**B.E. PROJECT**

ON

**OPTIMIZATION OF ANALOG CIRCUITS USING CUCKOO SEARCH ALGORITHM**

Submitted by:

**AMITESH KUMAR SINGH 024/EC/14**

Under the Guidance of

**Dr. KUNWAR SINGH**

A Project in partial fulfilment of requirement for the award of

B.E. in

Electronics and Communication Engineering



NETAJI SUBHAS INSTITUTE OF TECHNOLOGY

(UNIVERSITY OF DELHI)

NEW DELHI-110078

2014-2018

**SELF DECLARATION**

We hereby declare that all the work presented in our project report entitled **“ Optimization of Analog Circuits using Cuckoo Search Algorithm”** in the partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in **Electronics and Communication Engineering,** from **Netaji Subhas Institute of Technology**, is an authentic record of our own work carried out under the guidance of **Mr. Kunwar Singh.**

Abhishek Sharma (010/EC/14)

Amitesh K Singh (024/EC/14)

Garv Kumar (053/EC/14)

Karan (070/EC/14)

**CERTIFICATE**

This is to certify that report entitled “**Optimization of Analog Circuits Using Cuckoo Search Algorithm**” being submitted by **Abhishek Sharma, Amitesh Kumar Singh**, **Garv Kumar**, **Karan** for the award of bachelor’s degree of engineering, is the record of the bonafide work carried out by them under our supervision and guidance. The results obtained in this have not been submitted either in part or in full to any other university or institute for the award of any degree or diploma.

**Dr. KUNWAR SINGH**

**ECE Division**

**NSIT, New Delhi, India**

**ACKNOWLEDGEMENT**

We wish to express our profound gratitude and indebtedness to **Dr. Kunwar Singh**, Department of Electronics and Communication Engineering, **Netaji Subhas Institute of Technology**, Delhi for introducing the present topic and for his inspiring guidance, constructive criticism and valuable suggestion throughout the project work. We would like to express our special gratitude towards our mentor, for his kind cooperation and encouragement which helped us in completion of this project.

Abhishek Sharma (010/EC/14)

Amitesh K Singh (024/EC/14)

Garv Kumar (053/EC/14)

Karan (070/EC/14)

Bachelor of Engineering

Dept. of Electronics and Communication

Engineering

Netaji Subhas Institute of Technology

University of Delhi.

**ABSTRACT**

Today Artificial Intelligence is one of the rising areas in engineering. It is applied to almost every field to make the intelligent machines , especially intelligent computer programs. Another important thing about IC Design nowadays is the sizing of the MOSFETs. The size of the MOSFETs are shrinking day by day but the performance of the system is kept intact.

In this project we focus on development of a technique for Analog CMOS Circuit Optimization. Our approach is based on using the widths and lengths and of transistors and bias current in the circuit as variables to achieve desired specifications. Using Cuckoo Search Algorithm, we find the maximum **Gain-Bandwidth Product** under certain phase margin Constraints using the Cuckoo Search Algorithm and compare the results obtained from them for various parameters of CSA.

