AN ABSTRACT OF THE THESIS OF

Volodymyr Kratyuk for the degree of Master of Science in Electrical and Computer Engineering presented on June 6, 2003.

Title: Algorithms and Tools for Optimization of Integrated RF VCOs.

Abstract approved: Kartikeya Mayaram

This thesis presents algorithms and tools for the automated design of RF LC CMOS voltage controlled oscillators (VCOs) with low phase noise given a set of specifications. The electromagnetic solver, ASITIC, combined with the circuit simulator, SpectreRF, allows optimization of the VCO circuit parameters and inductor layout. This approach gives a phase noise improvement of up to 20 dBC/Hz in the flicker noise region and up to 5 dBC/Hz in the thermal noise region.

An optimization program for the computer-aided design of on-chip spiral inductors has also been developed. This program allows the designer to obtain the layout of an inductor with a required inductance value and maximal quality factor, thus enabling a reduction in the phase noise of the VCO being designed.

The circuit simulator SPICE3 has been extended to handle phase noise analysis based on a non-linear perturbation analysis for oscillators. The implemented technique allows for an accurate simulation of phase noise due to devices described either by analytical or numerical models. With this extension, the automated design of RF LC oscillators can be performed within the SPICE3 framework. Furthermore, the technique is available in a public domain software and can be extended to other application domains.
© Copyright by Volodymyr Kratyuk

June 6, 2003

All Rights Reserved
Algorithms and Tools for Optimization of Integrated RF VCOs

by
Volodymyr Kratyuk

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 6, 2003
Commencement June 2004
Master of Science thesis of Volodymyr Kratyuk presented on June 6, 2003

APPROVED:

[Redacted for Privacy]

Major Professor, representing Electrical and Computer Engineering

[Redacted for Privacy]

Director of the School of Electrical Engineering and Computer Science

[Redacted for Privacy]

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

[Redacted for Privacy]

Volodymyr Kratyuk, Author
ACKNOWLEDGEMENT

I would like to take this opportunity to express my gratitude towards all the people who assisted me during my schooling here at Oregon State University (OSU) in general and particularly while working on my research.

Dr. Oleg Mikulchenko, my advisor at National Technical University of Ukraine "Kyiv Polytechnic Institute", initiated my engineering career in computer-aided design by introducing me to the subject eight years ago. He has always inspired me to work hard thus stimulating my academic growth. It is with his help that I came to OSU and was able to complete my Master's program. It is but natural that I shall be forever indebted to him.

My sincere thanks go to Dr. Kartikeya Mayaram, my research advisor at OSU for having given me an opportunity to come to OSU and giving tremendous support to accomplish my goal. I took several of his classes, which supplied me with crucial theoretical knowledge, thus equipping me for my research. I also would like to thank him for his faith in my abilities and for having supported me throughout my work.

Speaking about research, Yutao Hu greatly helped me with theoretical questions and Nathen Barton with SpectreRF data post processing scripts. Taras Dudar and Yutao Hu graciously helped me out of many a tight spot when it came to software-related issues, for which I am very grateful to them.

I would like to express my deep appreciation for Madhu Chennam, Nathen Barton, Nilakantan Seshan, and Taras Dudar who patiently answered every question of mine and for the numerous enlightening discussions we had in the lab.

I am particularly grateful to Partick Birrer and Dr. Jean-Pierre Steger for helping me make it to the internship at HTA Burgdorf in Switzerland. A special thanks goes to Ivo Osh and Prof. Daniel von Grunigen, my mentors during the
internship and to the whole team of HTA Burgdorf co-workers for their time and
desire to teach me nuances of engineering and for a great international experience.

I am indebted to my friend, Ajit Sharma for his time and effort spent on
reviewing and making corrections to this thesis.

This research has been supported and funded by CDADIC, NSF and SRC. I
thank them all for the opportunity they have extended to me. I am also grateful to
the school of Electrical Engineering and Computer Science at Oregon State
University for providing me an excellent working environment.

The profound sense of gratitude I feel for my parents who showed me the
proper way to lead life cannot be expressed in words. I am also very grateful for my
friends from the KPI alpine team for emotional support during my years of study.

Finally, I would like to thank my friends and colleagues in the AMS group,
here at OSU, for the camaraderie and the enriching experiences over last couple of
years.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
<th>1</th>
<th>INTRODUCTION</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1.1 Motivation</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2 Summary of Previous Work</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3 Contributions of This Work</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4 Thesis Organization</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>OPTIMIZATION TECHNIQUES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.1 Optimization Basics</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.2 VCO Optimization (VCOOPT)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.3 Automated Inductor Design</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4 VCO Design Flow Using Inductor Optimization</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>OPTIMIZATION RESULTS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.1 Inductor optimization results</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.2 NMOS VCO</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.3 Complementary VCO</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.4 Summary</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>PHASE NOISE CALCULATION TECHNIQUES</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Implementation Aspects of Phase Noise Calculation</td>
<td>27</td>
</tr>
<tr>
<td>4.2 Mapping Matrix for White Noise Sources</td>
<td>30</td>
</tr>
<tr>
<td>4.3 Mapping Vector for Colored Noise Sources</td>
<td>32</td>
</tr>
<tr>
<td>4.4 PPV Calculation</td>
<td>33</td>
</tr>
<tr>
<td>4.5 Integration Method for PPV Calculation</td>
<td>35</td>
</tr>
<tr>
<td>4.6 Overall Implementation</td>
<td>36</td>
</tr>
<tr>
<td>5 PHASE NOISE SIMULATOR VALIDATION</td>
<td>39</td>
</tr>
<tr>
<td>5.1 Testing of Different Noise Models</td>
<td>39</td>
</tr>
<tr>
<td>5.2 Thirteen-stage Ring Oscillator</td>
<td>41</td>
</tr>
<tr>
<td>5.3 NMOS Cross-coupled Oscillator</td>
<td>43</td>
</tr>
<tr>
<td>5.4 Three-stage Differential Ring Oscillator with Buffer Circuit</td>
<td>45</td>
</tr>
<tr>
<td>6 CONCLUSIONS</td>
<td>48</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>50</td>
</tr>
<tr>
<td>APPENDICES</td>
<td>52</td>
</tr>
<tr>
<td>APPENDIX A SPICE3 bugs report</td>
<td>53</td>
</tr>
<tr>
<td>A.1 Objects Cleaning Routine Bug</td>
<td>53</td>
</tr>
<tr>
<td>A.2 BSIM3 Flicker Noise Model Bug</td>
<td>54</td>
</tr>
<tr>
<td>A.3 MOS1 Flicker Noise Model Bug</td>
<td>56</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Block diagram of a receiver's front end</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>The effect of the local oscillator phase noise in presence of noise and an adjacent channel interferer</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>VCO optimization flowchart</td>
<td>14</td>
</tr>
<tr>
<td>2.2</td>
<td>Inductor optimization methodology</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>An on-chip spiral inductor layout</td>
<td>16</td>
</tr>
<tr>
<td>2.4</td>
<td>An on-chip spiral inductor</td>
<td>17</td>
</tr>
<tr>
<td>2.5</td>
<td>VCO design flow using inductor optimization</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>NMOS cross-coupled pair</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td>NMOS VCO optimization results</td>
<td>22</td>
</tr>
<tr>
<td>3.3</td>
<td>NMOS-PMOS cross-coupled pair</td>
<td>23</td>
</tr>
<tr>
<td>3.4</td>
<td>Complementary VCO optimization results</td>
<td>24</td>
</tr>
<tr>
<td>4.1</td>
<td>Periodic noise and PPV for a Colpitts oscillator</td>
<td>29</td>
</tr>
<tr>
<td>4.2</td>
<td>Stamps for the $B_w$ matrix and $b_w$ vector</td>
<td>31</td>
</tr>
<tr>
<td>4.3</td>
<td>Stamp for the $B_{em}$ vector</td>
<td>32</td>
</tr>
<tr>
<td>4.4</td>
<td>A flowchart of the phase noise analysis implementation in SPICE3</td>
<td>37</td>
</tr>
<tr>
<td>5.1</td>
<td>Three-stage ring oscillator</td>
<td>40</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5.2</td>
<td>Thirteen-stage ring oscillator.</td>
<td>41</td>
</tr>
<tr>
<td>5.3</td>
<td>Inverter based delay cell.</td>
<td>41</td>
</tr>
<tr>
<td>5.4</td>
<td>Phase noise of the thirteen-stage ring oscillator with MOS level-one</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>transistor models.</td>
<td></td>
</tr>
<tr>
<td>5.5</td>
<td>Phase noise of the thirteen-stage ring oscillator with BSIM3 transistor</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>models.</td>
<td></td>
</tr>
<tr>
<td>5.6</td>
<td>NMOS cross-coupled oscillator.</td>
<td>43</td>
</tr>
<tr>
<td>5.7</td>
<td>Phase noise of the NMOS cross-coupled oscillator with MOS level-one</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>transistor model.</td>
<td></td>
</tr>
<tr>
<td>5.8</td>
<td>Phase noise of the NMOS cross-coupled oscillator with the BSIM3</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>transistor model.</td>
<td></td>
</tr>
<tr>
<td>5.9</td>
<td>Three-stage differential ring oscillator.</td>
<td>45</td>
</tr>
<tr>
<td>5.10</td>
<td>Maneatis load delay cell.</td>
<td>46</td>
</tr>
<tr>
<td>5.11</td>
<td>Buffer for differential circuits.</td>
<td>46</td>
</tr>
<tr>
<td>5.12</td>
<td>Phase noise of three-stage differential ring oscillator with and without</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>buffer.</td>
<td></td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Coefficients for the data-fitted monomial expression</td>
<td>18</td>
</tr>
<tr>
<td>3.1 Optimal inductor layout for an inductance value of 2nH, inductance tolerance of 2%, and an operating frequency of 2.4GHz</td>
<td>20</td>
</tr>
<tr>
<td>3.2 Optimal inductor layout for an inductance value of 4nH, inductance tolerance of 2%, and an operating frequency of 4.2GHz</td>
<td>20</td>
</tr>
<tr>
<td>3.3 Summary of NMOS VCO optimization results</td>
<td>22</td>
</tr>
<tr>
<td>3.4 Summary of complementary VCO optimization results</td>
<td>25</td>
</tr>
<tr>
<td>5.1 “Noimod” flag for different noise models</td>
<td>40</td>
</tr>
<tr>
<td>5.2 Phase noise simulation results for the three-stage ring oscillator</td>
<td>40</td>
</tr>
</tbody>
</table>
To my family
ALGORITHMS AND TOOLS FOR OPTIMIZATION OF INTEGRATED RF VCOS

