AUTOMATIC CIRCUIT SYNTHESIS

A Project Report

submitted by

PYNENI ROOPESH

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 27, 2022

THESIS CERTIFICATE

This is to certify that the thesis titled **AUTOMATIC CIRCUIT SYNTHESIS**, submitted by **Pyneni Roopesh (EE18B028)**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the research work done by him under the supervision of **Dr Sankaran Aniruddhan**. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. S Aniruddhan
Research Guide
Professor
Dept. of Electrical Engineering
IIT Madras, 600 036

Place: Chennai

Date: 27th May 2022

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my guide Dr. Sankaran Aniruddhan for your patience, guidance, and support. I have significantly benefited from your wealth of knowledge, and I am incredibly grateful that you took me on as a student and guided me.

I would also like to thank Pavan Sumanth for our numerous discussions during this project. I offer my sincere thanks to all other persons who knowingly or unknowingly helped me complete this project.

ABSTRACT

In this project, the design of an RF circuit is automated. A python code is written that will work along with circuit simulators to optimize the performance of an LNA. The code will perform the optimization for a given circuit topology, MOS technology, and an output specification. We will discuss the optimization of two circuit topologies, namely common source LNA and common gate LNA.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

1.1.1 Issues

In the present day, analog circuit designers design circuits manually by tweaking circuit parameters until the outputs meet a given specification. They use circuit simulators to verify their circuit designs. They optimize the circuit iteratively by changing some parameters and running the circuit in the simulator. They repeat this process until they reach a point where they cannot improve the circuit further. However, this is a highly automatable process. High speed computer can be used to perform circuit optimization. They can do this iterative process in short time. Automating this process will help save time as machines are quicker than humans. It will also save the manpower required to design circuits.

1.1.2 Solving the issue

Our project is based on the automation of RF circuit design. We write a code that designs a circuit which satisfies a given output conditions across different process, temperature, and over a given range of frequencies. The code should choose the value of different circuit components such as resistors, capacitors, MOSFETs, etc., such that we minimize power consumption and at the same time, satisfying certain output conditions. The code will take inputs as circuit topology, MOS technology, and circuit specifications. It will give values of circuit parameters as the output.

1.2 Approach to the Problem

1.2.1 Method to optimize the circuit

To optimize the circuit, we can use gradient descent algorithm. This algorithm will optimize the circuit in steps, with each step optimizing the circuit slightly more. We first have to select an initial point for this algorithm, i.e, a starting point with values for each circuit component. This is called **pre-optimization** step. Next, we will optimize the circuit starting from this initial point using gradient descent algorithm. This is called **optimization** step. This step will give the final circuit parameters.

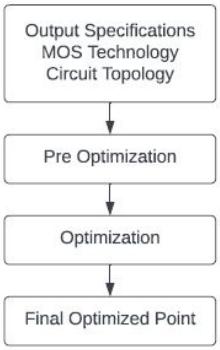

Figure 1.1: Code Structure

1.2.2 Implementation

We will write a code in python that will perform circuit optimization. This code will first call the pre-optimization function. This function will set the initial point of the circuit. Then, the code will call the optimization function. This function will perform gradient descent for multiple iterations, returning the final optimized point. The code wilal store these output results in a text file. To run the circuit and find the output parameters such as gain, iip3, nf, etc, this code will work with spectre circuit simulator. It will modify the netlist file, run the circuit using shell commands, and then read the outputs from the text files.

1.3 Pre-Optimization

This is the step that will set the initial point of the circuit before we start performing optimization. Choosing a good initial point is crucial as it will allow optimization to occur smoothly. Suppose we do not choose a good initial point, optimization will take much longer to optimize to the final optimized point. There are different algorithms that we will use for pre-optimization:

- Manual Initial Point
- Automatic Initial Point
- Initial Point Update

1.3.1 Manual Initial Point

For this step, we will choose the initial point manually. This will allow us to choose the initial point ourselves and run optimization at a starting point, which we have chosen manually. We can only use manual initial point as a standalone part of pre-optimization. For this step, we will set the initial circuit parameters as a dictionary in the python code. The code will then run the circuit with these point and store the output results. This will be used as the starting point for optimization.

1.3.2 Automatic Initial Point

For this step, we will use a code to choose the initial point. This code will have the circuit specifications (such as IIP3, NF, Gain, s11) as the input. For example, gain=gm*Rd. If we know the value of gm and gain, the value of Rd can be set. We will use numerous circuit equations such as s11, gain, and nf to set the initial circuit parameters, such as MOS width, length, Io, etc. This can be used standalone or along with initial point update.

1.3.3 Initial Point Update

This has to be used only after automatic initial point. It will update certain circuit parameters, which can be further fine tuned after automatic initial point step. For example, in a CS LNA, we might not be able to choose Lg and Ls precisely to get a good s11. Initial point update will make these small changes to better fine tune the circuit before optimization. While automatic initial point step is done only once, we can repeat initial point update step multiple times.

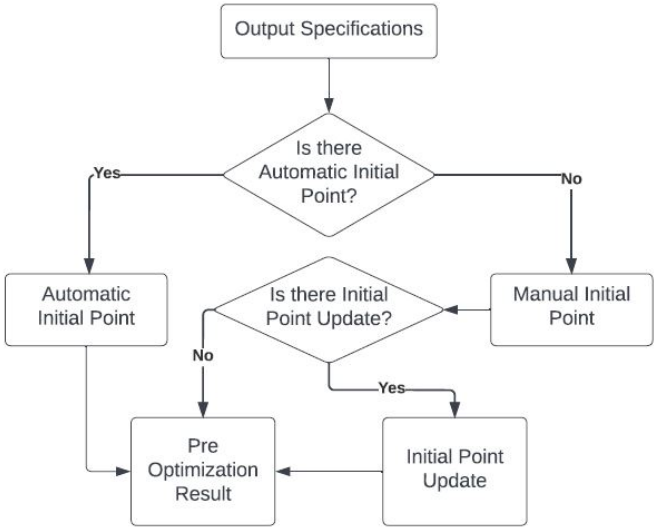

Figure 1.2: Pre Optimization Structure

1.4 Optimization

For optimization, we will use gradient descent algorithm. This algorithm will change the value of circuit parameters during each iteration. It will change the parameters such that a loss function is minimized. We will have to choose an appropriate loss function for our circuit.

1.4.1 Gradient Descent Algorithm

For every iteration, we will calculate the gradient of slope for each parameter. After we calculate the gradient, we will vary the parameters.

$$P = P - P^2 * \alpha \frac{\partial loss}{\partial P} \tag{1.1}$$

We can't find $\frac{\partial loss}{\partial P}$ exactly. So, we will calculate $\frac{\Delta loss}{\Delta P}$, where ΔP=0.001P

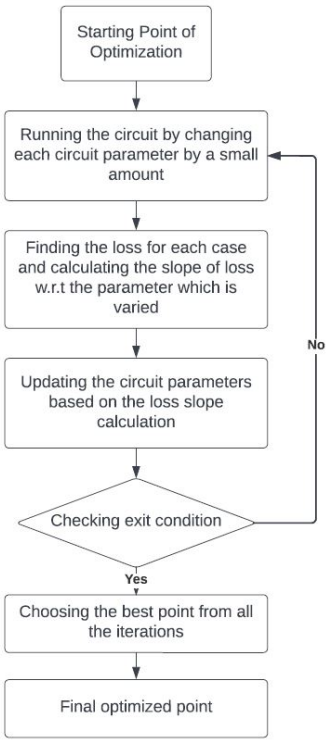

Figure 1.3: Optimization Structure

1.4.2 Loss Function

The loss function will describe how good our circuit is. For a given loss function, a lower loss value means we have a better circuit. We choose loss functions such that lower current will have a lower loss, higher NF will have a higher loss, etc,. Selecting a good loss function is critical to optimizing the circuit in the best way.

CG LNA

Loss =

A1*ramp(Gain$_{ref}$-Gain) +

A2*ramp(IIP3$_{ref}$-IIP3) +

A3*ramp(s11$_{ref}$-s11) +

A4*ramp(NF$_{ref}$-NF) +

A5*Io

CS LNA

Loss =

A1*ramp(Gain$_{ref}$-Gain) +

A2*ramp(IIP3$_{ref}$-IIP3) +

A3*ramp(s11$_{ref}$-s11) +

A4*ramp(NF$_{ref}$-NF) +

A5*Io +

A6*(Gain$_{delta_0}$) + A6*(Gain$_{delta_2}$) +

A7*ramp(s11$_{middle_{ref}}$-s11$_{middle}$) +

A8*|Gain$_{fo-\Delta f}$ - Gain$_{fo+\Delta f}$|

here, ramp(x)=

- x if x $>$ 0
- 0 if x $<$ 0

We need to choose the coefficients (A1, A2, ..., A8) appropriately. A1, A2, A3, A4, A6, and A7 will be much higher than A5 and A8. This is because we want those terms to be exactly zero. Though we want Gain$_{fo-\Delta f}$ - Gain$_{fo+\Delta f}$ to be zero, this will not always happen. So, if we choose a higher value of A8, then we will not be able to optimize Io very well.

1.4.3 Choosing Loss Coefficients

We need to choose loss coefficients such that the circuit gets optimized well. Suppose we choose a very small loss coefficient for a particular parameter and a high loss coefficient for another parameter. In that case, the second parameter gets optimized well, and the first parameter does not get optimized well.

To avoid over-optimization of a particular parameter, we will set the loss coefficient as the inverse of the output specification of that parameter.

Table 1.1: CG LNA Loss Coefficients

Parameter Name	Loss Name	Value
A1	Gain	0.1
A2	IIP3	0.2
A3	S11	0.067
A4	NF	0.25
A5	Io	100

Table 1.2: CS LNA Loss Coefficients

Parameter Name	Loss Name	Value
A1	Gain	0.05
A2	IIP3	0.1
A3	S11	0.1
A4	NF	0.25
A5	Io	100
A6	Gain Delta	0.1
A7	S11 middle	0.1
A8	Gain Delta 2	0.01

1.4.4 Optimization Specifications

While doing optimization, we need to decide the value of alpha, the number of iterations, etc. We will list out the different parameters that we should set before optimization.