**LIST OF CONTENTS**

Certificate I

Acknowledgement II

Abstract III

List of figures VII

List of Tables IX

1. **Introduction**  **1**

1.1 MOSFET 1

1.1.1 Structure of MOSFET 1

1.1.2 Regions of Operation of MOSFET 2

1.2 CMOS 4

1.2.1 Application of CMOS circuits 4

1.2.2 Power Dissipation in CMOS circuits 4

1.2.3 Analog Applications 4

1.3 Inverting Amplifier 5

1.4 Current Mirrors 7

1.4.1 Need for Current Mirrors 7

14.2 Basic Current Mirrors 7

1.5 Differential Pair 8

1.5.1 Single Ended Signal And Differential Pair 8

1.5.2 Basic Differential Pair 9

1.6 Current Source 10

1.7 Characteristics of OTAs 10

2. **Nature Inspired Algorithms**  **12**

2.1 Introduction 12

2.2 Cuckoo Search Algorithm 12

2.2.1 Inspiration from Cuckoo Birds 12

2.2.2 Levy Flights 12

2.2.3 Idealized Rules 13

2.3 Pseudo Code of Cuckoo Search Algorithm 14

2.3.1 Flowchart for Cuckoo Search Code

2.3.2 Pseudo Code

2.4 Levy Flights Implementation 14

2.5 Choice of Parameters 15

2.6 An Example 16

2.6.1 Functions

2.6.2 Results

**3. Conventional CMOS OTA Design 17**

3.1 Introduction 17

3.2 Design procedure of the OTA 17

3.3 Calculation 19

3.4 Observations and Results 22

3.4.1 Target Parameters

3.4.2 DC Analysis 25

3.4.3 AC Analysis 26

3.4.4 Transient Analysis 27

3.4.5 Transient Analysis for slew rate

3.4.6 Observed Parameters

3.4.7 Table for MOSFETs aspect ratios

**4. Inversion Coefficient Theory 29**

4.1 Inversion Coefficient 29

4.2 Data for Calculations 30

4.2.1 Transistor width Ranges 31

**5. Results and Conclusions 35**

5.1 Integration Flowchart 35

5.2 Flowchart of the Automated Optimization Methodology 36

5.3 Working 37

5.4 Results 38

REFERENCES 42

APPENDIX 43

A.1 LT Spice 43

A.2 MATLAB 44

A.3 Netlist for circuit

A.4 BASH 45

A.5 TSMC018 46

**LIST OF FIGURES**

|  |  |
| --- | --- |
| A NMOS Structure | Fig. 1.1 |
| An NMOS transistor with VGS >Vt | Fig. 1.2 |
| Drain Current versus drain-to-source voltage when VGS > VT | Fig. 1.3 |
| Common Source Stage | Fig. 1.4 |
| Input-Output Characteristics | Fig. 1.5 |
| Resistive Divider Circuit | Fig. 1.6 |
| Applications of Current Mirror | Fig. 1.7 |
| Basic Current Mirror | Fig. 1.8 |
| Wilson Current Mirror | Fig. 1.9 |
| Effect of noise on single-ended and differential circuits | Fig. 1.10 |
| Basic Differential Pair 1 | Fig. 1.11 |
| Basic Differential Pair 2 | Fig. 1.12 |
| Symbol of a current source | Fig. 1.13 |
| Flowchart of Cuckoo Search Optimization | Fig. 2.1 |
| Method of optimization | Fig. 2.2 |
| Schumer Steiglitz Function | Fig. 2.3 |
| Sphere Function | Fig. 2.4 |
| CMOS OTA Schematic | Fig. 3.1 |
| DC Analysis of the circuit | Fig. 3.2 |
| AC Analysis of the circuit | Fig. 3.3 |
| Transient Analysis of the circuit | Fig. 3.4 |
| Transient Analysis with Slew Rate | Fig. 3.5 |
| I.C. performance tradeoffs in various inversion regions | Fig. 4.1 |
| Gm vs IC | Fig. 4.2 |
| Gain variation with M3 and M4 | Fig. 4.3 |
| Gain variation with M7 | Fig. 4.4 |
| Integration Flowchart | Fig. 5.1 |
| Steps followed in optimization process | Fig. 5.2 |
| Set 1 of Gain & PM | Fig. 5.3 |

|  |  |
| --- | --- |
| Set 1 of Widths | Fig. 5.4 |
| Set 2 of Gain & PM | Fig. 5.5 |
| Set 2 of Widths | Fig. 5.6 |
| Set 3 of Gain & PM | Fig. 5.7 |
| Set 3 Widths | Fig. 5.8 |
| GBP Variation with number of iterations | Fig. 5.9 |
| Mosfet width variation with of number of iterations | Fig. 5.10 |
| Mosfet width value at end of each iteration | Fig. 5.11 |
| AC Analysis for 2500 iterations | Fig. 5.12 |

**LIST OF TABLES**

|  |  |
| --- | --- |
| Target Parameters | Table 3.1 |
| Observed Parameters | Table 3.2 |
| Table for Aspect Ratio of Mosfet | Table 3.3 |
| Table for Parameters and MosfetWidth | Table 5.1 |

**CHAPTER - 1**

**INTRODUCTION**

**1.1 MOSFET**

A Metal Oxide Semiconductor Field Effect Transistor is a transistor used for amplifying or switching electronic signals. The body of a MOSFET is usually connected to the source terminal which makes it a three-terminal device similar to other Field Effect

Transistors (FET). Field effect transistors form a large family of switchable devices. They are widely used in computer and communications technology. Current flowing within field effect transistors between the main electrodes (source and drain) is controlled by a voltage at the gate electrode. The gate voltage opens or closes a conducting channel between source and drain.