1  INTRODUCTION

Wireless products, such as cellular phones, GPS receivers, wireless LAN and computer peripherals (wireless mice and keyboards) have become an integral part of our day-to-day life. A customer wants to see low cost, high performance, compact, and up-to-date solutions on the market. The advancements in integrated circuit technology have been possible because of the gigantic leaps made in the fields of computer hardware and electronic design automation (EDA). The increased computing power, coupled with sophisticated design automation tools have fuelled the semiconductor revolution, and have helped make Moores’ law a reality.

The requirement that modern-day wireless devices have to meet, translate into demands for very accurate and fast systems for computer-aided design (CAD) of analog and radio frequency (RF) integrated circuits (IC’s). These CAD tools must be able to assist a designer during each step of the development of a new product: conceptualization, circuit design, circuit simulation, layout optimization, physical verification, parasitic extraction and verification including the extracted parasitic elements.

The modern EDA industry can be said to have had its roots in SPICE. SPICE revolutionized circuit design, by giving designers a whole new way to analyze circuits and unprecedented enhancement in the speed of analysis. Now circuits can be analyzed much faster and more accurately thus shortening design cycles and enabling designers to concentrate more on the architecture and design of circuits themselves. SPICE, in simple terms, was the first “CAD tool.” As mentioned earlier, over the last couple of decades, EDA tools have evolved
significantly since the early days of SPICE. Today CAD tools must not only have simulation capabilities for a specified task, but must also be able to help the designer choose an optimal circuit architecture, help with layout placement and routing to save chip area and reduce effects of substrate and supply noise coupling, and also help optimize circuit as well as layout parameters. In this thesis the focus is on the automated design of integrated RF voltage controlled oscillators.

1.1 Motivation

Voltage controlled oscillators (VCOs) are important building blocks in RF ICs. Usually VCOs are used as an integral part of the local oscillator that upconverts or downconverts signals. For these applications, VCOs have very stringent phase noise specifications. Spectral impurity of local oscillators can lead to channel interference and degradation of signal-to-noise ratio (SNR). The destructive effect of phase noise can be explained with the example of a simple radio receiver [1]. Figure 1.1 shows the block-diagram of the front end of a radio receiver, which includes an antenna (ANT), a low noise amplifier (LNA), a mixer (MIX), and a local oscillator (LO).

![Block diagram of a receiver's front end.](image)

Figure 1.1 Block diagram of a receiver's front end.
Figure 1.2 The effect of the local oscillator phase noise in presence of noise and an adjacent channel interferer.

An RF signal is received by the antenna and amplified by the LNA. The RF signal contains a desired signal and might contain a signal from an adjacent radio channel which lies in the LNA pass band. If the local oscillator has a large phase noise, the adjacent radio channel will be downconverted to the intermediate frequency (IF) together with the desired signal as shown in Figure 1.2. Also, additional noise, in the vicinity of the desired signal, will get downconverted to the IF, because the LO power spectral density (PSD) has a skirt. This will have a strong impact on the performance of the receiver. Improving the phase noise of the LO will improve the signal-to-noise ratio of the desired signal and thereby reduce interference from nearby channels.
The VCO phase noise depends on a number of circuit and layout parameters such as supply current, transistor sizing, inductor quality factor, etc. Often it is not clear how these parameters should be chosen in order to reduce phase noise.

One of the major issues during the design of integrated RF VCOs is the design of on-chip spiral inductors. Inductors are present in a number of circuits such as LNAs, VCOs, phase locked loops (PLLs). Due to parasitics, substrate losses and proximity effects the quality factor, $Q$, of inductors is small. This low quality factor obtained with on-chip spirals results in poor phase noise performance of VCOs. Since present day wireless applications have stringent phase noise demands, ideally, inductors with a large $Q$ are desirable. To be able to layout a suitable inductor that meets all requirements, the designer needs to spend a significant amount of time simulating alternate configurations. For such tasks computer-aided optimization can simplify the design process.

1.2 Summary of Previous Work

Several recent publications present different kinds of optimizers for the phase noise improvement of VCOs. A simulator-optimizer program for spiral inductors on silicon substrates is presented in [2]. The program performs an optimization loop around the electromagnetic solver FastHenry [3] to find an inductor with a high quality factor. The designed inductor is then used for VCO design. In this approach, phase noise improvement can be achieved only due to an increase in the quality factor of the on-chip spiral inductor. Other circuit parameters such as the width and length of transistors have to be chosen by the designer without any computer aids.

A more sophisticated tool, CYCLONE, is presented in [4]. CYCLONE is an automated, layout aware RF LC-oscillator design tool. This tool optimizes the layout of a VCO, including that of the inductor to obtain as low a cost function as
possible. The cost function includes phase noise, consumed power and the tuning range. The main disadvantage of CYCLONE is that it uses approximate models for the cost function calculation in general and for phase noise in particular. Phase noise is calculated using a linear time invariant approach with the assumption of no amplitude limiting [5]. This approach is not very accurate as explained in [6] and may lead to sub-optimal results.

1.3 Contributions of This Work

In this work several different approaches have been combined to develop an optimization tool that is accurate and efficient. Analytical expressions combined with simulations provide the foundation for this work.

Analytical expressions and electromagnetic solvers provide the inductance value and parasitics of a spiral inductor only after the layout has been created. While analytical expressions are easy to use, they yield only approximate results. On the other hand, electromagnetic solvers give accurate results but at a significant computation cost. The optimization approach presented in this thesis is based on both analytical expressions and electromagnetic solvers for the efficient computer-aided design of on-chip spiral inductors. The user can obtain the layout of an inductor that meets the specifications and that has the best possible quality factor.