Table 1.3: Parameters for optimization

Parameter Name	**Meaning**
n_runs***	This is the number of optimization runs
max_iteration	Max no. of iterations for each run.
consec_iter	Max no. of consecutive iterations where loss increases before we stop optimization. For ex: if the value is 3 and the value of loss increases during 3 consecutive optimization steps, we will stop optimization.
alpha_min	Min value of alpha before we stop optimization. The value of alpha can change during optimization, and we can set a minimum value of alpha before we stop optimization.
alpha_mult	Value by which we should multiply alpha if loss increases during an optimization step (this value will be between 0 and 1). Ex: when the value of loss increases, it means that the value of alpha is too big. So, we will choose alpha_mult between 0 and 1 to reduce the value of alpha for the next optimization iteration.
delta_threshold	The amount by which a parameter should be increased to calculate slope (usually 0.001) i.e,ΔP=delta_threshold*P
loss_type*	Specifies which loss values should be considered when we perform gradient descent.
optimization_type	This will mention if we should try to increase or decrease the loss function. We will try to reduce the value of loss and increase the value of FoM.
optimizing_parameters	This is the list of parameters that get optimized during the circuit optimization step.
loss_weights	This dictionary will contain the value of loss weights (A1, A2, ...)
alpha**	This is a dictionary that will tell how to vary values of alpha during optimization.

Notes:

* loss_type can take 3 values : 0, 1, and 2. These will indicate which loss values to consider for optimization.

- 0 - Considers slope of loss of Io when loss of all the other parameters is 0 ; Considers slope of loss of other parameters when loss of all the other parameters is not zero
- 1 - Considers slope of loss of all parameters always
- 2 - Considers slope of loss of those parameters whose loss is non-zero

** alpha is a dictionary that contains 4 values

- Type - This can take either of "Normal", "Log", or "Linear"
- Value - This is the starting value of alpha for "Normal" type
- Start - This is the starting value of alpha for "Log" and "Linear" type
- End - This is the ending value of alpha for "Log" and "Linear" type

The "Normal" type is used to have a constant value of alpha for all the iterations (with the exception when alpha_mult not equal to 1).
The "Log" and "Linear" type is used if we want to pre-define how the value of alpha reduces over iterations. We can here choose alpha to reduce as a linear or a logarithmic function over iterations. The starting and ending value of alpha are chosen based on "start" and "end" values. In this case, we need to set alpha_mult equal to 1

*** We use n_runs to perform multiple optimization runs. Suppose we want to optimize the circuit initially by varying only certain parameters (say W, Io, and R) with a higher value of alpha. Then in a later run, we might want to vary many more parameters. For such cases, we will use multiple optimization runs. For each run, we need to choose the value of all the other parameters such as max_iteration, loss_weights, alpha, etc,.

1.4.5 Optimization Result

Usually, the last iteration will have the most optimized result for a gradient descent algorithm. But in our case, for circuit optimization, it might not always be true that the last iteration carries the best result. So, we need to find the best result by looking at all the iterations.

We need a criteria to judge which point is the best. We can't just say that the best point is the one with the least loss because we have imposed a condition for gain, s11, nf, and iip3. And if those conditions are not met, we can't say that a particular circuit is acceptable.

From among the loss functions, we will define the following:

- Zero Loss Array - This is the list of all the loss values that **must** be zero. The loss due to gain, nf, iip3, s11 are present in zero loss array.
- Minimize Loss Array - This is the list of all the loss values which are not present in zero loss array. We need to minimize this loss value and it does not necessarily need to be zero.

We will look at all the iteration results and see which iterations have zero loss for the loss values in zero loss array. Among these iterations, we will find which iteration has the minimum loss among the sum of losses of the minimize loss array parameters. This iteration which has the minimum loss is the final optimized point.
If none of the iterations have zero loss for the zero loss array parameters, then our circuit is not optimized.

1.4.6 Figure of Merit (FOM)

Apart from minimizing loss, we will also try to maximize FOM for the circuit. Loss minimization is done for a circuit when we are given the specifications and we have to minimize the power consumed. It is done when we want to design a circuit for a particular application.

On the other hand, FOM optimization is done when we want to evaluate a MOS technology and see how much better specifications it can give compared to other MOS technologies. For FOM, we need to maximize the value, unlike loss where we try to minimize the value. The higher the value of FOM, the better the circuit is.

While maximizing FOM, we need to keep s11 lower than a certain threshold. When we are maximizing the FOM for CG LNA circuit, we will keep a limit of -15dB for s11.

$$\text{FOM} = \frac{\text{Gain*IIP3(in mW)}}{\text{(NF-1)*Power(in mW)}} \tag{1.2}$$

$$\text{FOM}_{dB} = \text{Gain}_{dB} + \text{IIP3}_{dBm} - \text{(NF-1)}_{dB} - \text{Power}_{dBm} \tag{1.3}$$

CHAPTER 2

RUNNING THE CIRCUIT

We need to run the circuit in a circuit simulator to get the extracted values. Initially, we used Eldo to run the circuit. But we later moved to Spectre Circuit Simulator. Here, we will look into how we run the circuit from the python code.

2.1 Introduction

We will have four basic dictionaries in our code that are used for running the circuit:

- Circuit Simulation Parameters
- Initial Circuit Parameters
- Circuit Parameters
- Extracted Parameters

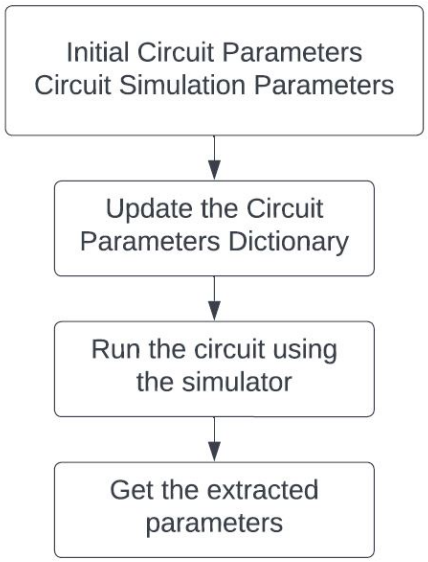

Figure 2.1: Circuit Run Structure

2.2 Circuit Simulation Parameters

This dictionary will contain different information needed to run the circuit in the simulation software. It will contain information such as :

- The file directory of the circuit netlist files
- The different frequencies at which to run the circuit
- The different process corners of the MOSFETs
- The different temperatures at which to run the circuit
- The type of circuit components (whether they are ideal or real)
- Parameters for iip3 calculation (range of p_{in}, whether to choose single or multiple p_{in}, number of harmonics)
- Maximum width of each finger for MOSFET

2.3 Initial Circuit Parameters

This dictionary contains the parameter values of different parts of the circuit such as resistors, capacitors, inductors, MOSFET sizes, and current source. However, it does not contain the values of width and length of resistors and capacitors when we use real components instead of ideal components.

2.4 Circuit Parameters

Circuit parameters will get updated just before executing the circuit in the circuit simulator software. Write circuit parameters function is called to update the circuit parameters dictionary. Circuit parameters is a copy of initial circuit parameters with some changes. These changes are for two reasons:

1. Setting limits for parameter values
2. Setting parameter values for non-ideal components

2.4.1 Setting Limits

We know that for our circuit, there are some limits on certain parameters.
For example, we can't have an inductor greater than 9nH when using TSMC 65nm inductors as that is the higher limit. So, if the inductance value is greater than 9nH in initial circuit parameters, we will set it to be 9nH in circuit parameters.
Also, we can't have capacitance, resistance, and inductance values less than zero. If this is the case, we will assign a minimum value of R, L, and C in the circuit parameters dictionary.

2.4.2 Setting Parameters Values for non-ideal components

For ideal circuit components such as resistors, capacitors, and inductors, the initial circuit parameters will only contain the component's value. But, for real components, we need other values. For example, we need to choose the value of W and L for resistors and capacitors. We need to choose W, L, radius, spacing, and the number of turns for inductors. These parameters are calculated and stored in the circuit parameters dictionary.

2.5 Extracted Parameters

This dictionary will get updated once we run the circuit simulator. It will store the values of output parameters such as gain, iip3, s11, NF, etc.
For CS LNA, we run the circuit at multiple frequencies, multiple temperatures, and different processes. So, the extracted parameter dictionary will store all those values. But, we will also need to choose a final, singular value of Io, Gain, IIP3, NF, and s11 to be used for loss calculation. To do this, we will follow the following procedure:

2.5.1 Method to Choose Extracted Parameters

Initially, we will get the extracted parameters at each frequency, process, and temperature. Let's say that we run the circuit at n_f frequencies, n_p processes, and n_t temperatures. So, we will get n_f*n_p*n_t python dictionaries. We need to combine these dictio-

naries to get a single dictionary. First, we will combine all the extracted parameters for all frequencies for each temperature and process to get n_p*n_t dictionaries. Secondly, we will combine all the extracted parameters for all processes for each temperature to get n_t dictionaries. Finally, we will combine all the extracted parameters for all temperatures to get a single dictionary.

2.5.2 Combining Parameters for all frequencies

- For DC parameters, we will only store them once. (Example: voltages at different nodes, Id of MOSFETs, MOSFET Operation Region)
- For AC parameters, we will store the values at all frequencies
- For Gain, s11, iip3, and NF, we will store the worst case value separately. (For ex, the worst case gain is the least gain. Worst case s11 is the highest s11.)
- For gain, we will store the difference in magnitude of gain between the highest and lowest frequency.
- For gain, we will store the ramp function of the difference in gain between edge frequencies and the centre frequency.

2.5.3 Combining Parameters for all process

- For all parameters, store them at different process corners.
- For Gain, s11, iip3, and NF, store the worst case value.
- For Io, store the value at TT corner.

2.5.4 Combining Parameters for all temperatures

- For all parameters, store them at different temperatures corners.
- For Gain, s11, iip3, and NF, store the worst case value.
- For Io, store the value at 27^oC.