Compared to the bipolar junction transistor (BJT), the MOS transistor occupies a relatively smaller silicon area and its fabrication involves fewer processing steps. These technological advancements together with the relative simplicity of MOSFET operation, have helped make the MOS transistor the most widely used switching device in LSI and VLSI circuits.

**1.1.1 STRUCTURE OF MOSFET**

The basic structure of an n-channel MOSFET consists of a p-type substrate, in which two n+ diffusion regions, the drain and the source, are formed. The surface of the substrate region between the drain and the source is covered with a thin oxide layer, and the metal (or polysilicon) gate is deposited on top of this gate dielectric. The two n+ regions will be the current-conducting terminals of this device. Note that the device structure is completely symmetrical with respect to the drain and source regions; the different roles of these two regions will be defined only in conjunction with the applied terminal voltages and the direction of the current flow.

A conducting channel will eventually be formed through applied gate voltage in the section of the device between the drain and the source diffusion regions. The distance between the drain and source diffusion regions is the channel length L, and the lateral extent of the channel (perpendicular to the length dimension) is the channel width W. Both the channel length and the channel width are important parameters which can be used to control some of the electrical properties of the MOSFET. The thickness of the oxide layer covering the channel region, tx, is also an important parameter.

A MOS transistor which has no conducting channel region at zero gate bias is called an enhancement-type (or enhancement-mode) MOSFET. If a conducting channel already exists at zero gate bias, on the other hand, the device is called a depletion-type (or depletion-mode) MOSFET. In a MOSFET with p-type substrate and with n+ source and drain regions, the channel region to be formed on the surface is n-type. Thus, such device with p-type substrate is called an n-channel MOSFET. In a MOSFET with n-type substrate and with p+ source and drain regions, on the other hand, the channel is p-type and the device is called a p-channel MOSFET.

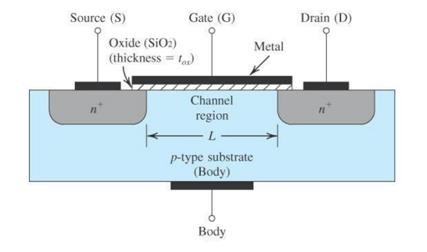


Fig 1.1 nMOSFET Structure

**1.1.2 REGIONS OF OPERATION OF MOSFET**

A MOSFET works in one of the three regions of operation depending upon the voltages across the three terminals, that is, gate, drain and source. The following figure depicts the structure of a MOSFET with currents and voltages at its terminals.

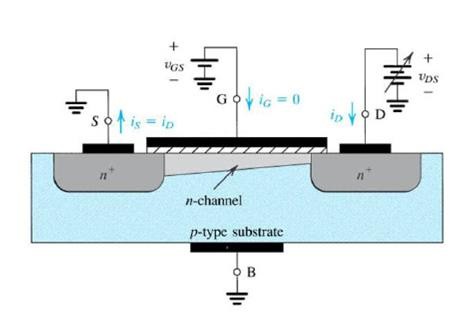


Fig 1.2 An NMOS transistor with VGS > VT

The three regions of operation are s follows:

**1. *Cut-off region:***

When the voltage applied at the gate terminal with respect to source terminal is below threshold voltage, the channel is not induced and there is no flow of electrons from source to the gate terminal. For MOSFET to be in cut-off region,

VGS < VTH (1.1)

where VTH is the threshold voltage and VGS is the voltage between the gate and source terminal.

**2. *Saturation region:***

As VGS is increased above the threshold voltage, a channel is formed for the flow of electrons. The voltage VDS is increased and it appears as a voltage drop across the length of the channel. That is, as we travel along the channel from source to drain, the voltage increases from 0 to VDS. Thus, the voltage between the gate and points along the channel decreases from VGS at the source end to VGS – VDS at the drain end. Since the channel depth depends on this voltage, we find that the channel is no longer of uniform depth; rather the channel will take the tapered form. As VDS is increased, the channel becomes more tapered and its resistance correspondingly.

