1397962
+U0      = 451.8826245   UA      = 1E-13           UB      = 1.489875E-18
+UC      = 1.893684E-11  VSAT    = 1.704053E5      A0      = 0.5662277
+AGS     = 0.1198161     B0      = 2.705871E-6     B1      = 5E-6
```

```
+KETA    = -2.301173E-3   A1      = 7.285662E-5    A2      = 0.3586004
+RDSW    = 1.159376E3     PRWG    = 0.0531026      PRWB    = 0.0349044
+WR      = 1              WINT    = 2.360078E-7    LINT    = 2.450767E-8
+XL      = 0              XW      = 0              DWG     = -1.296776E-8
+DWB     = 5.50766E-8     VOFF    = 0              NFACTOR = 0.821639
+CIT     = 0              CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0              ETA0    = 1.688074E-3    ETAB    = -8.785487E-4
+DSUB    = 0.1390103      PCLM    = 2.4002094      PDIBLC1 = -0.0558623
+PDIBLC2 = 2.163004E-3    PDIBLCB = -0.118451      DROUT   = 0.385872
+PSCBE1  = 5.569704E8     PSCBE2  = 5.935496E-5    PVAG    = 0
+DELTA   = 0.01           RSH     = 80.1           MOBMOD  = 1
+PRT     = 0              UTE     = -1.5           KT1     = -0.11
+KT1L    = 0              KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0              WLN     = 1              WW      = 0
+WWN     = 1              WWL     = 0              LL      = 0
+LLN     = 1              LW      = 0              LWN     = 1
+LWL     = 0              CAPMOD  = 2              XPART   = 0.5
+CGDO    = 2.07E-10       CGSO    = 2.07E-10       CGBO    = 1E-9
+CJ      = 4.256515E-4    PB      = 0.99           MJ      = 0.447835
+CJSW    = 3.329281E-10   PBSW    = 0.1            MJSW    = 0.1169342
+CJSWG   = 1.64E-10       PBSWG   = 0.1            MJSWG   = 0.1169342
+CF      = 0              PVTH0   = 0.0661673      PRDSW   = 201.5784264
+PK2     = -0.0327168     WKETA   = -0.0250765     LKETA   = 6.176997E-3
)