The phase noise performance of an integrated RF VCO depends not only on the quality factor of the spiral inductor but also on transistor sizing. To optimize all of these parameters in terms of phase noise performance, VCOOPT - an optimization program for VCO design has been developed. Some circuit simulators such as SpectreRF and HPADS have a built-in circuit optimization program. However, these cannot optimize inductor layout parameters together with the circuit parameters, which is a unique feature of VCOOPT.
The principal difference between our optimization technique and the approach in [2] is that we optimize not only the inductor layout parameters but also the VCO circuit parameters. In contrast to [4] we use a circuit simulator (SpectreRF, SPICE3) for accurate phase noise analysis. Our approach results in improved VCO performance.

The key to this optimization is an accurate simulation of phase noise. To facilitate this, the circuit simulator SPICE3 has been extended to handle phase noise analysis based on a non-linear perturbation analysis for oscillators. The implemented technique allows for an accurate simulation of phase noise due to devices described either by analytical or numerical models. Furthermore, the technique is available in a public domain software and can be extended to other application domains.

1.4 Thesis Organization

The thesis is organized as follows. Chapter 1 gives an overview of the phase noise problem and the motivation for the development of an automated tool for VCO design, describes related recent publications and identifies the contribution of this work.

Chapter 2 explains different approaches to optimization of the VCOs and on-chip spiral inductors. The theoretical material related to the optimization techniques used is presented followed by the VCO optimization program, VCOOPT, and the inductor optimization tool.

Chapter 3 presents the optimization results from techniques presented in Chapter 2. The optimal inductor design for different specifications is demonstrated and NMOS and complementary VCO optimization results are presented.

Implementation aspects of the phase noise calculation technique, which has been incorporated in SPICE3, are explained in Chapter 4.
Simulation results from several different oscillator structures, which have been simulated with different sets of device noise models are presented in Chapter 5. These help to validate the implemented phase noise technique. Accuracy issues are also addressed.

Conclusions and suggestions for future work are given in Chapter 6.
2 OPTIMIZATION TECHNIQUES

2.1 Optimization Basics

The ultimate goal of optimization is to find the best solution to a specified problem. Mathematically, this means that the maximum or minimum of a function of \( n \) variables, \( f(x_1, x_2, \ldots, x_n) \) has to be found [7]. Consider a minimization problem.

Define \( f(X) = f(x_1, x_2, \ldots, x_n) \) to be the function which needs to be minimized (this may be referred to as a cost or objective function), where \( X = \{x_1, x_2, \ldots, x_n\} \) is a vector of variables. If \( X_i \) is an \( i \)-th approximation to the minimum point of a cost function, \( f(X) \) can be represented by a Taylor series expansion around \( X_i \):

\[
f(X) = f(X_i) + \nabla f(X_i)^T (X - X_i) + \frac{1}{2} (X - X_i)^T H(X_i) (X - X_i)
\]  

(2.1)

where \( H(X_i) \) is the Hessian matrix evaluated at the point \( X = X_i \), and:

\[
H(X) = 
\begin{bmatrix}
\frac{\partial^2 f(X)}{\partial x_1^2} & \frac{\partial^2 f(X)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(X)}{\partial x_1 \partial x_n} \\
\frac{\partial^2 f(X)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(X)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(X)}{\partial x_2 \partial x_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 f(X)}{\partial x_n \partial x_1} & \frac{\partial^2 f(X)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(X)}{\partial x_n^2}
\end{bmatrix}
\]  

(2.2)

At the minimum:

\[
\nabla f(X) = 0
\]  

(2.3)
A new approximation for the minimum point \(X_{i+1}\) can be found by applying (2.3) to (2.1).

\[
\nabla f(X_{i+1}) = \nabla f(X_i) + H(X_i)(X_{i+1} - X_i)
\]

(2.4)

Then,

\[
X_{i+1} = X_i - H^{-1}(X_i)\nabla f(X_i)
\]

(2.5)

In practice, it is not efficient to calculate the inverse of the Hessian matrix because the matrix inversion is computationally expensive. Instead a system of linear equations (2.6) can be solved for \(\Delta X_i\) (which is called the direction of descent), and \(X_{i+1}\) can then be found from (2.7).

\[
H(X_i)\Delta X_i = -\nabla f(X_i)
\]

(2.6)

\[
X_{i+1} = X_i + \Delta X_i
\]

(2.7)

The method shown in (2.6) is called the Newton method. If the initial guess \(X_0\) is given close enough to the solution \(X^*\), then this method has quadratic convergence (2.8).

\[
\|X_{i+1} - X^*\| \leq \|X_i - X^*\|^2
\]

(2.8)

The point \(X_{i+1}\) is not necessarily a minimum in the direction of descent \(\Delta X_i\), since a cost function is in general not quadratic. Equation (2.7) can be modified by introducing a multiplication coefficient \(\alpha\).
The coefficient $\alpha$ has to be chosen in such a way that $X_{i+1}$ is the point of the minimum in the direction of descent, $\Delta X_i$. $\alpha$ can be found by minimizing an objective function in the direction of descent. This means that the problem as shown in (2.10) has to be solved where $\alpha$ is an unknown.

$$\min_{\alpha} f(X_i + \alpha \Delta X_i)$$

(2.10)

Newton-type methods calculate the Hessian directly, solve the system of linear algebraic equations (2.6), and then perform a line search in the direction of descent, thereby solving (2.10), to find $X_{i+1}$.

One of the major disadvantages of using the Newton method is that a large number of objective function evaluations are necessary in order to numerically compute the Hessian matrix. Quasi-Newton methods avoid this problem by using an approximate Hessian. An approximation to the Hessian matrix can be obtained by using updating techniques. A large number of updating techniques have been developed. The most effective updating technique for use is the general propose method by Broyden, Fletcher, Goldfarb, and Shanoo (BFGS) [7]. This updating method is given by the Eq. (2.11).

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k s_k^T s_k H_k}{s_k^T H_k s_k}$$

(2.11)

where $s_k = X_{k+1} - X_k$ and $q_k = \nabla f(X_{k+1}) - \nabla f(X_k)$. 

$$X_{i+1} = X_i + \alpha \Delta X_i$$

(2.9)
At a starting point for the Hessian calculation the identity matrix can be used. To avoid solving a system of algebraic equations or calculating the inversion of the Hessian matrix, an update method for the inverse of the Hessian can be derived.

\[ H_{k+1}^{-1} = H_k^{-1} + \frac{q_k q_k^T}{q_k^T s_k} - \frac{1}{s_k^T H_k^{-1} s_k} \left( \frac{s_k^T (H_k^{-1})^T s_k}{q_k^T s_k} \right) \]  \hspace{1cm} (2.12)

The Quasi-Newton approach with the Hessian updating technique can use Eq. (2.13) for the direct calculation of a direction, then Eq. (2.10) can be solved to find a step in the direction of descent.

\[ \Delta X_i = -H^{-1}(X_i) \nabla f(X_i) \]  \hspace{1cm} (2.13)

This technique has a performance similar to a second-order Newton method but needs fewer objective function evaluations per step [7].

2.2 VCO Optimization (VCOOPT)

To achieve a desirable phase noise performance for RF VCOs, several different parameters such as inductor design, transistor sizing, and power need to be optimized simultaneously. In our approach, a Quasi-Newton method with BFGS (Broyden-Fletcher-Goldfarb-Shanno) formula for Hessian update [8] has been used. This technique has a performance similar to a second-order Newton method but needs fewer objective function evaluations. The circuit parameters which are optimized include the layout parameters for on-chip spiral inductors. The objective function is chosen to minimize the phase noise at specific frequency offset points. Each objective function evaluation is performed with the help of the
electromagnetic solver ASITIC [9] and the circuit simulator SpectreRF. ASITIC is a tool for computer aided-design of passive metal structures residing on a lossy substrate. SpectreRF is a circuit simulator with additional set of analyses for RF circuit simulation.