Eventually, when VDS is increased to the value that reduces the voltage between gate and channel at the drain end to VT, that is, VGD = VT, the channel depth at the drain end decreases to almost zero. Increasing VDS beyond this value has little effect on the channel shape and the current through the channel remains constant at the value reached for VDS = VGS – VT. the drain current thus saturates at this value, and the mosfet enters the saturation region of operation.

For saturation region,

VGS >= VTH (1.2) VGD <= VT (1.3)

**3. *Triode region:***

For every value of VGS >= VTH, there is a corresponding Value of VDSsat. The device operates in the saturation region if VDS > VDSsat. The region obtained for VDS < VDSsat, is called the triode region.

This region is also called as the linear region since the drain current varies almost linearly with VDS.

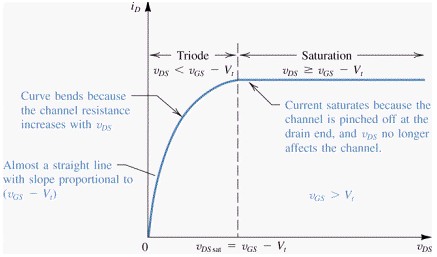


Fig 1.3 The drain current versus drain-to-source voltage when VGS > VT

**1.2 CMOS**

**1.2.1 Applications of CMOS Circuits**

CMOS is a technology for constructing integrated circuits. The important characteristics of CMOS devices are high noise immunity and low static power consumption. Since one transistor of the pair is always off, the series combination draws significant power only momentarily during switching between on and off states. Consequently, CMOS devices do not produce as much waste heat as other forms of logic, for example transistor–transistor logic (TTL) or NMOS logic, which normally have some standing current even when not changing state. CMOS also allows a high density of logic functions on a chip. Complementary MOSFET (CMOS) technology is widely used today to form circuits in numerous and varied applications. Today’s computers CPUs and cell phones make use of CMOS due to several key advantages. CMOS offers low power dissipation, relatively high speed, high noise margins in both states, and will operate over a wide range of source and input voltages.

**1.2.2 Power Dissipation In CMOS Circuits**

CMOS logic dissipates less power than NMOS logic circuits because CMOS dissipates power only when switching. NMOS logic dissipates power whenever the transistor is on, because there is a current path from Vdd to Vss through the load resistor and the n-type network. Static CMOS gates are very power efficient because they dissipate nearly zero power when idle.

**1.2.3 Analog Applications**

Besides digital applications, CMOS technology is also used in analog applications. For example, there are CMOS operational amplifiers ICs available in the market. Transmission gates may be used instead of signal relays. CMOS technology is also widely used for RF circuits all the way to microwave frequencies, in mixed-signal.

**1.3 INVERTING AMPLIFIER**

The common-source (CS) amplifier is used as an inverting amplifier and may be viewed as a transconductance amplifier or as a voltage amplifier. By virtue of its transconductance, a MOSFET converts variations in its gate-source voltage to a small-signal drain current, which can pass through a resistor to generate an output voltage.

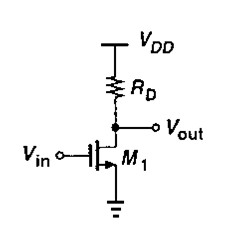


Fig. 1.4 Common Source Stage

If the input voltage increase from zero, M1 is off and Vout = VDD. As Vin approaches Vth, M1 begins to turn on, drawing current from Rd and lowering Vout. If VDD is not excessively low, M1 turns on in saturation and we have



(1.4)

Where channel length modulation is neglected. With further increase in Vin, Vout drops more and the transistor continues to operate in saturation until Vin exceeds Vout by Vth. At this point,

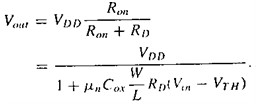
 (1.5)

From which (Vin1 – Vth) and hence Vout can be calculated.