CHAPTER 3

MOS TECHNOLOGIES

3.1 List of MOS Technologies

For our project, we will use 3 MOS Technologies. They are:

- TSMC 180nm
- IBM 130nm
- TSMC 65nm

3.2 MOSFET Characteristics

Table 3.1: MOSFET Characteristics

Parameter	TSMC 180nm	IBM 130nm	TSMC 65nm
Vdd (V)	1.8	1.3	1.0
L_{min} (nm)	180	130	60
μ_o (cm^2/V)	273.8	434.7	212.1
t_{ox} (nm)	4.1	3.2	2
C_{ox} (fF/μm^2)	8.42	10.8	17.2
$\mu n.C_{ox}$ ($\mu A/V^2$)	230.5	469.4	364.8
v_{th_o} (v)	0.366	0.057	0.317

3.3 MOSFET Usage

We will use TSMC 180nm and IBM 130nm when working with Eldo initially for CG LNA circuit. We can't use these packages for non-ideal components. We will use TSMC 65nm process for CG LNA just before we progress onto using real resistors and real capacitors.

For CS LNA, we will initial use IBM 130nm and later move on to TSMC 65nm process.

CHAPTER 4

REAL COMPONENTS

When we run the circuit, we need to use real resistors, capacitors, and inductors. For these components, we will use the TSMC 65nm technology.

4.1 Resistors

4.1.1 Resistor Selection

We will choose rnpolywo resistor from the TSMC 65nm technology.

- It has a wide range of resistance values: 10 Ω to 31 kΩ.
- It has a low temperature coefficient and small variation with process corner.

4.1.2 Resistor Parameters Selection

The initial circuit parameters will contain the value of the resistance. From this resistance, we need to choose the value of W and L.
For rnpolywo, the value of resistance is:

$$R = R_{sheet} * \frac{L}{W - \Delta W} \tag{4.1}$$

$$\Delta W = 69.1nm \tag{4.2}$$

We will always choose W=W$_{min}$=800μm.

$$L = (W_{min} - \Delta W) * \frac{R}{R_{sheet}} \tag{4.3}$$

4.2 Capacitors

We will use MIMCAPS and MOSFET as a capacitor. We will use MIMCAPS for signal paths whereas MOSFETs as capacitors for gate of cascode, where linearity of the capacitor is not of importance.

4.2.1 MIMCAP Parameters Selection

For MIMCAP, the variation of capacitance with W and L is complex. So, we will choose minimum W and L and vary mf (multiplicative factor).

$$mf = ceil(\frac{C}{C_{min}}) \tag{4.4}$$

where ceil(x) is the smallest integer larger than x.
We choose minimum width and length so that a unit change in mf results in a small change in capacitance. So, this allows us to make small changes in capacitance value.

4.2.2 MOS Capacitor Parameters Selection

For MOS capacitor, C=C_{ox}.W.L
here, W_{min}=120nm and L_{max}=20μm

W_{check}=$\sqrt{\frac{C}{Cox}}$

- If W_{check}>20μm:
  - L=20μm
  - W=$\frac{C}{Cox.L}$
- If 120nm<W_{check}<20μm:
  - W=W_{check}
  - L=L_{check}
- If W_{check}<120nm:
  - W=120mm
  - L=$\frac{C}{Cox.W}$

4.3 Inductors

For TSMC 65nm technology inductors, we need to choose multiple factors such as:

- width
- radius
- number of turns
- gdis
- spacing

Similar to capacitor, it is very difficult to choose the values of these parameters using the complex inductor model.
When we use spectre GUI, the spectre inductor finder is used to choose the inductor parameters' values. But, we did not have access to spectre inductor finder while running the python code. So, we will create our own inductor finder.

4.3.1 Process of designing an inductor finder

To design our inductor finder, we need to sweep the parameters and store the values of L and Q. We will then sort the data in ascending order of L. Next, we will write another code to choose the value of parameters by looking up the data we have already stored.

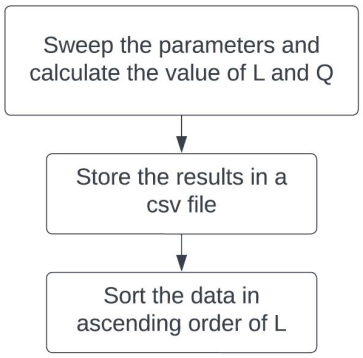

Figure 4.1: Storing the data for Inductor Finder

4.3.2 Making a table of Q and L values

We will linearly sweep the different parameters.

- Width : 3μm to 30μm - 10 values
- Radius : 15μm to 90μm - 10 values
- # turns : 0.5 to 5.5 - 11 values
- gdis : 10μm to 50μm - 5 values
- spacing : 2μm to 4μm - 3 values

We will find the value of Q and L at 1 GHz for these inductor parameters. We will store the values in a csv file.

4.3.3 Selecting the parameters for our desired inductor

When we want an inductor with a given Q and L, we will search all the values in this table, whose inductance is between 0.98*L to 1.02*L . We will see which value has the smallest value of inductance cost function : $(\frac{L-\hat{L}}{L})^2$ + $(\frac{Q-\hat{Q}}{Q})^2$. We will choose this as our inductor in the circuit.

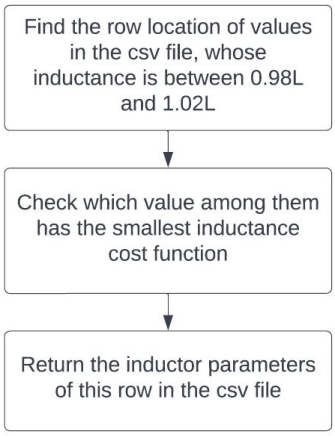

Figure 4.2: Getting the required inductor parameters

CHAPTER 5

COMMON GATE LNA

At the start of the project, we worked on the Common Gate LNA circuit and tried to optimize it. We designed a pre-optimization method for this circuit and then ran optimization. We performed optimization for only a single frequency, process corner, and temperature for this circuit.

5.1 Circuit

5.1.1 Circuit Diagram

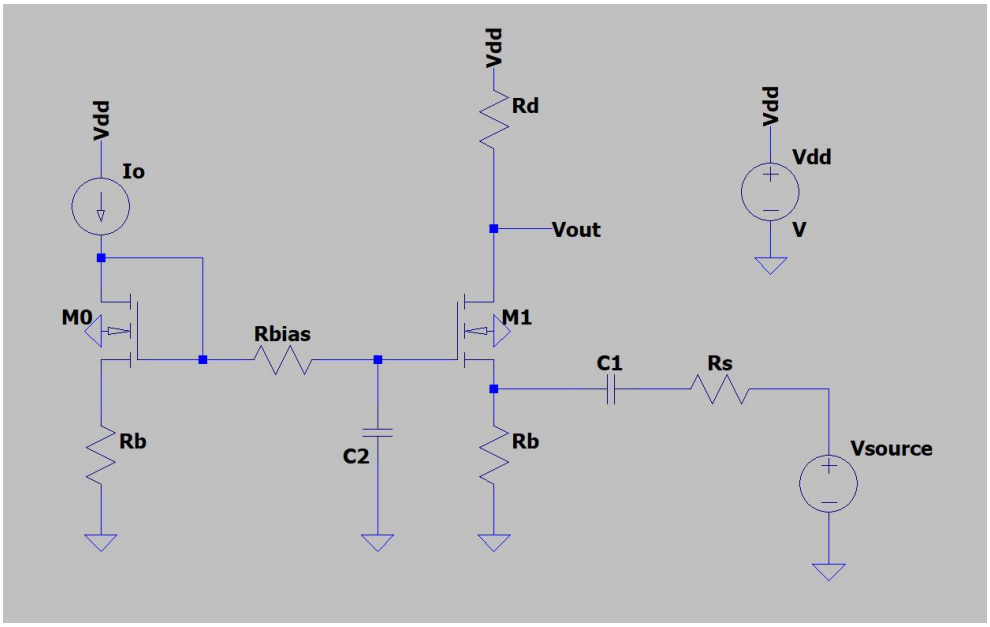

Figure 5.1: CG LNA Circuit Diagram

5.1.2 Circuit Equations

- Gain = gm*Rd
- gm=$\sqrt{2 * Io * \mu_n * Cox * \frac{W}{L}}$
- F=1+$\frac{Rs}{Rd}$*($\frac{1+gm.Rs}{gm.Rs}$)^2+$\frac{Rs}{Rb}$+$\frac{\gamma}{gm.Rs}$=1+$\frac{4.Rs}{Rd}$+$\frac{Rs}{Rb}$+γ

5.1.3 Output Conditions

Table 5.1: CG LNA Output Conditions

Parameter Name	Condition
Gain (dB)	$>$ 10.0
s11 (dB)	$<$ -15.0
NF (dB)	$<$ 4.0
IIP3 (dBm)	$>$ -5.0
Io (μA)	Minimize

5.2 Pre-Optimization

We have 3 stages of pre-optimization for CG LNA:

- Automatic Initial Point
- Initial Point Update
- Gm-Vdsat Optimize (or) Gm Optimize

5.2.1 Automatic Initial Point - 1

We will follow these steps to choose the value:

- **Choosing Io and W**
- Y=$\frac{2}{Rs}$*$10^{-|\frac{s11}{20}|}$
- W=Wmax=$\frac{3*Y}{2*wo*Lmin*Cox}$
- Io=$\frac{Lmin}{2*\mu n*Cox*Wmax*Rs^2}$

- **Choosing Rd**
- Rd1=2*Rs*Gain$_{required}$
- Rd2=$\frac{5*Rs}{F-1-\gamma}$
- Rd=max(Rd1,Rd2)