The overall flowchart of the VCO optimization program is shown in Figure 2.1. First, the inductor layout parameters are passed to ASITIC, which returns a π-model of the inductor. This π-model is included in the circuit netlist, which is then simulated by SpectreRF. A transient analysis is performed to check if the circuit oscillates and how long it takes to reach a stable amplitude. From the obtained transient response, an approximate frequency of oscillation can be found and used as an initial guess for the periodic steady state (PSS) analysis. Note, that at this step the PSS analysis cannot be used, since the frequency of oscillation and stabilization time are still unknown. An analysis is also performed to check if the circuit has a startup loop gain greater than two. Then a PSS analysis is run to calculate the frequency of oscillation. Due to the change in circuit parameters, the frequency of oscillation $f_{osc}$ also changes. To restore the frequency of oscillation back to the specified value $f_{specs}$, a fixed capacitor value $C_{fix}$ is changed with the help of a secant method, which is a method of solving nonlinear equations. Therefore, the next problem that needs to be solved is:

$$\left| f_{osc}(C_{fix}) - f_{specs} \right| = 0 \quad (2.14)$$

If the circuit oscillates, has a sufficient startup loop gain, and the output signal magnitude is not less than the specified value, the program performs periodic steady state (PSS) and phase noise (PNOISE) analyses in SpectreRF.

The results of the phase noise analysis are used to calculate the value of the objective function. The objective function (2.15) may be a simple sum of the phase noise at particular frequency offset points or a sum with weighted coefficients. This objective function evaluation technique is included in the optimization loop.
\[ F = \sum_{i=1}^{N} w_i \times PN_i \]  (2.15)

where \( N \) = the number of specified points, \( PN_i \) = phase noise in dBc/Hz at a specified frequency offset \( f_i \), and \( w_i \) = weighting coefficient for the phase noise contribution at an offset frequency \( f_i \).
Figure 2.1 VCO optimization flowchart.
2.3 Automated Inductor Design

In several design flows the designer is allowed to use only a set of pre-specified inductance values. These inductors are separately designed and characterized for manufacturability. The designer uses an inductor design from a library and does not have to spend time generating inductor layouts using electromagnetic solvers. However, if one has the flexibility of designing the inductors, then a separate optimizer can be used to generate an inductor topology with the highest possible quality factor $Q$. An optimization technique for the design of high $Q$ inductors has been developed and is presented in this section.

![Inductor optimization methodology](image.png)

Figure 2.2 Inductor optimization methodology.

The overall flowchart of the inductor optimization program is shown in Figure 2.2. Specifications for the inductor design include the frequency of operation, the inductance value and tolerance, and the process parameters. The program has two optimization loops. In the first optimization loop, a random search
is used given the maximum area for the inductor layout and the process corners as shown by Eqs. (2.16) – (2.19).

\[ R_{\text{min}} \leq R \leq R_{\text{max}} \]  
(2.16)

\[ w_{\text{min}} \leq w \leq w_{\text{max}} \]  
(2.17)

\[ s_{\text{min}} \leq s \leq s_{\text{max}} \]  
(2.18)

\[ N_{\text{min}} \leq N \leq N_{\text{max}} \]  
(2.19)

where \( R \) = radius of the inductor,
\( w \) = metal trace width,
\( s \) = spacing between metal traces, and
\( N \) = the number of turns for the spiral inductor.

These parameters are shown in Figure 2.3 for a square spiral inductor.

Figure 2.3 An on-chip spiral inductor layout.
Analytical expressions are optimized to find the initial guess for the second optimization loop. The data fitted monomial expressions (2.20) from [10] are used for inductance calculations.

\[ L = \beta d_{out}^{\alpha_1} w^{\alpha_2} d_{avg}^{\alpha_3} N^{\alpha_4} s^{\alpha_5} \]  

(2.20)

where \( d_{out} \) = outer diameter,

\( d_{in} \) = inner diameter,

\( d_{avg} = 0.5 \cdot (d_{out} - d_{in}) \) = average diameter,

\( w \) = metal trace width,

\( s \) = spacing between metal traces, and

\( N \) = the number of turns for the spiral inductor.

These parameters are shown in Figure 2.4 for a square spiral inductor.

Figure 2.4 An on-chip spiral inductor.

Coefficients \( \beta \) and \( \alpha_i \) are obtained by a data fitting technique [10] and are listed in Table 2.1.
Table 2.1 Coefficients for the data-fitted monomial expression.

<table>
<thead>
<tr>
<th>Layout</th>
<th>$\beta$</th>
<th>$\alpha_1$</th>
<th>$\alpha_2$</th>
<th>$\alpha_3$</th>
<th>$\alpha_4$</th>
<th>$\alpha_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Square</td>
<td>1.62e-3</td>
<td>-1.21</td>
<td>-0.147</td>
<td>2.40</td>
<td>1.78</td>
<td>-0.030</td>
</tr>
<tr>
<td>Hexagonal</td>
<td>1.28e-3</td>
<td>-1.23</td>
<td>-0.174</td>
<td>2.47</td>
<td>1.77</td>
<td>-0.049</td>
</tr>
<tr>
<td>Octagonal</td>
<td>1.33e-3</td>
<td>-1.21</td>
<td>-0.163</td>
<td>2.43</td>
<td>1.75</td>
<td>-0.049</td>
</tr>
</tbody>
</table>

The quality factor $Q$ of the inductor is calculated from the expression $Q=\omega_0 L / r$, where $\omega_0$ is the frequency of operation, $L$ is the inductance value calculated by Eq. (2.20), and $r$ is the series resistance which can be calculated from process parameters. All these expressions are functions of the inductor layout parameters: radius ($R$), metal width ($w$), spacing ($s$), the number of turns ($N$) and the number of sides (4 for square, 6 for hexagonal, and 8 octagonal).

A disadvantage of the analytical expressions, is that they are not very accurate. However, analytical expressions provide a quick initial guess for the second optimization loop. In the second optimization loop, ASITIC is used to obtain accurate results. ASITIC can calculate the equivalent $\pi$-model, the quality factor and self-resonance frequency of an on-chip spiral inductor. This solver is incorporated in an optimization loop and the objective function (2.21) is chosen to get as high a quality factor as possible. This combination of the two optimization loops results in the desired inductor in a small amount of computational time. A Quasi-Newton method with the BFGS technique for Hessian update is used for optimization. The objective function is given by:

$$F = e^{|L_{\text{req}} - L|} + e^{|Q_g - Q|}$$  \hspace{1cm} (2.21)

where $L_{\text{req}}$ = the required inductance value, and $Q_g$ = the desired inductor quality factor.
2.4 VCO Design Flow Using Inductor Optimization

One difficulty with using VCOOPT is that a designer has to specify an initial guess for the inductor layout parameters, when only the required value of inductance is known. To avoid this problem we propose a VCO design flow (Figure 2.5) that incorporates the inductor optimization program described in the previous section. When an initial design is done and a required value of inductance is known, the inductor optimization program can be used to obtain an inductor with the best possible quality factor. After this, the inductor model is used by the VCO optimizer and only the circuit parameters are optimized. Optimization results from this approach are provided in the next chapter.

Figure 2.5 VCO design flow using inductor optimization.
3 OPTIMIZATION RESULTS

3.1 Inductor optimization results

The results for the inductor topology optimization that were described in the previous chapter are shown in Table 3.1 and Table 3.2. All inductors have twelve sides and were designed in the 0.35μm TSMC process. Without the use of an optimization technique the inductor quality factor is less than 6 for the given specifications.

Table 3.1 Optimal inductor layout for an inductance value of 2nH, inductance tolerance of 2%, and an operating frequency of 2.4GHz.

<table>
<thead>
<tr>
<th>Q</th>
<th>L(nH)</th>
<th>w(μm)</th>
<th>S(μm)</th>
<th>Turns</th>
<th>R(μm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.0</td>
<td>2.00</td>
<td>29.3</td>
<td>1.0</td>
<td>$3^{1/12}$</td>
<td>159</td>
</tr>
<tr>
<td>9.4</td>
<td>2.03</td>
<td>29.0</td>
<td>4.3</td>
<td>$2^{5/12}$</td>
<td>190</td>
</tr>
<tr>
<td>9.2</td>
<td>2.02</td>
<td>30.0</td>
<td>2.5</td>
<td>$3^{5/12}$</td>
<td>166</td>
</tr>
<tr>
<td>8.5</td>
<td>1.98</td>
<td>26.1</td>
<td>1.0</td>
<td>$3^{5/12}$</td>
<td>146</td>
</tr>
</tbody>
</table>

Table 3.2 Optimal inductor layout for an inductance value of 4nH, inductance tolerance of 2%, and an operating frequency of 4.2GHz.