For Vin > Vin1, M1 is in the triode region:

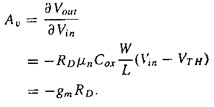
 (1.6)

If Vin is high enough to drive M1 into deep triode region, Vout << 2(Vin – Vth), and thus we get,



(1.7)

Since the transconductance drops in the triode region, we usually ensure that Vout > Vin – Vth. From the small signal model, we have,



The equation predicts certain effects if the circuit senses a large signal swing. Since gm itself varies with the input signal according to gm = µCox(W/L)(VGS- Vth), the gain of the circuit changes substantially if the input signal is large.

This result of the common source stage indicates that, this circuit acts as an amplifier if the input signal is kept large and inverts the input signal, therefore is used as an inverting amplifier.

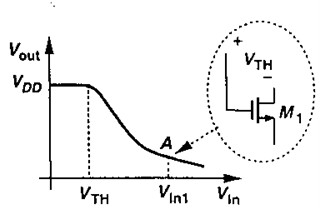


Fig.1.5 Input-Output characteristic

As a transconductance amplifier, the input voltage is seen as modulating the current going to the load. As a voltage amplifier, input voltage modulates the amount of current flowing through the mosfet, changing the voltage across the output resistance according to Ohm's law. However, the mosfet’s output resistance typically is not high enough for a reasonable transconductance amplifier (ideally infinite), nor low enough for a decent voltage amplifier (ideally zero).

**1.4 CURRENT MIRRORS**

A simple resistive divider circuit used to obtain current as output is shown below:

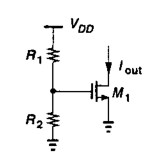


Fig.1.6 Resistive Divider circuit

Here if M1 is assumed to be in saturation then the current Iout is given by

Iout = 0.5unCox(W/L)((R2/ R1 + R2)Vdd – Vth)2 (1.8)

Iout is thus poorly defined since it is process, supply and temperature dependent.

**1.4.1 Why need current mirrors?**

Current mirrors are primarily used to replicate a current and to provide a stable output. They are extensively used for the following applications in analog circuits:

(1) As biasing elements to set the D.C operating point of analog CMOS circuits. (2) As active load resistances

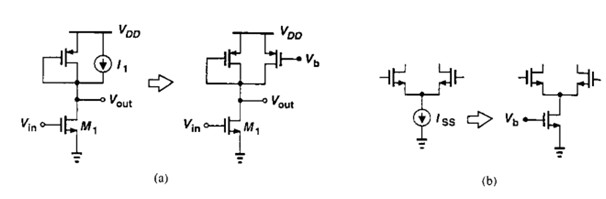


Fig. 1.7 Applications of current mirror

**1.4.2 Basic Current Mirror**

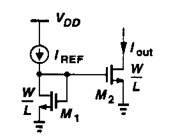
The simplest configuration of a current mirror comprises of a diode connected FET. The circuit is as shown.

Fig.1.8 Basic Current Mirror

The equations giving currents through the FETs are

Iref = 0.5unCox(W/L)1(Vgs– Vth)2 (1.9)

Iout = 0.5unCox(W/L)1(Vgs– Vth)2 (1.10)

Thus the output and reference current are related by

Iout/Iref= (W/L)2/(W/L)1 (1.11)

This topology can precisely copy currents and there is no process or temperature dependence. The output resistance, capacitance and voltage headroom of a current source trades with the magnitude of the output current. In the test circuit presented, the simple current mirror described above is used to set the D.C operating point. The simple current mirror has the disadvantage that the output resistance is not very high (as is characteristic of an ideal current source). Moreover there is dependence on the current gain, due to finite value of base current. Hence most analog circuits employ the Wilson current mirrors. The Wilson current mirror provides better performance in terms of higher output resistance.

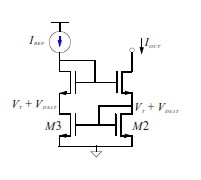


Fig. 1.9 Wilson Current Mirror

**1.5 DIFFERENTIAL PAIR**

**1.5.1 Single-ended and Differential Operation**

A single-ended signal is one that is measured with respect to a fixed reference potential. A differential signal is one that is measured between two nodes that have equal and opposite excursions about a fixed potential. An important advantage of differential configuration over single- ended signal are-

(1) The higher immunity against environmental noise, for instance, common mode rejection against noisy supply voltages.