.MODEL CMOSP PMOS (                               LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 1.39E-8
+XJ      = 1.5E-7         NCH     = 1.7E17         VTH0    = -0.9259178
+K1      = 0.5493891      K2      = 8.966666E-3    K3      = 8.9116777
+K3B     = -0.5844741     W0      = 1E-8           NLX     = 7.795747E-8
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 2.6496816      DVT1    = 0.5037615      DVT2    = -0.0963638
+U0      = 216.8004604    UA      = 2.933658E-9    UB      = 1E-21
+UC      = -5.60899E-11   VSAT    = 2E5            A0      = 0.8656114
+AGS     = 0.1446194      B0      = 8.79758E-7     B1      = 5E-6
+KETA    = -3.911589E-3   A1      = 0              A2      = 0.3
+RDSW    = 3E3            PRWG    = -0.054537      PRWB    = -0.0379172
+WR      = 1              WINT    = 2.899182E-7    LINT    = 4.581285E-8
+XL      = 0              XW      = 0              DWG     = -1.617949E-8
+DWB     = 2.330863E-8    VOFF    = -0.063762      NFACTOR = 0.9168444
+CIT     = 0              CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0              ETA0    = 0.0228777      ETAB    = -0.112099
+DSUB    = 1              PCLM    = 2.0845927      PDIBLC1 = 0.1016884
+PDIBLC2 = 5.000285E-3    PDIBLCB = -0.0444413     DROUT   = 0.292315
+PSCBE1  = 1.444005E10    PSCBE2  = 1.405429E-9    PVAG    = 0
+DELTA   = 0.01           RSH     = 104.1          MOBMOD  = 1
+PRT     = 0              UTE     = -1.5           KT1     = -0.11
+KT1L    = 0              KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0              WLN     = 1              WW      = 0
+WWN     = 1              WWL     = 0              LL      = 0
+LLN     = 1              LW      = 0              LWN     = 1
+LWL     = 0              CAPMOD  = 2              XPART   = 0.5
+CGDO    = 2.38E-10       CGSO    = 2.38E-10       CGBO    = 1E-9
+CJ      = 7.275007E-4    PB      = 0.9494394      MJ      = 0.4937011
+CJSW    = 2.884359E-10   PBSW    = 0.99           MJSW    = 0.3331605
+CJSWG   = 6.4E-11        PBSWG   = 0.99           MJSWG   = 0.3331605
+CF      = 0              PVTH0   = 5.98016E-3     PRDSW   = 14.8598424
+PK2     = 3.73981E-3     WKETA   = 5.957334E-3    LKETA   = -3.385326E-3)
```

Appendix 4 ON SEMI 1.5 µm Technology Model Parameters

MOSIS WAFER ACCEPTANCE TESTS

RUN: T83T VENDOR: AMIS
TECHNOLOGY: SCN15 FEATURE SIZE: 1.6 microns

INTRODUCTION: This report contains the lot average results obtained by MOSIS from measurements of MOSIS test structures on each wafer of this fabrication lot. SPICE parameters obtained from similar measurements on a selected wafer are also attached.

COMMENTS: SCNA16_AMIS

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	4.0/1.6			
Vth		0.59	-1.03	volts
SHORT	20.0/1.6			
Idss		187	-69	uA/um
Vth		0.55	-0.98	volts
Vpt		10.0	-10.0	volts
WIDE	20.0/1.6			
Ids0		< 2.5	< 2.5	pA/um
LARGE	50/50			
Vth		0.59	-0.93	volts
Vjbkd		16.6	-14.8	volts
Ijlk		<50.0	<50.0	pA
Gamma		0.64	0.48	V^0.5
K' (Uo*Cox/2)		35.4	-11.6	uA/V^2
Low-field Mobility		647.91	212.31	cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask bias use the appropriate value for the parameter XL in your SPICE model card.

Design Technology	XL (um)
SCN (lambda=0.8)	0.00

POLY2 TRANSISTORS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	4.8/3.2			
Vth		0.93	-1.17	volts
SHORT	9.6/3.2			
Vth		0.92	-1.13	volts
LARGE	28.8/28.			
Vth		0.93	-1.12	volts
K' (Uo*Cox/2)		21.1	-6.5	uA/V^2

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>15.0	<-15.0	volts
BIPOLAR PARAMETERS	W/L	NPN		UNITS
2X1	2X1			
Beta		124		
V_early		41.7		volts
Vce,sat		0.2		volts
2X2	2X2			
Beta		125		
V_early		41.4		volts
Vce,sat		0.2		volts
2X4	2X4			
Beta		129		
V_early		41.2		volts
Vce,sat		0.2		volts
2X8	2X8			
Beta		125		
V_early		41.4		volts
Vce,sat		0.2		volts
BVceo		13.5		volts
BVcbo		30.0		volts
BVebo		8.1		volts

PROCESS PARAMETERS	N+	P+	POLY	POLY2	PBASE	M1	M2	UNITS
Sheet Resistance	53.2	77.3	25.9	21.4	2236.0	0.05	0.03	ohms/sq
Contact Resistance	54.3	42.6	24.9	15.5		0.05		ohms
Gate Oxide Thickness	316	angstrom						

PROCESS PARAMETERS	N_W	UNITS
Sheet Resistance	1533	ohms/sq
Contact Resistance		ohms

CAPACITANCE PARAMETERS	N+	P+	POLY	POLY2	M1	M2	N_W	UNITS
Area (substrate)	290	304	37	37	24	16	112	aF/um^2
Area (N+active)			1094	696	52	27		aF/um^2
Area (P+active)			1079	690				aF/um^2
Area (poly)				581	46	23		aF/um^2
Area (poly2)					47	23		aF/um^2
Area (metal1)						38		aF/um^2
Fringe (substrate)	73	157			30	26		aF/um
Fringe (poly)					60	43		aF/um
Fringe (metal1)						55		aF/um
Overlap (N+active)			256					aF/um
Overlap (P+active)			265					aF/um

CIRCUIT PARAMETERS			UNITS
Inverters	K		
Vinv	1.0	1.78	volts
Vinv	1.5	2.06	volts
Vol (100 uA)	2.0	0.42	volts
Voh (100 uA)	2.0	4.31	volts
Vinv	2.0	2.25	volts
Gain	2.0	-15.42	
Ring Oscillator Freq.			
DIV64 (31-stg,5.0V)		40.32	MHz

```
 Ring Oscillator Power
  DIV64 (31-stg,5.0V)                 1.50  uW/MHz/gate

 T83T SPICE LEVEL3 parameters are available for classroom instructional
purposes  but not for actual IC design work.