<table>
<thead>
<tr>
<th>Q</th>
<th>L(nH)</th>
<th>w(μm)</th>
<th>S(μm)</th>
<th>Turns</th>
<th>R(μm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.7</td>
<td>4.01</td>
<td>14.0</td>
<td>2.7</td>
<td>$3^{10/12}$</td>
<td>141</td>
</tr>
<tr>
<td>7.2</td>
<td>4.01</td>
<td>16.7</td>
<td>1.7</td>
<td>3</td>
<td>173</td>
</tr>
<tr>
<td>7.4</td>
<td>4.01</td>
<td>13.4</td>
<td>4.3</td>
<td>$5^{10/12}$</td>
<td>132</td>
</tr>
<tr>
<td>7.3</td>
<td>3.99</td>
<td>12.6</td>
<td>1.6</td>
<td>$2^{7/12}$</td>
<td>163</td>
</tr>
</tbody>
</table>
3.2 NMOS VCO

The NMOS cross-coupled VCO (Figure 3.1) was optimized for improved phase noise performance by VCOOPT. The oscillator has a 3V power supply and consumes 10mA of current. The inductor layout parameters (radius (R), metal width (w), spacing (s) and the number of turns (N)) and transistor sizing (width and length) were chosen as the variables for optimization. The time required for simulation was approximately 10 hours on a Sun Ultra 10 workstation. The phase noise plots before and after optimization are shown in Figure 3.2, where the x-axis is the offset frequency in logarithmic scale and the y-axis is the phase noise in decibels below the carrier.

Figure 3.1 NMOS cross-coupled pair.
Figure 3.2 NMOS VCO optimization results.

As seen from these plots, the improvement in the flicker noise region is up to 20 dBc/Hz and in the thermal noise region is up to 5 dBc/Hz. Table 3.3 compares the circuit parameters before and after optimization.

Table 3.3 Summary of NMOS VCO optimization results.

<table>
<thead>
<tr>
<th>Circuit parameters</th>
<th>Initial</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>W / L of NMOS transistors</td>
<td>80µm/0.4µm</td>
<td>60µm/0.74µm</td>
</tr>
<tr>
<td>Capacitance</td>
<td>1.55pF</td>
<td>1.3pF</td>
</tr>
<tr>
<td>Inductance</td>
<td>1.1nH</td>
<td>1.26nH</td>
</tr>
<tr>
<td>Inductor series resistance</td>
<td>3Ω</td>
<td>2.66Ω</td>
</tr>
<tr>
<td>Inductor quality factor</td>
<td>5.3</td>
<td>6.4</td>
</tr>
</tbody>
</table>
3.3 Complementary VCO

The NMOS-PMOS cross-coupled VCO (Figure 3.3) was designed for an oscillation frequency of 2.4GHz using the approach described in Section 2.4. An initial design was done with an ideal inductor. Specifications for the required inductor were then given to the inductor optimization program and an inductor with a quality factor of $Q=6.7$ was generated. The inductor model and VCO circuit parameters were passed to VCOOPT where transistor sizes were optimized. The phase noise plots before and after optimization are shown in Figure 3.4, where the x-axis is the offset frequency in logarithmic scale and the y-axis is the phase noise in decibels below the carrier.

![NMOS-PMOS cross-coupled pair](image)

Figure 3.3 NMOS-PMOS cross-coupled pair.
As seen from these plots, the improvement in the flicker noise region is up to 25 dBc/Hz but in the thermal noise region there is a degradation of about 2 dBc/Hz. This could be changed by the use of different weights in the objective function. A comparison with the use of VCOOPT only is also provided in Figure 3.4. From this figure it is clear that this approach gives better performance compared to VCOOPT. Table 3.4 compares the circuit parameters before and after optimization.
Table 3.4 Summary of complementary VCO optimization results.

<table>
<thead>
<tr>
<th>Circuit parameters</th>
<th>Initial</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>W / L of NMOS transistors</td>
<td>120μm/0.4μm</td>
<td>42μm/1.87μm</td>
</tr>
<tr>
<td>W / L of PMOS transistors</td>
<td>120μm/0.4μm</td>
<td>130μm/0.42μm</td>
</tr>
<tr>
<td>Capacitance</td>
<td>1.18pF</td>
<td>1.17pF</td>
</tr>
<tr>
<td>Inductance</td>
<td>1.09nH</td>
<td></td>
</tr>
<tr>
<td>Inductor series resistance</td>
<td></td>
<td>2.38Ω</td>
</tr>
<tr>
<td>Inductor quality factor</td>
<td></td>
<td>6.7</td>
</tr>
</tbody>
</table>

3.4 Summary

Optimization programs for the computer-aided design of on-chip spiral inductors and integrated RF VCOs have been developed. Carefully chosen optimization techniques allow for efficient automated design. The inductor optimization program generates the layout of the inductor for a given set of specifications with a maximal quality factor. This program can be used in conjunction with any electromagnetic solver by adding proper software interface. The VCO optimization program optimizes the circuit parameters and the inductor layout for minimum phase noise. This program combines the circuit simulator SpectreRF with the electromagnetic solver ASITIC. Results from these programs demonstrate the benefits of using automated design tools for circuit design.
The two most widely used methods for analyzing oscillator phase noise are (i) Hajimiri and Lee theory [6] and [14], and (ii) the method described in [11], [12], and [13]. The phase noise solver based on Hajimiri and Lee theory was presented in [15]. The key aspect is the calculation of the impulse sensitivity function (ISF) [6] using transient analysis. The ISF is basically the transfer function from a current impulse to the phase deviation at the output. The advantage of that method is that only transient analysis is needed for the simulations. However, the ISF has to be calculated for each node perturbed by a noise source, so multiple transient analyses are needed and the ISF has to be extracted using post processing. This makes the Hajimiri and Lee method slow for large circuits.

The method based on nonlinear perturbation analysis by Demir, et. al., was chosen for the purpose of implementation into the circuit simulator SPICE3, in this work, because of following features [11]:

1) The method requires only the steady-state solution and the values of the noise sources. This information can be easily obtained from any circuit simulator which provides steady-state or transient analyses.

2) The computational time and required memory scale linearly with circuit size, hence allowing efficient simulation of large circuits.

3) The contribution of each noise source to the overall phase noise can be obtained at the expense of minimal increase in the computational time and required memory.
4.1 Implementation Aspects of Phase Noise Calculation

Almost all circuit simulators, and SPICE3 in particular, use the modified nodal analysis (MNA) formulation of circuit equations. A system of differential algebraic equations (DAEs) of the form shown in Eq. (4.1) are solved.

\[
\frac{d}{dt} q(x) + g(x) = 0
\]  

(4.1)

where \( q(x) \) is a vector that represents the sum of capacitor charges at each node, and \( g(x) \) is a vector that represents the sum of resistor currents and current sources at each node.

Consider a circuit which consists of \( p \) white noise sources and \( M \) colored noise sources and which can be described by a system of \( n \) DAEs of the form shown in (4.1). The system in (4.1) can be modified into the form shown in Eq. (4.2) to describe the impact of noise sources:

\[
\frac{d}{dt} q(x) + g(x) + B_w(x)b_w(t) + \sum_{m=1}^{M} B_{cm}(x)b_{cm}(t) = 0
\]  

(4.2)

where \( B_w \) is a state dependent \( n \times p \) matrix that maps a vector of white noise sources \( b_w(t) \) with unity PSD to the system of DAEs (4.1) and \( B_{cm} \) is a state dependent \( n \)-dimensional vector that maps the \( m \)-th colored noise source, \( b_{cm}(t) \), with the PSD \( S_{Nm}(f) \) to the system of DAEs (4.1).