This noise immunity can be illustrated by the following figure.

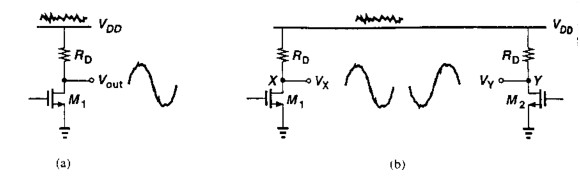


Fig. 1.10 Effect of noise in single-ended and differential circuits

(2) An increase in the maximum allowable voltage swings. (3) Simpler biasing

(4) Higher linearity

**1.5.2 Basic Differential Pair**

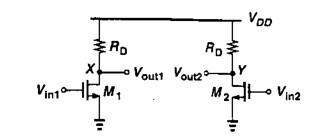


Fig.1.11 Basic Differential Pair 1

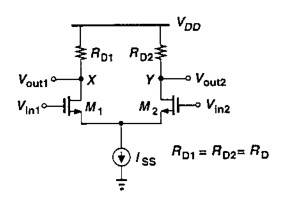


Fig.1.12 Basic Differential Pair 2

In the first circuit shown above, as the input CM level changes, the bias currents and hence the transconductance of the devices vary. This issue is resolved by using the second configuration as shown shown above, which uses a current source Iss to make the difference of the two drain currents independent of the common mode input.

In summary, a differential pair rejects the common mode signals and amplifies the difference of the signals applied to the two gates of the FETs. A differential pair forms the fundamental building block of an operational amplifier.

**1.6 CURRENT SOURCE**

A current source is an electronic circuit that delivers or absorbs an electric current which is independent of the voltage across it. In circuit theory, an ideal current source is a circuit element where the current through it is independent of the voltage across it.

The internal resistance of an ideal current source is infinite. An independent current source with zero current is identical to an ideal open circuit. The voltage across an ideal current source is completely determined by the circuit it is connected to. When connected to a short circuit, there is zero voltage and thus zero power delivered. When connected to a load resistance, the voltage across the source approaches infinity as the load resistance approaches infinity (an open circuit).

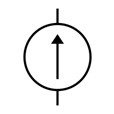


Fig.1.13 Symbol of a Current source

**1.7 CHARACTERISTICS OF OTAs**

Operational transconductance amplifiers (OTA) are devices that convert an input voltage to an output current. They are primarily voltage-to-current amplifiers. Unlike traditional operational amplifiers, or op-amps, OTAs represent a voltage-controlled current source (VCCS). Their transconductance parameter is controlled by an external, amplifier-bias current and expressed as a function of the applied voltage. Because an OTA’s output impedance is high, some operational transconductance amplifiers have on-chip controlled impedance buffers for driving resistive loads. The dependence of open-loop bandwidth, closed-loop bandwidth, and frequency responses are similar to those of conventional operational amplifiers, however. For OTA circuits that use negative feedback, there is a very close relationship between the closed-loop bandwidth, the amplifier bias current, and the closed-loop gain.

**CHAPTER 2**

**CUCKOO SEARCH ALGORITHM**

* 1. **Introduction**

Nature provides some of the efficient ways to solve problems. Algorithms imitating processes in nature/inspired from nature are referred to as Nature Inspired Algorithms.

One such area is Optimization, which is an act, process, or methodology of making something (as a design, system, or decision) as perfect, functional, or effective as possible.

Many metaheuristics (Nature Inspired) implement some form of stochastic optimization, so that the solution found is dependent on the set of random variables generated. By searching over a large set of feasible solutions, metaheuristics can often find good solutions with less computational effort than optimization algorithms, iterative methods, or simple heuristics. As such, they are useful approaches for optimization problems.

**Properties of metaheuristic algorithm.**

* Metaheuristics are strategies that guide the search process.
* The goal is to efficiently explore the search space in order to find near optimal solutions.
* Techniques that constitute metaheuristic algorithms range from simple local search procedures to complex learning processes.
* Metaheuristic algorithms are approximate and usually non-deterministic.
* Metaheuristics are not problem-specific.

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a *generation*. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations have been produced, or a satisfactory fitness level has been reached for the population.