- **Choosing C1**
- C1=$\frac{threshold1}{wo*Rs}$

- **Choosing Rb**
- Gain=$\frac{Rd}{2*Rs}$
- P_{1dB}=IIP3(dBm)-39.6
- $vi_{swing,min}$=$\sqrt{8*Rs*P_{1dB}}$
- $vo_{swing,min}$=Gain*$vi_{swing,min}$
- vdsat=2*Io*Rs
- Rb1=$\frac{Vdd-Io*Rd-Vdsat-vo_{swing,min}}{Io}$
- Rb2=$\frac{Vdd-\Delta V-vdsat-v_T}{Io}$
- Rb=min(Rb1,Rb2)

- **Choosing C2 and Rbias**
- C2=$\frac{2}{3}$*W*L*Cox*threshold2
- Rbias=max($\frac{threshold3}{wo*C2}$,$Rbias_{min}$)

5.2.2 Automatic Initial Point - 2

We will follow these steps to choose the value:

- **Choosing Io and W**
- Io=$\frac{gm*vdsat_{required}}{2}$=$\frac{vdsat_{required}}{2*Rs}$
- W=$\frac{L}{Rs*\mu n*Cox*vdsat_{required}}$

- **Choosing Rd**
- Rd1=2*Rs*Gain$_{required}$
- Rd2=$\frac{5*Rs}{F-1-\gamma}$
- Rd=maxRd1,Rd2

- **Choosing C1**
- C1=$\frac{threshold1}{wo*Rs}$

- **Choosing Rb**
- Gain=$\frac{Rd}{2*Rs}$
- P_{1dB}=IIP3(dBm)-39.6
- $vi_{swing,min}$=$\sqrt{8*Rs*P_{1dB}}$
- $vo_{swing,min}$=Gain*$vi_{swing,min}$
- vdsat=2*Io*Rs
- Rb1=$\frac{Vdd-Io*Rd-Vdsat-vo_{swing,min}}{Io}$
- Rb2=$\frac{Vdd-\Delta V-vdsat-v_T}{Io}$
- Rb=minRb1,Rb2

- **Choosing C2 and Rbias**
- C2=$\frac{2}{3}$*W*L*Cox*threshold2
- Rbias=max($\frac{threshold3}{wo*C2}$,$Rbias_{min}$)

The difference between automatic initial point 1 and 2, is that in method 1, we do not worry about the value of vdsat. When we run the circuit, we find that the value of IIP3 is worse. But in method 2, the IIP3 is better because the circuit is optimized at a vdsat at which gm3 is low and IIP3 is good.

5.2.3 Initial Point Update

Here, we will update the value of Vt and use it make changes to the calculation of Rb. We will also calculate the value of Cgd and Cgs of M1. Using that, we will update the value of C2 and Rbias.

- Gain=$\frac{Rd}{2*Rs}$
- P_{1dB}=IIP3(dBm)-39.6
- $vi_{swing,min}$=$\sqrt{8 * Rs * P_{1dB}}$
- $vo_{swing,min}$=Gain*$vi_{swing,min}$
- vdsat=2*Io*Rs
- Rb1=$\frac{Vdd-Io*Rd-Vdsat-vo_{swing,min}}{Io}$
- Rb2=$\frac{Vdd-\Delta V-vdsat-v_T}{Io}$
- Rb=min(Rb1,Rb2)

- C2a=threshold2*Cgs1
- C2b=threshold2*Cgd1*$\frac{Rd}{2*Rs}$
- C2=max(C2a,C2b)
- Rbias=max($\frac{threshold3}{wo*C2}$,$Rbias_{min}$)

We will only perform initial point update once. We mainly do initial point update because it is difficult to accurately calculate the value of Vt and parasitic capacitances initially.

5.2.4 Gm Optimize

To get a good s11, we need gm=1/Rs=20mS. However, due to circuit non-idealities, the value of gm might not be exactly 20mS. So, we will make some changes to the circuit parameters to get gm closer to 20mS.

- k=$\frac{1}{gm*Rs}$
- if k>2, set k=2
- W=W*k^p
- Io=Io*k^{2-p}

- Gain=$\frac{Rd}{2*Rs}$
- P_{1dB}=IIP3(dBm)-39.6
- $vi_{swing,min}$=$\sqrt{8 * Rs * P_{1dB}}$
- $vo_{swing,min}$=Gain*$vi_{swing,min}$
- vdsat=2*Io*Rs
- Rb1=$\frac{Vdd-Io*Rd-Vdsat-vo_{swing,min}}{Io}$
- Rb2=$\frac{Vdd-\Delta V-vdsat-v_T}{Io}$
- Rb=min(Rb1,Rb2)

Here, if p is closer to 0, W will not change much.
If p is closer to 2, Io will not change much.
We will choose p=1 so that both W and Io changes.

We will set maximum value of k to 2 so that W and Io don't change by a lot after each iteration. If we don't set a value for k, Io and W might change by a huge number. We do not want such high changes. So, we will set a maximum value of 2.

5.2.5 Gm-Vdsat Optimize

To get a good s11, we need gm=1/Rs=20mS. However, due to circuit non-idealities, the value of gm might not be exactly 20mS. So, we will make some changes to the circuit parameters to get gm closer to 20mS. We will also try to get vdsat closer to the point where gm3 0 so that IIP3 is very good.

- k1=$\frac{1}{gm*Rs}$
- k2=$\frac{vdsat_{required}}{vdsat}$
- W=W*$\frac{k1}{k2}$
- Io'=Io*k1*k2

- Gain=$\frac{Rd}{2*Rs}$
- P_{1dB}=IIP3(dBm)-39.6
- $vi_{swing,min}$=$\sqrt{8 * Rs * P_{1dB}}$
- $vo_{swing,min}$=Gain*$vi_{swing,min}$
- vdsat=2*Io*Rs
- Rb1=$\frac{Vdd-Io*Rd-Vdsat-vo_{swing,min}}{Io}$
- Rb2=$\frac{Vdd-\Delta V-vdsat-v_T}{Io}$
- Rb=min(Rb1,Rb2)

5.2.6 Automatic Initial Point - 3

This method is mainly used for FOM Optimization. It will choose an initial point whose s11 is achieved by having Rb close to Rs.

- Rb=Rs
- C1=$\frac{threshold1}{wo*Rs}$
- Io=100 μA
- gm=2 mS
- W=$\frac{Lmin*gm^2}{\mu n*Cox*2*Io}$
- Rd=$\frac{Gain}{gm}$
- C2=$\frac{2}{3}$*W*Lmin*Cox*threshold2
- Rbias=max($\frac{threshold3}{wo*C2}$,Rbias$_{min}$)

5.2.7 Initial Point Update - 3

Here, we will calculate the value of Cgd and Cgs of M1. Using that, we will update the value of C2 and Rbias.

- C2a=threshold2*Cgs1
- C2b=threshold2*Cgd1*$\frac{Rd}{2*Rs}$
- C2=max(C2a,C2b)
- Rbias=max($\frac{threshold3}{wo*C2}$,Rbias$_{min}$)

Like before, we will only perform initial point update once.

5.2.8 Pre-Optimization Results

We will do two different types of pre-optimization.

Method I : Automatic Initial Point 1 + Initial Point Update + gm Optimize

Method II : Automatic Initial Point 2 + Initial Point Update + gm-vdsat Optimize

Method III : Automatic Initial Point 3 + Initial Point Update 3

We get the following values after performing Method I:

Table 5.2: CG LNA Pre Optimization Circuit Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Rb (Ω)	1356	849	32
Rd (Ω)	316	316	316
W (μm)	1120	1210	1640
Io (μA)	674	732	1440
C1 (pF)	318	318	318
C2 (pF)	299	162	112
Rbias (Ω)	1000	1000	1000

Table 5.3: CG LNA Pre Optimization Output Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Gain (dB)	9.45	12.8	14.1
s11 (dB)	-14.4	-15.4	-5.35
NF (dB)	3.55	3.22	6.74
IIP3 (dBm)	-5.97	-4.56	5.23
Io (μA)	674	732	1690

We get the following values after performing Method II:

Table 5.4: CG LNA Pre Optimization Circuit Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Rb (Ω)	784	424	85
Rd (Ω)	316	316	316
W (μm)	261	97.5	69.4
Io (μA)	1120	1100	1250
C1 (pF)	318	318	318
C2 (pF)	99.7	33	7.55
Rbias (Ω)	1000	1000	2100

Table 5.5: CG LNA Pre Optimization Output Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Gain (dB)	16.3	15.74	13.4
s11 (dB)	-18.2	-27.9	-16.8
NF (dB)	2.98	3.24	5.23
IIP3 (dBm)	3.4	10.6	9.3
Io (μA)	1100	1090	1310

Except for the case of NF of TSMC 65nm, the other parameters are within the acceptable output condition. Method II is clearly better than Method I. We will use the results from Method II to set as the initial point for optimization. Now, we will perform optimization to minimize the current.

We get the following values after performing Method III:

Table 5.6: CG LNA Pre Optimization Circuit Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Rb (Ω)	50	50	50
Rd (Ω)	1580	1580	1580
W (μm)	15.6	5.54	3.27
Io (μA)	100	100	100
C1 (pF)	318	318	318
C2 (pF)	40.7	11.7	3.07
Rbias (Ω)	1000	1350	5150

Table 5.7: CG LNA Pre Optimization Output Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Gain (dB)	12.6	11.3	9.26
s11 (dB)	-23.7	-25.1	-27.0
NF (dB)	12.9	13.3	14.5
IIP3 (dBm)	3.8	0.8	7.9
Io (μA)	144	141	158

This result will not give good NF. But since we are not doing loss optimization, this does not matter.

5.3 Optimization with Ideal Circuit

We peform optimization after doing pre-optimization. Here, we will compare the results with different MOSFET technologies (TSMC 180nm, IBM 130nm, and TSMC 65nm). We will perform these optimizations at 27^{o}C and TT corner.