Let us define \( x_0(t) \) as the steady-state solution of an unperturbed oscillator and \( x_0(t + \alpha(t)) \) as the steady-state solution of an oscillator with perturbed phase which satisfies (4.2). A single-sided spectral density, \( S_{ss}(f) \), of \( x_0(t + \alpha(t)) \) [11] is given by:
\[ S_{ss}(f) = 2 \sum_{i=-\infty}^{\infty} X_i X_i^* \frac{f_0^2 i^2 c(f)}{\pi^2 f_0^4 i^4 c^2(f) + (f + if_0)^2} \]  \hspace{1cm} (4.3)

where \( f \) = a frequency of interest and \( 0 \leq f < \infty \),
\( f_0 \) = frequency of oscillation,
\( i \) = the index of summation.
\( X_i \) = \( i \)-th Fourier series coefficient of \( x_i(t) \),
\( c \) = a scalar that is a function of frequency.

Therefore the single-sideband phase noise spectrum \( L(f_m) \) in dBc/Hz is given by:

\[ L(f_m) = 10 \log_{10} \left( \frac{S_{ss}(f_0 + f_m)}{2|X_1|^2} \right) \]  \hspace{1cm} (4.4)

where \( f_m \) is an offset frequency.

If the offset frequency is less than the frequency of oscillation, \( 0 \leq f_m \ll f_0 \), and \( c \) is small, then Eq. (4.4) can be approximated as:

\[ L(f_m) = 10 \log_{10} \left( \frac{f_0^2 c(f_m)}{\pi^2 f_0^4 c^2(f_m) + f_m^2} \right) \]  \hspace{1cm} (4.5)

The scalar constant \( c \) is frequency dependent in the general case and is given by:

\[ c(f) = c_w + \sum_{m=1}^{M} |c_{cm}(f)|^2 \]  \hspace{1cm} (4.6)
where \( c_w \) is a contribution to the scalar \( c \) from white noise sources [11], and is given by:

\[
c_w = \frac{1}{T} \int_{0}^{T} \nu^{T}_{\tau}(\tau)B_{w}(x_{\tau}(\tau))B_{w}^{T}(x_{\tau}(\tau))\nu_{\tau}(\tau) d\tau
\]  \hspace{1cm} (4.7)

where \( \nu_{\tau}(\tau) \) stands for the perturbation projection vector (PPV) [13]. The PPV is a periodic vector which serves as a transfer function from the noise sources to the scalar \( c \), and hence to the overall phase noise power spectral density. Figure 4.1 shows the time dependent PSD of the channel thermal noise of the MOS transistor and the PPV for that noise source for a Colpitts oscillator. As can be seen from Eq. (4.7), the PPV scales the amount of noise transferred to the scalar \( c \) at each point of time. Simply, the curves on Figure 4.1 are multiplied and then integrated for one period.

![Graph of Figure 4.1: Periodic noise and PPV for a Colpitts oscillator.](image-url)
\( c_{cm} \) in Eq. (4.6), is the contribution to the scalar \( c \) from the \( m \)-th colored noise source [12] and is given by:

\[
c_{cm} = \frac{1}{T} \int_0^T v_i^T(\tau) B_{cm} (x_i(\tau)) d\tau
\]  

(4.8)

Thus, in order to get the single-sideband phase noise spectrum \( L(f_m) \) in dBc/Hz, first a scalar \( c \) (4.6) needs to be calculated using (4.7) and (4.8), and then the phase noise spectrum is computed using (4.5). The method to obtain the PPV and matrices \( B_w \) and \( B_{cm} \) will be described in the next two sections.

### 4.2 Mapping Matrix for White Noise Sources

The process of creating the matrix \( B_w(x(t)) \) can be explained by using a noisy diode as an example as shown in Figure 4.2. The matrix \( B_w(x(t)) \) has to map stationary white noise sources (with unity PSD) to the circuit equations. Thus, it must take care of adding information about the PSD of the white noise sources in addition to the state dependent modulation terms. In most practical cases, white noise sources are either stationary and do not need modulation terms (like a resistor thermal noise source) or nonstationary, where the modulation terms naturally get included into the PSD formula (like a diode shot noise source).
Assume a diode is connected between nodes $N_+$ and $N_-$ and $I_{dd}(x(t))$ is the state-dependent current that flows through it. The diode is a source of white noise, shot noise, with double-sideband PSD given by $qI_{dc}(x(t))$. Therefore, the matrix $B_n(x(t))$ will get two entries in rows which correspond to $N_+$ and $N_-$ nodes as shown in Figure 4.2.

For implementation purposes, it is more efficient to form $B_n(x(t))B_n^T(x(t))$ (required in Eq.(4.7)), instead of forming $B_n(x(t))$ and then multiplying it by $B_n^T(x(t))$. It is easy to see that for the example in Figure 4.2 $B_n(x(t))B_n^T(x(t))$ is given by:
4.3 Mapping Vector for Colored Noise Sources

The method of stamping mapping vectors for colored noise sources is similar to that for stamping the columns of the mapping matrix for white noise sources, as can be seen in Figure 4.3. Eq. (4.8) requires such a representation of the mapping vector. Also, the mapping vector for a colored noise source has to be evaluated for each frequency of interest.

\[
B_n(x(t))B_n^T(x(t)) = \begin{bmatrix} 
... & ... & ... & ... & ... \\
... & +qI_{DC}(x(t)) & ... & -qI_{DC}(x(t)) & ... \\
... & ... & ... & ... & ... \\
... & -qI_{DC}(x(t)) & ... & +qI_{DC}(x(t)) & ... \\
... & ... & ... & ... & ... 
\end{bmatrix}_{n \times n} 
\] (4.9)

Figure 4.3 Stamp for the \(B_{cm}\) vector.
4.4 PPV Calculation

The PPV calculation is the most challenging part in the described phase noise calculation routine. It becomes even more difficult to implement PPV computation, when periodic steady state analysis is not available as in the default version of SPICE3. Let,

\[ C(t) = \frac{d}{dx} q(x) \bigg|_{x=x_i} \]  
(4.10)

\[ G(t) = \frac{d}{dx} g(x) \bigg|_{x=x_i} \]  
(4.11)

Then the \( C \) and \( G \) matrices are the imaginary and real parts of a circuit matrix, respectively (Eqs. (4.10-11)). \( x \) is a vector of circuit variables, such as node voltages and currents through devices. The algorithm for PPV or \( v_f(t) \) calculation in the time domain [13] is as follows:

1) Compute the periodic steady-state solution \( x_f(t) \) for one period \( 0 \leq t \leq T \) by integrating Eq. (4.1). At each time point, the \( C \) and \( G \) matrices have to be saved, state-dependent noise sources have to be evaluated and the matrix \( B_w \) and vectors \( B_{cm} \) have to be stamped and saved. The integration can be done using a steady-state or transient analysis. In case of using transient analysis we need to run it for a long enough time to ensure that the circuit reaches a steady state. Also from the transient analysis, the period of oscillation \( T \) has to be calculated.

2) Compute the monodromy matrix \( \Omega(-T,0) = Y(-T) \) by numerically integrating:

\[ C^T(t) \frac{d}{dt} Y - G^T(t)Y = 0 \]  
(4.12)
backward in time for the time interval \( t=\{0...-T\} \). As an initial condition an identity matrix can be used: \( Y(0)=I_n \). Forward integration in time can lead to numerical instability.

3) Perform a complete eigenvector-eigenvalue analysis of the monodromy matrix. \( v_j(0) \) is an eigenvector of the monodromy matrix \( \Omega(-T,0) \) which corresponds to the eigenvalue 1. If there are several eigenvalues close to one, it is hard to distinguish them numerically, and the inner product needs to be calculated for each eigenvector:

\[
v_j(0)^T C(0) \dot{x}_j(0) = (4.13)
\]

The correct eigenvector is then the one that has the largest inner product. Finally \( v_j(0) \) has to be scaled in order to satisfy:

\[
v_j(0)^T C(0) \dot{x}_j(0) = 1
\] (4.14)

4) Compute the periodic vector \( v_j(t) \) (PPV) by numerical integration of the adjoint system:

\[
C^T(t) \frac{d}{dt} y - G^T(t)y = 0
\] (4.15)

backward in time for the time interval \( t=\{T...0\} \). Use \( v_j(T)=v_j(0) \) as an initial condition.
4.5 Integration Method for PPV Calculation

The Gear 2 method, given by Eq. (4.16), was chosen for integration of Eqs. (4.12) and (4.15). For a uniform time step

\[
\dot{Y}_n = \frac{1}{h} \left( \frac{3}{2} Y_n - 2Y_{n-1} + \frac{1}{2} Y_{n-2} \right)
\]  
(4.16)

where \( h \) is the integration step and \( n \) is the step number.