* DATE: Apr 21/08
* LOT: T83T                  WAF: 9102
* DIE: N_Area_Fring          DEV: N3740/10
* Temp= 27
.MODEL CMOSN NMOS (                                LEVEL   = 3
+ TOX     = 3.16E-8         NSUB    = 1.076635E16    GAMMA   = 0.7683227
+ PHI     = 1               VTO     = 0.6139639      DELTA   = 0.6363466
+ UO      = 540.5166232     ETA     = 7.102441E-4    THETA   = 0.0725637
+ KP      = 7.372278E-5     VMAX    = 2.603588E5     KAPPA   = 0.5
+ RSH     = 22.1568863      NFS     = 5.567912E11    TPG     = 1
+ XJ      = 3E-7            LD      = 8.001696E-15   WD      = 6.432153E-7
+ CGDO    = 2.56E-10        CGSO    = 2.56E-10       CGBO    = 1E-10
+ CJ      = 2.872567E-4     PB      = 0.8226187      MJ      = 0.5
+ CJSW    = 1.141246E-10    MJSW    = 0.05           )
*
* DATE: Apr 21/08
* LOT: T83T                  WAF: 9102
* DIE: P_Area_Fring          DEV: P3740/10
* Temp= 27
.MODEL CMOSP PMOS (                                LEVEL   = 3
+ TOX     = 3.16E-8         NSUB    = 1E17           GAMMA   = 0.4983453
+ PHI     = 0.7             VTO     = -0.9427842     DELTA   = 0.3016627
+ UO      = 101.6231199     ETA     = 1.439377E-4    THETA   = 0.1285575
+ KP      = 2.421504E-5     VMAX    = 1.341191E5     KAPPA   = 50
+ RSH     = 0               NFS     = 5.548753E11    TPG     = -1
+ XJ      = 2E-7            LD      = 1.00095E-14    WD      = 9.558722E-7
+ CGDO    = 2.65E-10        CGSO    = 2.65E-10       CGBO    = 1E-10
+ CJ      = 3.018089E-4     PB      = 0.8            MJ      = 0.4534709
+ CJSW    = 1.586951E-10    MJSW    = 0.0859905      )
*

 T83T SPICE BSIM3 VERSION 3.1 PARAMETERS

SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8

* DATE: Apr 21/08
* LOT: T83T                  WAF: 9102
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                LEVEL   = 49
+VERSION = 3.1             TNOM    = 27             TOX     = 3.16E-8
+XJ      = 3E-7            NCH     = 7.5E16         VTH0    = 0.5532293
+K1      = 0.9035181       K2      = -0.0643307     K3      = 2.5479002
+K3B     = -2.8027479      W0      = 3.208573E-6    NLX     = 2.437748E-8
+DVT0W   = 0               DVT1W   = 0              DVT2W   = 0
+DVT0    = 0.7417824       DVT1    = 0.4992713      DVT2    = -0.4963435
+U0      = 668.7098806     UA      = 1.606939E-9    UB      = 1.161183E-18
+UC      = 2.084488E-11    VSAT    = 1.15642E5      A0      = 0.5520578
+AGS     = 0.0735062       B0      = 2.123316E-6    B1      = 5E-6
+KETA    = -3.240368E-3    A1      = 0              A2      = 1
+RDSW    = 3E3             PRWG    = -0.0295407     PRWB    = -0.0293523
+WR      = 1               WINT    = 6.860525E-7    LINT    = 2.905606E-7
+XL      = 0               XW      = 0              DWG     = -2.112208E-8
+DWB     = 2.909887E-8     VOFF    = -0.0475545     NFACTOR = 0.7389868
+CIT     = 0               CDSC    = 2.230152E-6    CDSCD   = 2.587466E-6
+CDSCB   = 1.055349E-4     ETA0    = -0.0329209     ETAB    = -6.587451E-3
+DSUB    = 5.634216E-3     PCLM    = 0.1186109      PDIBLC1 = 0.0110461
```