Table 5.8: CG LNA Optimization Circuit Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Rb (Ω)	215	194	265
Rd (Ω)	1330	946	382
W (μm)	116	67.7	328
Io (μA)	536	553	657
C1 (pF)	318	318	318
C2 (pF)	99.7	33	7.55
Rbias (Ω)	1000	1000	2100

Table 5.9: CG LNA Optimization Output Parameters

Parameter Name	TSMC 180nm	IBM 130nm	TSMC 65nm
Gain (dB)	21.9	18.7	12.3
s11 (dB)	-16.3	-15.6	-15.7
NF (dB)	3.99	3.99	3.97
IIP3 (dBm)	-4.99	2.1	-0.6
Io (μA)	563	575	731

5.4 Optimization with Real Resistors

We perform optimization, by using the circuit with non-ideal resistors, but ideal capacitors. We will perform these optimizations at 27^{o}C and TT corner.

Table 5.10: CG LNA Optimization Circuit Parameters

Parameter Name	Ideal Resistor	Non-Ideal Resistor
Rb (Ω)	265	272
Rd (Ω)	382	360
W (μm)	328	275
Io (μA)	657	666
C1 (pF)	318	159
C2 (pF)	7.55	163
Rbias (Ω)	2100	2100

Table 5.11: CG LNA Optimization Output Parameters

Parameter Name	Ideal Resistor	Non-Ideal Resistor
Gain (dB)	12.3	12.5
s11 (dB)	-15.7	-15.0
NF (dB)	3.97	3.99
IIP3 (dBm)	-1.07	-0.87
Io (μA)	745	727

5.5 Optimization with Real Capacitors

We perform optimization, by using the circuit with non-ideal resistors, and non-ideal capacitors. We will perform these optimizations at 27^oC and TT corner.

In case I, we will use mimcaps for C2.

In case II, we will use MOS for C2.

Table 5.12: CG LNA Optimization Circuit Parameters

Parameter Name	Ideal	Case I	Case II
Rb (Ω)	265	343	331
Rd (Ω)	382	320	314
W (μm)	328	183	190
Io (μA)	657	911	1270
C1 (pF)	318	21.7	31.8
C2 (pF)	7.55	110	124
Rbias (Ω)	2100	1000	1000

Table 5.13: CG LNA Optimization Output Parameters

Parameter Name	Ideal	Case I	Case II
Gain (dB)	12.3	12.0	12.0
s11 (dB)	-15.7	-15.0	-15.0
NF (dB)	3.97	3.94	3.99
IIP3 (dBm)	-1.07	-1.12	0.77
Io (μA)	745	937	929

5.6 FOM Optimization with Ideal Components

We will maximize the FOM by having an s11 smaller than -15dB. We only use TSMC 180nm and IBM 130nm technologies for FOM optimization.

Table 5.14: TSMC 180nm : FoM over multiple frequencies

f (GHz)	$I_D(\mu A)$	Gain(dB)	IIP_3(dBm)	NF(dB)	P(μW)	FoM(dB)
1	151	15.1	9.01	11.4	455	16.4
2	104	17.4	4.61	13.2	321	16.9
3	109	14.8	4.78	12.8	335	16.5
4	120	14.2	4.35	12.6	375	16.5
5	1690	16.8	2.00	4.21	6110	15.8
6	1120	17.1	0.93	5.51	4030	15.6
7	1430	16.1	1.52	4.69	5130	16.1
8	1470	15.6	2.34	5.11	5310	16.2
9	1290	15.2	1.83	5.32	4630	16.1
10	1260	14.8	2.40	5.90	4560	16.0

Table 5.15: IBM 130nm : FoM over multiple frequencies

f (GHz)	$I_D(\mu A)$	Gain(dB)	IIP_3(dBm)	NF(dB)	P(μW)	FoM(dB)
1	1300	19.1	-2.06	2.92	3390	11.9
2	1140	19.2	-2.06	3.50	2980	14.5
3	1200	18.8	-1.59	3.59	3130	15.9
4	1300	18.0	-0.86	3.60	3400	16.7
5	956	18.2	-2.10	4.10	2480	17.2
6	1040	18.0	-1.78	3.98	2730	17.8
7	865	18.0	-2.15	4.65	2260	17.9
8	847	17.6	-2.12	4.62	2190	18.3
9	988	17.2	-1.80	4.21	2570	18.7
10	1200	16.7	-1.20	3.78	3150	19.1

If we observe the result, we see that they are of two types:

- High power, low NF
- Very low power, very high NF

So, we will use Pre-Optimization Method III and see if the FOM is better optimized.

Table 5.16: TSMC 180nm : FoM over multiple frequencies

f (GHz)	$I_D(\mu A)$	Gain(dB)	IIP_3(dBm)	NF(dB)	P(μW)	FoM(dB)
1	128	11.7	14.0	12.1	381	18.0
2	122	17.5	4.36	12.6	379	16.7
3	121	16.6	3.39	12.9	381	16.3
4	113	13.4	3.69	11.5	350	16.5
5	86.9	13.3	5.05	14.0	265	17.2
6	1210	17.1	0.78	5.12	4340	15.8
7	1140	16.6	1.16	5.68	4100	15.7
8	1340	15.7	1.68	5.06	4840	16.2
9	1270	15.3	1.98	5.55	4580	16.1
10	1580	14.6	2.94	5.26	5730	16.2

Table 5.17: IBM 130nm : FoM over multiple frequencies

f (GHz)	$I_D(\mu A)$	Gain(dB)	IIP_3(dBm)	NF(dB)	P(μW)	FoM(dB)
1	148	14.7	20.2	11.7	328	28.2
2	210	12.7	7.61	10.2	444	17.0
3	3.46	12.5	1.79	21.1	6.47	19.8
4	145	10.4	5.71	10.9	308	16.6
5	166	11.2	7.02	11.5	354	18.4
6	990	18.1	-1.67	4.40	2580	17.7
7	124	8.82	9.49	12.1	266	20.6
8	170	13.8	5.67	11.2	372	21.9
9	102	17.3	6.90	12.3	252	27.7
10	734	17.4	-2.12	5.42	1900	18.5

We can observe that IBM 130nm technology has better values of FoM at most frequencies. This shows that it is a superior technology compared to TSMC 180nm.

CHAPTER 6

COMMON SOURCE LNA

Later on in the project, we worked on optimizing the common source LNA circuit. Similar to common gate LNA, we designed pre-optimization method for this circuit and then ran optimization.

6.1 Circuit

6.1.1 Circuit Diagram

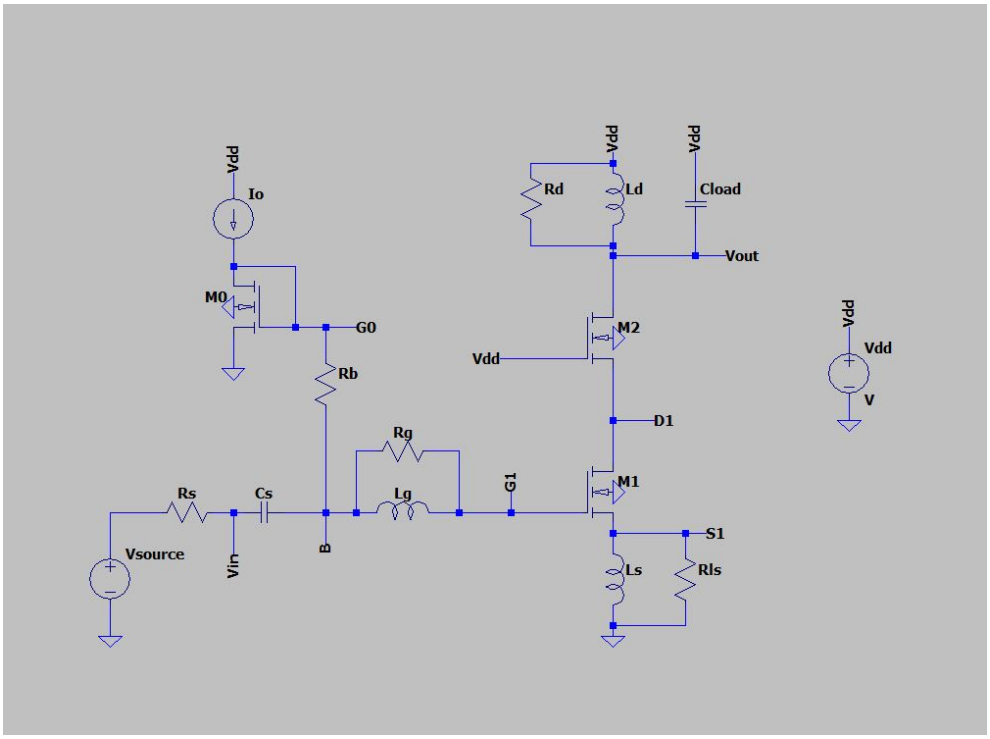

Figure 6.1: CS LNA Circuit Diagram

6.1.2 Circuit Equations

- Gain = 2*gm_1*Rd*Q_{in}
- Q_{in}=$\frac{1}{2*Rs*\omega*Cgs_1}$=$\frac{\omega*(Ls+Lg)}{2*Rs}$
- Zin=$\frac{1}{j\omega*Cgs_1}$+jω(Ls+Lg)+$\frac{gm_1*Ls}{Cgs_1}$
- F=1+gm_1*Rs*$\gamma * \frac{\omega_o}{\omega_T}^2$+4*$\frac{Rs}{Rd}$*$\gamma\frac{\omega_o}{\omega_T}^2$
- ω_T=$\frac{gm_1}{Cgs_1}$

6.1.3 Output Conditions

- fo=1 GHz
- s11 $<$ -10 dB
- IIP3 $>$ -10 dBm
- Gain $>$ 20 dB
- NF $<$ 1.5 dB $\implies$ F $<$ 1.41
- Cload=2000 fF

6.2 Pre-Optimization

We have 2 stages of pre-optimization for CS LNA:

- Automatic Initial Point
- Initial Point Update

We will keep making changes to pre-optimization to make it give a better initial point. We will discuss them here.