Since the Gear 2 method needs to use function values from two previous time points, the first step of integration uses the Backward Euler (BE) method.

\[
\dot{Y}_n = \frac{1}{h} (Y_n - Y_{n-1})
\]  
(4.17)

Eq. (4.12) can be represented in the discrete time domain as:

\[
C_n^T \dot{Y}_n - G_n^T Y_n = 0
\]  
(4.18)

Applying the integration formula (4.17) for the first step, and (4.16) for the other steps, to the problem given in (4.18), a system of linear algebraic equations can be obtained with \( Y_n \) as a vector of unknowns. For the first step Equation (4.19) has to be solved. For the rest of the time steps, Equation (4.20) has to be solved. LU decomposition techniques can be used to solve these equations.

\[
\begin{pmatrix}
\frac{1}{h} C_n^T - G_n^T \\
\end{pmatrix} Y_n = \frac{1}{h} C_n^T Y_{n-1}
\]  
(4.19)
\[
\left(\frac{\alpha_0}{h} C_n^T - G_n^T\right)Y_n = -\frac{1}{h} C_n^T \left(\alpha_1 Y_{n-1} + \alpha_2 Y_{n-2}\right)
\]  
\hspace{1cm} (4.20)

Since time steps for the integration of Eqs. (4.12) and (4.15) are given by transient analysis, it is impossible to change the step during integration, and thus we cannot control local error on the fly. But it is possible to calculate the local error and give a warning if it is more than a specified threshold. A local error for the Gear 2 method at the \(n\)-th step is given by:

\[
LE = -\frac{1}{3} h^3 \dddot{Y}_n
\]  
\hspace{1cm} (4.21)

### 4.6 Overall Implementation

The phase noise calculation procedure described in the earlier sections was implemented in SPICE3. The overall flowchart of the implementation is shown in Figure 4.4. The first part of the algorithm was implemented in the transient analysis routine, where all the necessary data was collected and saved. The second part of the algorithm, that deals with the phase noise calculation itself, was implemented as a separate analysis (pnoise analysis). The sequence of operations is as follows. First, a transient analysis is run for a long enough time to ensure that the circuit reaches a steady state. During transient analysis, the period of oscillation \(T\) has to be calculated. When the period of oscillation is known, a time interval of one period has to be taken to save all the necessary data for phase noise calculation. At each time point of this time interval, the \(C\) and \(G\) matrices have to be saved, state-dependent noise sources have to be evaluated and the matrix \(B_w\) and the vectors \(B_{cm}\) have to be stamped and saved.
Figure 4.4 A flowchart of the phase noise analysis implementation in SPICE3.
The phase noise calculation routine starts with the perturbation projection vector calculation and then the scalar $c$ is calculated. Finally, the single-sideband phase noise spectrum $L(f_m)$ in $dBc/Hz$ can be calculated using Eq. (4.5).

SPICE3 is an unsupported public domain software, and hence, has bugs. There are a number of patches developed to fix known bugs in SPICE3. Therefore, work should be done to apply those patches to our SPICE3 version. Several bugs were found during the implementation of phase noise analysis in SPICE3. Some of these bugs and possible ways for fixing are described in Appendix A.
In order to validate the phase noise analysis that has been implemented in SPICE3, several different circuits with the MOS level-one and BSIM3 models were simulated. The results were compared with those obtained from SpectreRF. SpectreRF is a commercial simulator and has been validated by measurement data, hence it is considered to be the reference for basic oscillator circuits.

5.1 Testing of Different Noise Models

The BSIM3 transistor model has a capability to choose different models for simulating flicker and channel thermal noise sources [16]. There are two models for the flicker noise named as SPICE2 flicker noise model and the BSIM3 flicker noise model. Also, there are two models for the channel thermal noise. Similarly, they are named as SPICE2 channel thermal noise model and the BSIM3 channel thermal noise model. The type of models can be switched by the “noimod” flag. All possible combinations for the “noimod” flag are summarized in Table 5.1. To test the SPICE3 phase noise implementation for different sets of models, a three-stage ring oscillator (Figure 5.1) with inverter based delay cells (Figure 5.3) was chosen. For the simulations, the BSIM3 flicker noise model was used with default parameters. And for SPICE2 flicker noise model the $K_f$ parameter was set to the value of 1e-27.
Table 5.1 “Noimod” flag for different noise models.

<table>
<thead>
<tr>
<th>noimod flag</th>
<th>Flicker noise model</th>
<th>Thermal noise model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPICE2</td>
<td>SPICE2</td>
</tr>
<tr>
<td>2</td>
<td>BSIM3</td>
<td>BSIM3</td>
</tr>
<tr>
<td>3</td>
<td>BSIM3</td>
<td>SPICE2</td>
</tr>
<tr>
<td>4</td>
<td>SPICE2</td>
<td>BSIM3</td>
</tr>
</tbody>
</table>

Table 5.2 Phase noise simulation results for the three-stage ring oscillator.

<table>
<thead>
<tr>
<th>“noimod” flag</th>
<th>Phase noise at 1kHz offset frequency</th>
<th>Phase noise at 1MHz offset frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SpectreRF</td>
<td>SPICE3</td>
</tr>
<tr>
<td>1</td>
<td>-47.97</td>
<td>-48.08</td>
</tr>
<tr>
<td>2</td>
<td>-53.54</td>
<td>-52.88</td>
</tr>
<tr>
<td>3</td>
<td>-53.55</td>
<td>-52.88</td>
</tr>
<tr>
<td>4</td>
<td>-47.97</td>
<td>-48.09</td>
</tr>
</tbody>
</table>

As seen from Table 5.2, a good agreement between both simulators is obtained for all the different choices of the “noimod” flag.

Figure 5.1 Three-stage ring oscillator.
5.2 Thirteen-stage Ring Oscillator

A thirteen-stage ring oscillator (Figure 5.2) with delay cells, shown in Figure 5.3, was simulated in SPICE3 and in SpectreRF. The oscillator has an oscillation frequency of 28MHz when the MOS transistors are modeled as level-one models, and 96MHz when the BSIM3 model from the TSMC 0.35μm process is used. The "noimod" flag has been set to 1, and $K_f = 1\times10^{-27}$.

![Figure 5.2 Thirteen-stage ring oscillator.](image1)

![Figure 5.3 Inverter based delay cell.](image2)
Figure 5.4 Phase noise of the thirteen-stage ring oscillator with MOS level-one transistor models.

Figure 5.5 Phase noise of the thirteen-stage ring oscillator with BSIM3 transistor models.
Figure 5.4 shows the results for the thirteen-stage ring oscillator with the MOS level-one transistor models. The same results for the circuit with the BSIM3 transistor model are presented in Figure 5.5. As can be seen from these plots, SPICE3 matches SpectreRF very closely. The difference in the simulated values is less than 1.0dBc/Hz.

5.3 NMOS Cross-coupled Oscillator

A NMOS cross-coupled oscillator shown in Figure 5.6 was simulated in SPICE3 and in SpectreRF. The oscillator has an oscillation frequency of 2.4GHz when the MOS transistors are modeled by level-one models, and 2.1GHz, when the BSIM3 model from the TSMC 0.35μm process is used. The “noimod” flag has been set to 1, and $K_f = 1e-27$.

![NMOS Cross-coupled Oscillator](image)

Figure 5.6 NMOS cross-coupled oscillator.
Figure 5.7 Phase noise of the NMOS cross-coupled oscillator with MOS level-one transistor model.

Figure 5.8 Phase noise of the NMOS cross-coupled oscillator with the BSIM3 transistor model.
Figure 5.7 shows the results for NMOS cross-coupled oscillator with the MOS level-one transistor model. The same results for the circuit with the BSIM3 transistor model are presented in Figure 5.8. As can be seen from the plots, the results from SPICE3 match those from SpectreRF very closely.