```
+PDIBLC2 = 3.644817E-3    PDIBLCB = -0.1          DROUT   = 0.0795384
+PSCBE1  = 2.153596E9     PSCBE2  = 5.005E-10     PVAG    = 0.2222499
+DELTA   = 0.01           RSH     = 53.2          MOBMOD  = 1
+PRT     = 0              UTE     = -1.5          KT1     = -0.11
+KT1L    = 0              KT2     = 0.022         UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11      AT      = 3.3E4
+WL      = 0              WLN     = 1             WW      = 0
+WWN     = 1              WWL     = 0             LL      = 0
+LLN     = 1              LW      = 0             LWN     = 1
+LWL     = 0              CAPMOD  = 2             XPART   = 0.5
+CGDO    = 2.56E-10       CGSO    = 2.56E-10      CGBO    = 1E-9
+CJ      = 2.823234E-4    PB      = 0.99          MJ      = 0.547221
+CJSW    = 1.292241E-10   PBSW    = 0.99          MJSW    = 0.1
+CJSWG   = 6.4E-11        PBSWG   = 0.99          MJSWG   = 0.1
+CF      = 0               )
*
.MODEL CMOSP PMOS (                               LEVEL   = 49
+VERSION = 3.1            TNOM    = 27            TOX     = 3.16E-8
+XJ      = 3E-7           NCH     = 2.4E16        VTH0    = -0.8476404
+K1      = 0.4513608      K2      = 2.379699E-5   K3      = 13.3278347
+K3B     = -2.2238332     W0      = 9.577236E-7   NLX     = 1E-6
+DVT0W   = 0              DVT1W   = 0             DVT2W   = 0
+DVT0    = 2.8137786      DVT1    = 0.7604621     DVT2    = -0.052421
+U0      = 236.8923827    UA      = 3.833306E-9   UB      = 1.487688E-21
+UC      = -1.08562E-10   VSAT    = 1.159861E5    A0      = 0.9443065
+AGS     = 0.259481       B0      = 3.299132E-6   B1      = 5E-6
+KETA    = 9.832612E-4    A1      = 0             A2      = 0.364
+RDSW    = 3E3            PRWG    = 0.0729646     PRWB    = -0.1851255
+WR      = 1              WINT    = 7.565065E-7   LINT    = 1.415433E-7
+XL      = 0              XW      = 0             DWG     = -2.13917E-8
+DWB     = 3.857544E-8    VOFF    = -0.0877184    NFACTOR = 0.2508342
+CIT     = 0              CDSC    = 2.924806E-5   CDSCD   = 1.497572E-4
+CDSCB   = 1.091488E-4    ETA0    = 0.25103       ETAB    = 4.268713E-3
+DSUB    = 0.2873         PCLM    = 6.403032E-10  PDIBLC1 = 7.477411E-4
+PDIBLC2 = 3.271335E-3    PDIBLCB = -1E-3         DROUT   = 1E-3
+PSCBE1  = 3.515038E9     PSCBE2  = 5.273648E-10  PVAG    = 14.985
+DELTA   = 0.01           RSH     = 77.3          MOBMOD  = 1
+PRT     = 0              UTE     = -1.5          KT1     = -0.11
+KT1L    = 0              KT2     = 0.022         UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11      AT      = 3.3E4
+WL      = 0              WLN     = 1             WW      = 0
+WWN     = 1              WWL     = 0             LL      = 0
+LLN     = 1              LW      = 0             LWN     = 1
+LWL     = 0              CAPMOD  = 2             XPART   = 0.5
+CGDO    = 2.65E-10       CGSO    = 2.65E-10      CGBO    = 1E-9
+CJ      = 3.017493E-4    PB      = 0.8           MJ      = 0.4487672
+CJSW    = 1.634365E-10   PBSW    = 0.99          MJSW    = 0.1219817
+CJSWG   = 3.9E-11        PBSWG   = 0.99          MJSWG   = 0.1219817
+CF      = 0               )
```

BIOGRAPHY

Date of Birth	04.08.1980	
Place of Birth	Kocaeli	
High School	1991-1998	Kocaeli Anatolian High School
B.Sc	1998-2002	Yildiz Technical University Faculty of Electric and Electronics Department of Electronics and Comm. Enginnering
M.Sc	2002-2004	Yildiz Technical University Institute of Sciences Division of Electronics and Comm. Engineering Electronics Program
Ph.D	2004-2011	Yildiz Technical University Institute of Sciences Division of Electronics and Comm. Engineering Electronics Program

Work Experience

2002-Present	Research Assistant in Department of Electronics and Comm. Engineering, Yildiz Technical University

Research Projects

1) **"A Mixed Mode Chip Implementation of General Purposed Neural Network",** supported by TUBITAK (The Scientific and Technological Research Council of Turkey), Project Number: 104E133, Start Date: August 2005, End Date: August 2008.
2) "**Electronic Circuit Design Automation with Particle Swarm Optimization and Artificial Neural Networks**", supported by YTU-BAPK (Yildiz Technical University, Organization of Scientific Research Project) Project Number: 29-04-03-KAP01, Start Date: May 2010, End Date: May 2012.
3) "**Technology Independent Neural Network Modeling for VLSI Design Automation**", supported by YTU-BAPK (Yildiz Technical University, Organization of Scientific Research Project), Project Number: 26-04-03-01, Start Date: July 2006, End Date: April 2009.
4) "**Investigation of Usage of Artificial Neural Networks in Medical Area**", supported by YTU-BAPK (Yildiz Technical University, Organization of Scientific Research Project), Project Number: 24-04-03-01, Start Date: April 2004, End Date: April 2005.

Contests

- **Technology Independent Neural Network Modeling for VLSI Design Automation (**by Nihan Kahraman, Burcu Erkmen and Revna Acar Vural) project was elected as **4th** best project in CADENCE DESIGN CONTEST 2008.
- **A Mixed Mode Chip Implementation of General Purposed Neural Network** (by Burcu Erkmen, Nihan Kahraman and Revna Acar Vural) project was elected as **7th** best project in CADENCE DESIGN CONTEST 2008.