6.2.1 Pre Optimization - Method I

Automatic Initial Point

We will follow these steps to choose the values:

- Calculating Ld
  - Ld=$\frac{1}{\omega^2 * C_{Load}}$
- Calculating W
  - gm=$\frac{Gain}{2*Rd*Q_{in}}$, where Q_{in}=2.5
  - Cgs_1=$\frac{gm_1}{\omega_T}$, where ω_T=$\omega_o\sqrt{\frac{2*gm_1*Rs+4*Rs/Rd}{F-1}}$
  - W=$\frac{3*Cgs_1}{2*Lmin*C_{ox}}$
- Calculating Io
  - Io=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Ls, Lg, Rb, and Cs
  - Ls=$\frac{Rs*Cgs_1}{gm_1}$
  - Lg=$\frac{1}{\omega^2*Cgs_1}$-Ls
  - Rb=100*Rs = 5000
  - Cs=$\frac{100}{\omega*Rs}$

Pre-Optimization Results

We get the following values after performing pre-optimization. We will show the result of the value before initial point update.

Table 6.1: CS LNA Method I Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	12.6
Ls (nH)	8.81
Lg (nH)	76.7
Io (μA)	1.23
W (μm)	316
Cs (pF)	318
Rb (Ω)	5000

Table 6.2: CS LNA Method I Pre Optimization Output Parameters

Parameter Name	Value
Gain (dB)	-12.6
s11 (dB)	-2.18
NF (dB)	25.4
IIP3 (dBm)	0.0
Io (μA)	3.13
gm (μS)	86.1

We see that gm is very small and this is affecting the NF of the circuit. So, we need to get a larger value of gm.

6.2.2 Pre Optimization - Method II

Automatic Initial Point

We will follow these steps to choose the values:

- Calculating Ld
  - Ld=$\frac{1}{\omega^2 * C_{Load}}$
- Calculating W
  - gm=20mS
  - Q_{in}=2.5
  - Cgs_1=$\frac{1}{2*Rs*Q_{in}*\omega}$
  - W=$\frac{3*Cgs_1}{2*Lmin*C_{ox}}$
- Calculating Io
  - Io=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Ls, Lg, Rb, and Cs
  - Ls=$\frac{Rs*Cgs_1}{gm_1}$
  - Lg=$\frac{1}{\omega^2*Cgs_1}$-Ls
  - Rb=100*Rs = 5000
  - Cs=$\frac{100}{\omega*Rs}$

Pre-Optimization Results

We get the following values after performing pre-optimization. We will show the result of the value before initial point update. We will use IBM 130nm technology for evaluating this pre-optimization result.

Table 6.3: CS LNA Method II Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	12.6
Ls (nH)	1.58
Lg (nH)	38.1
Io (μA)	81.4
W (μm)	680
Cs (pF)	318
Rb (Ω)	5000

Table 6.4: CS LNA Method II Pre Optimization Output Parameters

Parameter Name	Value
Gain (dB)	13.3
s11 (dB)	-6.86
NF (dB)	3.49
IIP3 (dBm)	21.8
Io (μA)	153
gm (mS)	3.97

Now, NF has improved and Gain is also good. But s11 is still worse off. So, we will make the changes to Lg and Ls in Initial Point Update.

Initial Point Update

We will update Ls and Lg to get better s11

- Calculating Lg and Ls
  - Ls'=Ls*$\frac{50}{Re(Zin)}$
  - Lg'=Lg-$\frac{Im(Zin)}{\omega}$

Initial Point Update Results

Table 6.5: CS LNA Method II Initial Point Update Circuit Parameters

Parameter Name	Value
Ld (nH)	12.6
Ls (nH)	1.35
Lg (nH)	29.7
Io (μA)	81.4
W (μm)	680
Cs (pF)	318
Rb (Ω)	5000

Table 6.6: CS LNA Method II Initial Point Update Output Parameters

Parameter Name	Value
Gain (dB)	16.6
s11 (dB)	-43.0
NF (dB)	3.21
IIP3 (dBm)	19.6
Io (μA)	153
gm (mS)	3.97

Now, we can see that the value of s11 is very good after performing initial point update.

6.2.3 Working with TSMC65 Technology

Now, we will use TSMC 65nm Technology with the circuit. We will stop using IBM 130nm technology.

6.2.4 MOSFET Changes

We will add fingers for MOSFETs. We will choose the fingers such that the width of each MOSFET does not exceed 2μm.

6.2.5 Running Circuit at multiple frequencies

Another change that we make is running the circuit at multiple frequencies. We will run at fo, fo - Δf and fo + Δf. We will show the results for Δf=50 MHz.
When we do this, we realize that the s11 is very good at fo but poor at fo - Δf and fo + Δf. So, we will make changes to the calculation of Qin.

6.2.6 Changes to the output conditions

We make a change to the output condition. Cload is now 400fF. We make this change because we will connect the LNA Output to Mixer Input and we expect the input capacitance of the mixer to be around 400fF.

6.2.7 Pre Optimization - Method III

Automatic Initial Point

We will follow these steps to choose the values:

- Calculating Ld
  - Ld=$\frac{1}{\omega^2 * C_{Load}}$
- Calculating W
  - Kw=max[(1+$\frac{\Delta\omega}{\omega_o}$)-($\frac{1}{1+\frac{\Delta\omega}{\omega_o}}$),(1-$\frac{\Delta\omega}{\omega_o}$)-($\frac{1}{1-\frac{\Delta\omega}{\omega_o}}$)]
  - Ks=$\frac{1}{|s_{11}|^2}$-1
  - Q_{in}=$\frac{1}{K_w}\sqrt{\frac{1}{K_s}}$
  - gm=100mS
  - Cgs_1=$\frac{1}{2*Rs*Q_{in}*\omega}$
  - W=$\frac{3*Cgs_1}{2*Lmin*C_{ox}}$
- Calculating Io
  - Io=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Ls, Lg, Rb, and Cs
  - Ls=$\frac{Rs*Cgs_1}{gm_1}$
  - Lg=$\frac{1}{\omega^2*Cgs_1}$-Ls
  - Rb=100*Rs = 5000
  - Cs=$\frac{100}{\omega*Rs}$

Initial Point Update

- Calculating W
  - Z_max=100*$\sqrt{\frac{1}{\frac{1}{|s11|^2}-1}}$
  - Z_diff=| Im($Zin_{fo-\Delta f}$)-Im($Zin_{fo+\Delta f}$) |
  - W'=W*$\frac{Z\_diff}{Z\_max}$*1.2
- Calculating Io
  - Io'=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Lg and Ls
  - Ls'=Ls*$\frac{50}{Re(Zin)}$
  - Lg'=Lg-$\frac{Im(Zin)}{\omega}$

Pre-Optimization Results

Table 6.7: CS LNA Method III Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	63.3
Ls (nH)	0.97
Lg (nH)	22.8
Io (μA)	526
W (μm)	1550
Cs (pF)	318
Rb (Ω)	5000

Table 6.8: CS LNA Method III Pre Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	25.0	24.4	23.5
s11 (dB)	-15.0	-19.0	-17.2
NF (dB)	0.59	0.61	0.65
IIP3 (dB)	-10.9	-11.1	-11.2
Io (μA)		1780	

6.2.8 Circuit Changes

Addition of Cd

We will add Cd in parallel with Ld. This will allow use to better choose the value of Ld. Earlier, Ld was constrained to be decided by the value of Cload and Cgd of M2. Now, we can set Cd and Ld together.

Addition of Cg, R1, and R2

We will connect supply voltage with resistive divider to the gate of M2 (the cascode). This will give us the freedom to set the gate voltage of M2 instead of being constrained by choosing the value Vdd.

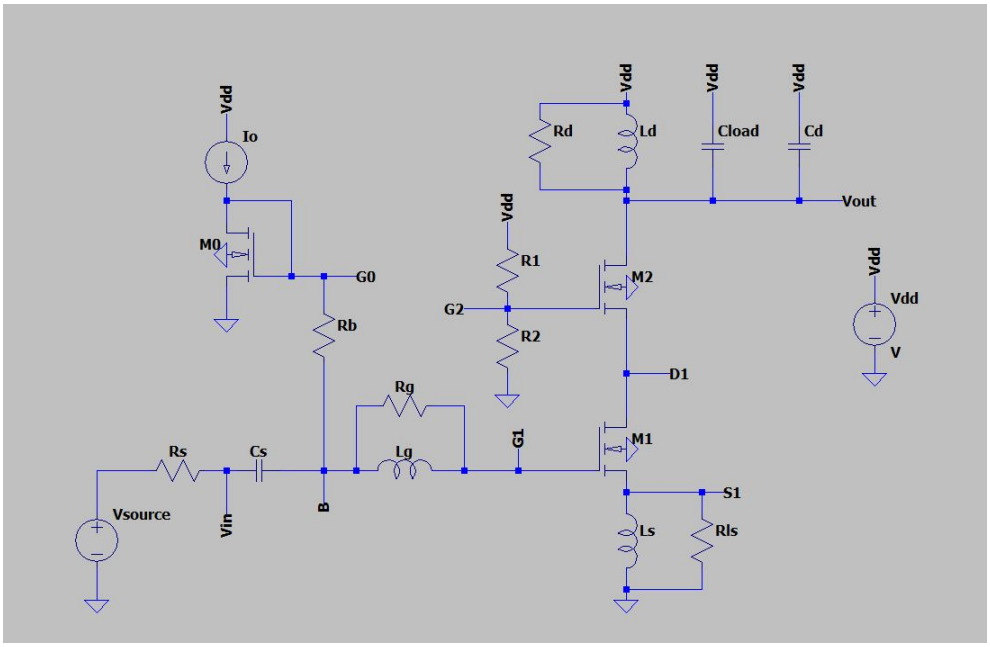

Figure 6.2: CS LNA Circuit Diagram

6.2.9 Pre Optimization - Method IV

Automatic Initial Point

We will follow these steps to choose the values:

- Calculating Ld and Cd
  - Cd=Cload
  - Ld=$\frac{1}{\omega^2*(Cd+C_{Load})}$
- Calculating W
  - Kw=max[(1+$\frac{\Delta\omega}{\omega_o}$)-($\frac{1}{1+\frac{\Delta\omega}{\omega_o}}$),(1-$\frac{\Delta\omega}{\omega_o}$)-($\frac{1}{1-\frac{\Delta\omega}{\omega_o}}$)]
  - Ks=$\frac{1}{|s_{11}|^2}$-1
  - Q_{in}=$\frac{1}{K_w}\sqrt{\frac{1}{K_s}}$
  - gm=100mS
  - Cgs_1=$\frac{1}{2*Rs*Q_{in}*\omega}$
  - W=$\frac{3*Cgs_1}{2*Lmin*C_{ox}}$
- Calculating Io
  - Io=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Cg, R1, and R2
  - Cg=10*Cgs_1
  - Vg=0.9
  - Reff=max($\frac{10}{wo.Cg}$,1000)
  - R1=$\frac{Reff.Vdd}{Vg}$
  - R2=$\frac{R1.Vg}{Vdd-Vg}$
- Calculating Ls, Lg, Rb, and Cs
  - Ls=$\frac{Rs*Cgs_1}{gm_1}$
  - Lg=$\frac{1}{\omega^2*Cgs_1}$-Ls
  - Rb=100*Rs = 5000
  - Cs=$\frac{100}{\omega*Rs}$

Initial Point Update

- Calculating W
  - Z_max=100*$\sqrt{\frac{1}{\frac{1}{|s11|^2}-1}}$
  - Z_diff=| Im(Zin$_{fo-\Delta f}$)-Im(Zin$_{fo+\Delta f}$) |
  - W'=W*$\frac{Z\_diff}{Z\_max}$*1.2
- Calculating Io
  - Io'=$\frac{gm_1^2}{2*\mu_n*C_{ox}*\frac{W}{Lmin}}$
- Calculating Lg and Ls
  - Ls'=Ls*$\frac{50}{Re(Zin)}$
  - Lg'=Lg-$\frac{Im(Zin)}{\omega}$

Table 6.9: CS LNA Method IV Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	31.5
Ls (nH)	1.11
Lg (nH)	22.6
Io (μA)	514
W (μm)	1590
Cs (pF)	318
Cd (pf)	400
Cg (pF)	10.9
Rb (Ω)	5000
R1 (Ω)	1000
R2 (Ω)	9000

Table 6.10: CS LNA Method IV Pre Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	24.0	23.1	22.0
s11 (dB)	-15.6	-22.3	-19.0
NF (dB)	0.68	0.70	0.73
IIP3 (dB)	-12.4	-12.6	-12.7
Io (μA)		1470	

6.2.10 Pre Optimization - Method V

Earlier, we arbitrarily chose gm=100mS. Now, we will choose gm based on the output NF specification. We will choose a larger value of gm than the one we get from calculations. This is because, when we run simulations, we find that choosing the value of gm from Equation 6.3, it will not satisfy the NF spec. So, we will choose a value of gm that it 4 times the value calculated from Equation 6.3.

Automatic Initial Point

$$F = 1 + gm.Rs.\gamma.(\frac{wo}{wT})^2 + 4\frac{Rs}{Rd}.(\frac{wo}{wT})^2 \tag{6.1}$$

$$F = 1 + Rs.\gamma.\frac{(wo.Cgs)^2}{gm} + 4\frac{Rs}{Rd}.\frac{(wo.Cgs)^2}{gm^2} \tag{6.2}$$

$$\frac{1}{gm} = \frac{-B + \sqrt{(B^2 - 4AC)}}{2A} \tag{6.3}$$

$$A = 4.\frac{Rs}{Rd}.(wo.Cgs)^2 \tag{6.4}$$

$$B = Rs.\gamma.(wo.Cgs)^2 \tag{6.5}$$

$$C = 1 - F \tag{6.6}$$

Table 6.11: CS LNA Method V Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	19.8
Ls (pH)	54.9
Lg (nH)	17.3
Io (μA)	359
W (μm)	1760
Cs (pF)	318
Cd (pf)	400
Cg (pF)	12.1
Rb (Ω)	5000
R1 (Ω)	1000
R2 (Ω)	9000

Table 6.12: CS LNA Method V Pre Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	27.7	26.4	25.8
s11 (dB)	-13.3	-12.5	-12.8
NF (dB)	0.92	0.90	0.90
IIP3 (dB)	-10.7	-13.6	-14.8
Io (μA)	1130		

6.3 Optimization with Ideal Circuit

Till now, we discussed pre-optimization. Now, we will look into the results from optimization. We will perform these optimizations at 27^oC and TT corner.

Table 6.13: CS LNA Ideal Circuit Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	22.2
Ls (nH)	0.90
Lg (nH)	34.1
Io (μA)	240
W (μm)	1110
Cd (fF)	392
Cg (pF)	12.1
Cs (pF)	318
Rb (Ω)	5000
R1 (Ω)	3500
R2 (Ω)	6500

Table 6.14: CS LNA Ideal Circuit Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	27.4	25.1	23.2
s11 (dB)	-11.9	-15.2	-16.4
NF (dB)	1.49	1.45	1.48
IIP3 (dBm)	23.4	22.7	22.8
Io (μA)		460	

6.4 Optimization with Real Resistors

Now, we will perform optimization on the circuit with real resistors. We will use rn-polywo resistor from the TSMC 65nm package.

Table 6.15: CS LNA Real Resistor Circuit Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	17.5
Ls (nH)	1.05
Lg (nH)	29.7
Io (μA)	418
W (μm)	1270
Cd (fF)	342
Cg (pF)	12.1
Cs (pF)	318
Rb (Ω)	5000
R1 (Ω)	5300
R2 (Ω)	4700

Table 6.16: CG LNA Real Resistor Circuit Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	30.0	28.4	26.3
s11 (dB)	-13.9	-11.2	-11.3
NF (dB)	1.45	1.49	1.45
IIP3 (dBm)	22.1	23.4	24.6
Io (μA)		514	

6.5 Optimization with Real Capacitors

Now, we will use real capacitors. For Cd and Cs, we will use mimcaps. For Cg, we will use MOS as a capacitor.

6.5.1 Pre-Optimization Change

When using real capacitors, we make a small change to pre-optimization. Cs=$\frac{10}{wo.Rs}$
We make this change because for an ideal capacitor, there is no parasitic capacitance to ground which will affect s11. But for real capacitor, there is a parasitic capacitance to ground which increases as the capacitance value increases. The larger the capacitance, the more that s11 gets affected. So, we will need to make a compromise. Optimization will fine tune the value of Cs.

6.5.2 Results

Table 6.17: CS LNA Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	14.2
Ls (nH)	1.92
Lg (nH)	15.7
Io (μA)	505
W (μm)	2280
Cd (fF)	751
Cg (pF)	12.1
Cs (pF)	28.7
Rb (Ω)	5000
R1 (Ω)	1700
R2 (Ω)	8300

Table 6.18: CG LNA Pre Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	23.1	25.3	23.1
s11 (dB)	-12.3	-20.4	-22.4
NF (dB)	1.42	1.44	1.49
IIP3 (dBm)	23.4	24.5	25.0
Io (μA)		932	

6.6 Using Real Inductors

6.6.1 Issue with using Real Inductors

When we use inductors from the TSMC 65nm technology, there is a maximum limit of around 10nH. So, we need to make changes to pre-optimization to accomodate this change.

6.6.2 Pre Optimization - Method VI

Automatic Initial Point

- Ld=9nH
- Cd=$\frac{1}{\omega^2 * Ld}$-Cload

Initial Point Update Issue

When using real inductors, we are getting an issue with the value of Ls. This value is increasing more than 10nH and the value of s11 of the circuit is getting worse. So, we will skip this step and perform optimization with large value of alpha and small iterations initially. Later, we will optimize the circuit with a smaller value of alpha. Here, n_runs parameter of optimization is 2.

6.6.3 Circuit Changes

To reduce the value of current consumed, we will scale the current mirror. We will scale the width and current by a factor 'k'. This 'k' will be set to 10.

6.6.4 Pre Optimization Results

Table 6.19: CS LNA Method VI Pre Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	0.77
Lg (nH)	15.6
Io (μA)	36.6
W (μm)	2230
Cs (pF)	31.8
Cd (pf)	2410
Cg (pF)	12.1
Rb (Ω)	5000
R1 (Ω)	1000
R2 (Ω)	9000

* - 10x scaling is used for Io above as it is the current through current mirror.

Table 6.20: CS LNA Method VI Pre Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	12.8	12.5	12.3
s11 (dB)	-4.8	-5.9	-7.3
NF (dB)	2.08	2.03	2.01
IIP3 (dB)	-16.2	-16.5	-16.6
Io (μA)		818	

6.7 Optimization with Real Inductors

Now, we will do optimization. Optimization is performed at 27^{o}C and TT corner.

Table 6.21: CS LNA Inductor Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	0.98
Lg (nH)	14.7
Io (μA)	101
W (μm)	2900
Cd (fF)	973
Cg (pF)	12.1
Cs (pF)	318
Rb (Ω)	5000
R1 (Ω)	500
R2 (Ω)	9500

Table 6.22: CG LNA Inductor Optimization Output Parameters

Parameter Name	Value		
	fo-Δf	fo	fo+Δf
Gain (dB)	26.6	24.2	22.1
s11 (dB)	-17.4	-23.9	-22.7
NF (dB)	0.86	0.88	0.91
IIP3 (dBm)	-6.7	-6.1	-7.9
Io (μA)		2200	

6.8 Optimization with different process

We will perform these optimizations at 27^{o}C and SS,TT,FF corner.

Table 6.23: CS LNA Process Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	3.18
Lg (nH)	16.2
Io (μA)	69.2
W (μm)	2670
Cd (fF)	423
Cg (pF)	12.1
Cs (pF)	14.2
Rb (Ω)	5000
R1 (Ω)	500
R2 (Ω)	9500

Table 6.24: CG LNA Process Optimization Output Parameters

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	22.0	22.6	21.7	21.0	21.6	20.9	20.3	20.7	20.1
s11 (dB)	-14.0	-20.2	-15.2	-16.9	-38.0	-16.8	-17.4	-20.6	-14.5
IIP3 (dBm)	-8.0	-6.2	-3.7	-9.6	-7.9	-4.9	-10.2	-8.8	-6.0
NF (dB)	1.18	1.23	1.29	1.28	1.34	1.41	1.32	1.38	1.47
Id (μA)		1900			1620			1540	

6.9 Optimization with different process and temperature

We will perform these optimizations at -40^{o}C, 27^{o}C, and 120^{o}C and SS,TT,FF process corner.