5.4 Three-stage Differential Ring Oscillator with Buffer Circuit

A more complicated example, a three-stage differential ring oscillator shown in Figure 5.9 with the Maneatis load delay cells, shown in Figure 5.10, was simulated in SPICE3 and in SpectreRF with and without the buffer in Figure 5.11. The oscillator has an oscillation frequency of 303MHz. MOS transistors are modeled by the BSIM3 model from the TSMC 0.35μm process. The “noimod” flag has been set to 2, and the BSIM3 flicker noise model has been used with default parameters.

![Figure 5.9 Three-stage differential ring oscillator.](image)

Figure 5.10 Maneatis load delay cell.

Figure 5.11 Buffer for differential circuits.
Figure 5.12 Phase noise of three-stage differential ring oscillator with and without buffer.

As can be seen from Figure 5.12 both of simulators, SpectreRF and SPICE3, are consistent in simulations of the oscillator phase noise with and without the buffer circuit. The phase noise before and after a buffer circuit remains the same. Also, the results from SPICE3 match those from SpectreRF very closely. This example demonstrate the capability of SPICE3 to simulate the phase noise of larger circuits.
Optimization programs for the computer-aided design of on-chip spiral inductors and integrated RF VCOs have been developed. Carefully chosen optimization techniques allow for efficient automated design. The inductor optimization program generates the layout of the inductor for a given set of specifications with a maximal quality factor. This program can be used in conjunction with any electromagnetic solver by adding proper software interface. The VCO optimization program optimizes the circuit parameters and the inductor layout for minimum phase noise. This program combines the circuit simulator SpectreRF with the electromagnetic solver ASITIC. Results from these programs demonstrate the benefits of using automated design tools for circuit design.

Phase noise analysis based on a non-linear perturbation analysis for oscillators has been implemented in the circuit simulator SPICE3. The implemented technique allows for an accurate simulation of phase noise due to devices described either by analytical or numerical models. Furthermore, the technique is available in a public domain software and can be extended to other application domains.

Future work should focus on the further development of techniques for circuit optimization. New techniques should include more physical insight and rely on global optimization methods. All techniques, proposed in this thesis are based on local optimization methods, hence sub-optimal results may be obtained.

The phase noise calculation technique should be implemented on top of a periodic steady-state analysis, such as a shooting Newton or harmonic balance based method. This will improve the accuracy and reduce the time for phase noise simulations. The current implementation considers one period of transient analysis as a steady-state solution, but there is no obvious criterion to determine how long a
transient analysis has to be run for the circuit to reach the steady state. This issue could lead to incorrect results.

The implementation of phase noise analysis in SPICE3 enables the accurate phase noise prediction of circuits simulated with numerical device models. A good application example is micro electro-mechanical systems (MEMS). RF MEMS-based VCOs have better phase noise performance than VCOs with traditional tanks [18]. However, there is no tool for accurate phase noise simulation of such circuits. The SPICE3 phase noise analysis developed as part of this work can be combined with a MEMS solver [19] for accurate phase noise simulation of RF MEMS VCOs.
BIBLIOGRAPHY


APPENDIX A SPICE3 bugs report

This appendix contains the explanation of several bugs in the circuit simulator SPICE3 and its standard device models. All these bugs were found during the implementation of phase noise analysis in SPICE3. Possible ways of fixing the bugs are also presented.

A.1 Objects Cleaning Routine Bug

In the home version, spice3f5 has an error included during the addition of the BSIM3 model. In the file /util/skeleton/make_def.bd, in the definition of the variable ALL_DEVICES (lines 117-119) missing bsim3 statement. To fix that, one can add "bsim3" in that code fragment. It should be:

```
117   ALL_DEVICES = asrc bjt bsim1 bsim2 bsim3 cap cccs ccvs csw dio ind isrc
117   jfet ltra mes mos1 mos2 mos3 mos6 res sw tra urc
119   vccs vcvs vsrc
```

The variable ALL_DEVICES is used only when a user wants to remove all compiled objects. When ./util/build is called with "clean" option, the make script will refer to ALL_DEVICES to take a list of folders with device objects to be cleaned. If "bsim3" is not specified in ALL_DEVICES, BSIM3 objects will not be removed, therefore they will not be recompiled with updated header files. This can lead to incorrect program operation.
A.2 BSIM3 Flicker Noise Model Bug

There is a bug in the BSIM3 flicker noise model. This bug was found in Spice3f5 with b3noi.c,v 3.1 96/12/08. The following netlist was used to perform a simple test and show that the bug really exists. The netlist represents a simple circuit: NMOS common source amplifier with resistive load.

Netlist:
* simple test
mln d g 0 0 mod1 l=1u w=100u ad=1e-10 as=1e-10 pd=202u ps=202u
vds dd 0.15
rr dd d 1
vgs g 0 4.3 ac 1
.op
.ac dec 10 1k 10meg
.noise v(d) vgs dec 10 1k 10meg 10
.model modl nmos level=8
+ tnom=27 tox=7.6E-9
+ xj=1.5E-7 nch=1.7E17 vth0=0.579412
+ k1=0.6443761 k2=3.081977E-4 k3=12.4576649
+ k3b=9.2807835 w0=1.663064E-5 nlx=2.715803E-7
+ dvt0w=0 dvt1w=5.3e6 dvt2w=-0.032
+ dvt0=14.5206288 dvt1=0.6981272 dvt2=-3.114278E-3
+ u0=400.4844971 ua=1.592734E-10 ub=1.613884E-18
+ uc=4.411969E-11 vsat=1.652718E5 a0=1.0000129
+ ags=0.2328181 b0=6.944434E-7 b1=5E-6
+ keta=7.06235E-3 a1=0 a2=1
+ rdsw=644.5019706 prwg=0.0585705 prwb=-1E-3
+ wr=1 wint=1.009819E-7 lint=5.443809E-8
+ dwg=6.733729E-9 dwb=1.536201E-9 voff=-0.1435754
Spice simulation result for the flicker noise of "m1n":

Spice 4 -> print onoise.m1n.1overf
* simple test
Noise Spectral Density Curves - (V^2 or > A^2)/Hz Mon Apr 28 16:12:30 2003

<table>
<thead>
<tr>
<th>Index</th>
<th>frequency</th>
<th>onoise.m1n.love</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.000000e+03</td>
<td>-6.413308e-20</td>
</tr>
</tbody>
</table>
As we can see, noise power spectral density is negative. For the same circuit, SpectreRF gives noise @ 1k = 6.41114e-20 V^2/Hz. This bug can be fixed by the following line: “noizDens[BSIM3FLNOIZ] = fabs( noizDens[BSIM3FLNOIZ] )” in the file b3noi.c line #300:

```
298  break;
299  }
300  noizDens[BSIM3FLNOIZ] = fabs( noizDens[BSIM3FLNOIZ] ); /*bug fixing */
301  lnNdens[BSIM3FLNOIZ] =
302  log(MAX(noizDens[BSIM3FLNOIZ], N_MINLOG));
```

### A.3 MOS1 Flicker Noise Model Bug

Another bug was found in the MOS1 flicker noise implementation (file mos1noi.c). The MOS1 flicker noise model, according to documentation [17], is given by:

\[
\bar{i}_{flick}^2 = \frac{\eta f_{DS}^2 \Delta f}{C_{ox} W_{eff} L_{eff} f_{ref}} 
\]  

(A.1)

However, the implementation in SPICE3 was with \(C_{ox}^2\) instead of \(C_{ox}\), i.e.,
This can be seen from the code fragment:

```c
64  if (model->MOS1oxideCapFactor == 0.0) {
65      coxSquared = 3.9 * 8.854214871e-12 / 1e-7;
66  } else {
67      coxSquared = model->MOS1oxideCapFactor;
68  }
69  coxSquared *= coxSquared;
```

```c
145  noizDens[MOS1FLNOIZ] *= model->MOS1fNcoef * 
146      exp(model->MOS1fNexp * 
147        log(MAX(FABS(inst->MOS1cd),N_MINLOG)) / 
148        (data->freq * inst->MOS1w * 
149        (inst->MOS11 - 2*model->MOS1latDiff) * coxSquared));
```

To fix this bug, one can comment line #69.