Table 6.25: CS LNA Temperature Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	3.18
Lg (nH)	16.5
Io (μA)	1150
W (μm)	2540
Cd (fF)	468
Cg (pF)	12.1
Cs (pF)	12.6
Rb (Ω)	5000
R1 (Ω)	500
R2 (Ω)	9500

Table 6.26: CG LNA Temperature Optimization Output Parameters T=-40^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	23.9	25.3	24.2	23.0	24.4	23.6	22.1	23.3	22.8
s11 (dB)	-10.0	-16.4	-13.8	-12.3	-25.2	-16.9	-13.7	-21.4	-15.3
IIP3 (dBm)	-6.1	-6.2	-6.0	-8.6	-7.3	-5.4	-10.4	-9.3	-6.7
NF (dB)	0.67	0.68	0.72	0.72	0.74	0.78	0.77	0.80	0.84
Id (μA)		2510			1950			1670	

Table 6.27: CG LNA Temperature Optimization Output Parameters T=27^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	23.2	24.1	23.0	22.6	23.5	22.6	22.1	22.9	22.2
s11 (dB)	-11.0	-16.3	-13.2	-13.4	-28.2	-16.1	-14.7	-24.0	-15.1
IIP3 (dBm)	-5.6	-5.7	-5.2	-7.0	-5.8	-4.6	-7.9	-6.7	-4.8
NF (dB)	0.94	0.97	1.00	0.98	1.01	1.06	1.00	1.04	1.10
Id (μA)		2780			2350			2210	

Table 6.28: CG LNA Temperature Optimization Output Parameters T=120^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	22.2	22.6	21.6	21.8	22.3	21.3	21.8	22.3	21.3
s11 (dB)	-12.3	-16.2	-13.1	-15.0	-26.5	-15.6	-16.0	-26.6	-14.9
IIP3 (dBm)	-5.0	-5.1	-4.5	-5.5	-4.8	-4.2	-5.5	-4.8	-4.6
NF (dB)	1.40	1.45	1.52	1.42	1.48	1.55	1.36	1.42	1.50
Id (μA)		2970			2750			2890	

6.10 Circuit Changes

6.10.1 Motivation for Change

For our circuit, we will add variation of Io with temperature. The value of Io will vary linearly with temperature. Io=Io$_{T=27}$+ **Iok** *(T-27)

6.10.2 Results

Table 6.29: CS LNA Temperature Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	3.17
Lg (nH)	17.5
Io (μA)	1390
Io_k(μA/K)	2.0
W (μm)	2640
Cd (fF)	490
Cg (pF)	12.1
Cs (pF)	12.9
Rb (Ω)	5000
R1 (Ω)	500
R2 (Ω)	9500

Table 6.30: CG LNA Temperature Optimization Output Parameters T=-40^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	25.8	26.0	24.1	24.8	25.0	23.3	23.6	23.7	22.4
s11 (dB)	-12.1	-24.8	-14.5	-14.1	-23.3	-14.5	-14.3	-15.9	-12.0
IIP3 (dBm)	-5.98	-6.33	-5.55	-7.94	-6.23	-4.66	-9.73	-7.97	-4.94
NF (dB)	0.776	0.8	0.833	0.82	0.849	0.89	0.863	0.899	0.949
Id (μA)	2700.0			2100.0			1800.0		

Table 6.31: CG LNA Temperature Optimization Output Parameters T=27^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	24.4	24.9	23.4	23.8	24.3	22.9	23.3	23.7	22.4
s11 (dB)	-11.1	-18.7	-15.2	-12.8	-27.5	-17.8	-13.2	-18.6	-14.8
IIP3 (dBm)	-5.35	-5.75	-5.09	-6.32	-5.31	-4.52	-7.18	-5.79	-4.41
NF (dB)	1.08	1.11	1.15	1.11	1.15	1.2	1.12	1.16	1.22
Id (μA)		3020.0			2550.0			2390.0	

Table 6.32: CG LNA Temperature Optimization Output Parameters T=120^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	24.2	23.7	22.0	23.8	23.3	21.6	23.8	23.1	21.4
s11 (dB)	-14.6	-19.3	-14.1	-17.3	-25.6	-14.9	-16.2	-17.4	-12.6
IIP3 (dBm)	-5.28	-5.81	-4.65	-4.77	-5.14	-4.59	-4.64	-4.98	-4.95
NF (dB)	1.45	1.5	1.57	1.45	1.51	1.59	1.4	1.47	1.55
Id (μA)		3790.0			3480.0			3600.0	

6.11 Circuit Changes

6.11.1 Motivation for Change

For our circuit, we will add capacitors with switches parallel to Cd. This will change the effective value of capacitance that is parallel to Ld. The switches are controlled by circuitry that responds to the process variations of the MOSFETs. This switch is added to negate the effect of process variation.

6.11.2 Circuit Diagram

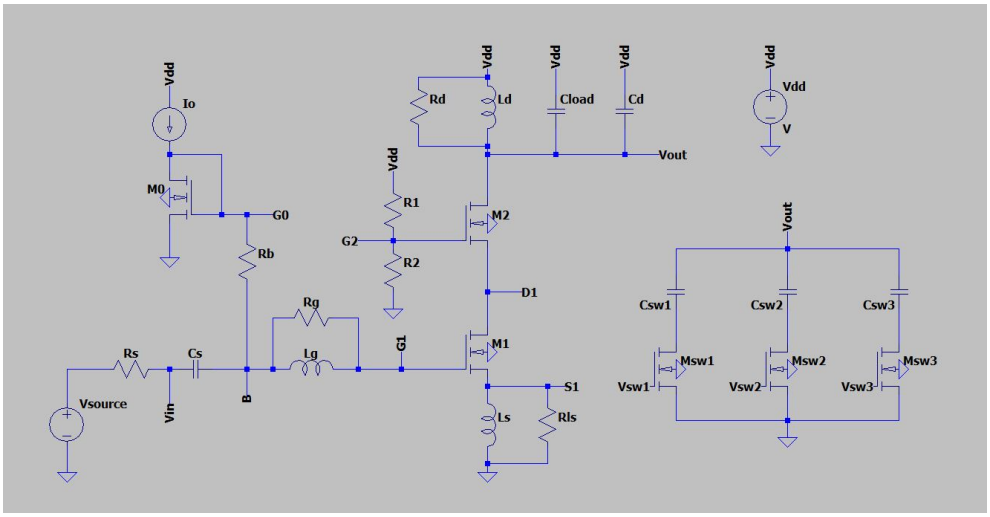

Figure 6.3: CS LNA Circuit Diagram

6.11.3 Results

Table 6.33: CS LNA Temperature Optimization Circuit Parameters

Parameter Name	Value
Ld (nH)	9.0
Ls (nH)	3.17
Lg (nH)	17.9
Io (μA)	1630
Io_k(μA/K)	2.21
W (μm)	2520
Cd (fF)	450
Cg (pF)	12.1
Cs (pF)	12.2
Rb (Ω)	5000
R1 (Ω)	500
R2 (Ω)	9500
Wpr (μm)	0.695
Cpr_W(μm)	4.06

Table 6.34: CG LNA Temperature Optimization Output Parameters T=-40^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	25.5	26.6	25.0	24.6	25.7	24.4	23.8	24.6	23.4
s11 (dB)	-10.0	-19.9	-15.7	-11.7	-21.9	-16.1	-12.8	-16.8	-12.8
IIP3 (dBm)	-5.61	-6.41	-5.8	-6.65	-6.25	-4.4	-9.04	-7.64	-5.26
NF (dB)	0.746	0.766	0.795	0.776	0.8	0.835	0.808	0.838	0.88
Id (μA)		3030.0			2350.0			2020.0	

Table 6.35: CG LNA Temperature Optimization Output Parameters T=27^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	25.0	25.6	24.0	24.4	25.1	23.6	24.0	24.5	23.0
s11 (dB)	-10.9	-19.7	-14.9	-12.7	-26.9	-16.1	-13.4	-17.9	-13.0
IIP3 (dBm)	-4.87	-5.83	-5.23	-6.14	-6.21	-5.54	-6.16	-5.29	-4.95
NF (dB)	0.99	1.01	1.06	1.01	1.04	1.09	1.02	1.05	1.11
Id (μA)		3610.0			3040.0			2840.0	

Table 6.36: CG LNA Temperature Optimization Output Parameters T=120^{o}C

Parameter Name	ss			tt			f		
	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf	fo-Δf	fo	fo+Δf
Gain (dB)	24.2	24.4	22.8	23.9	24.1	22.6	24.0	23.9	22.3
s11 (dB)	-12.2	-18.9	-14.4	-14.1	-39.6	-15.9	-14.4	-19.1	-13.3
IIP3 (dBm)	-4.68	-5.64	-5.01	-8.9	-9.63	-9.69	-4.48	-5.27	-5.63
NF (dB)	1.37	1.42	1.48	1.37	1.42	1.49	1.34	1.39	1.46
Id (μA)		4200.0			3840.0			3920.0	

CHAPTER 7

CONCLUSION

7.1 Work done till now

We have designed a code that optimizes a CG LNA circuit. We also try to maximize the FOM of this circuit with different MOS technologies and compare their performance.

We also optimize a CS LNA circuit. We optimize it at different temperatures, frequencies, and process corners. We also used real components for resistors, capacitors, and inductors to run the circuit.

7.2 Future work

We have worked on simple gradient descent algorithm to optimize the circuit. However, this will take a lot of time for optimization, especially if we need to optimize multiple circuit parameters. Therefore, looking into alternatives for simple gradient descent algorithm will help in improving the run time of the algorithm.

Furthermore, while we have worked on circuit design, an RF chip is not complete without the layout. So, the future work of this project involves the optimization of an analog circuit layout. After the circuit design is completed, the layout of the circuit will be done.