**Genetic operators:**

**1. Mutation** is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. In mutation, the solution may change entirely from the previous solution. Hence GA can come to a better solution by using mutation. Mutation occurs during evolution according to a user-definable mutation probability. This probability should be set low. If it is set too high, the search will turn into a primitive random search.

**2. Crossover** is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. Crossover is a process of taking more than one parent solution and producing a child solution from them. There are methods for selection of the chromosomes.

3. **Selection** is the stage of a genetic algorithm in which individual genomes are chosen from a population for later breeding (using the crossover operator).

We will, now study one such algorithm, which is the Cuckoo Search Algorithm.

**2.2 Cuckoo Search Algorithm**

**2.2.1 Inspiration from Cuckoo Birds**

Cuckoo are fascinating birds, not only because of the beautiful sounds they can make, but also because of their aggressive reproduction strategy. Some species such as the *ani* and *Guira* cuckoos lay their eggs in communal nests, though they may remove others’ eggs to increase the hatching probability of their own eggs. Quite a number of species engage the obligate brood parasitism by laying their eggs in the nests of other host birds (often other species). There are three basic types of brood parasitism: intraspecific brood parasitism, cooperative breeding, and nest takeover. Some host birds can engage direct conflict with the intruding cuckoos. If a host bird discovers the eggs are not their owns, they will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the New World brood-parasitic *Tapera* have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colour and pattern of the eggs of a few chosen host species. This reduces the probability of their eggs being abandoned and thus increase their reproduction. In addition, the timing of egg-laying of some species is also amazing. Parasitic cuckoos often choose a nest where the host bird just laid its own eggs. In general, the cuckoo eggs hatch slightly earlier than their host eggs. Once the first cuckoo chick is hatched, the first instinct action it will take is to evict the host eggs by blindly propelling the eggs out of the nest, which increases the cuckoo chick’s share of food provided by its host bird. Studies also show that a cuckoo chick can also mimic the call of host chicks to gain access to more feeding opportunity.

CS has been successfully used to solve scheduling problems and to solve design optimization problems in structural engineering. In many applications like speech reorganization, job scheduling, global optimization.

**2.2.2 Levy Flights**

Various studies have shown that flight behaviour of many animals and insects has demonstrated the typical characteristics of Levy flights.

A recent study by Reynolds and Frye shows that fruit flies or *Drosophila melanogaster*, explore their landscape using a series of straight flight paths punctuated by a sudden 900 turn, leading to a Levy-flight-style intermittent scale free search pattern. Studies on human behaviour such as the Ju/’hoansi hunter-gatherer foraging patterns also show the typical feature of Levy flights. Even light can be related to Levy flights.

Levy flight is therefore a random walk; in this, the steps are defined regarding the step-lengths, which have a certain probability distribution, with the directions being random.

The next movement is based on the current position.

Subsequently, such behaviour has been applied to optimization and optimal search, and preliminary results show its promising capability.

**2.2.3 Idealised Rules**

In a nest, each egg represents a solution and cuckoo egg represents a new and good solution. The obtained solution is a new solution based on the existing one and the modification of some characteristics. In the simplest form each nest has one egg of cuckoo in which each nest will have multiple eggs represents a set of solutions.

Cuckoo search idealized such breeding behaviour and can be applied to various optimization problems.

* Each cuckoo lays one egg at a time and dumps it in a randomly chosen nest.
* The best nests with the high quality of eggs will carry to the next generations.
* The number of available host nest is fixed and if a host bird identifies the cuckoo egg with the probability of pa= 0,1 then the host bird can either throw them away or abandon them and build a new nest.

**2.3 Pseudo Code Cuckoo Search Algorithm**

**2.3.1 Flowchart depiction of Cuckoo Search Algorithm**

****

Fig. 2.1 Flow chart of CSA

**2.3.2. Pseudo Code**

**Cuckoo Search via Levy Flights**

**Begin**

Objective function *f*(*x*)*, x* = (*x*1*, ..., xd*)*T*

Generate initial population of *n* host nests *xi*

**while** (*t <*MaxGeneration) or (stop criterion)

Get a cuckoo randomly

Generate a solution by Levy flights

Evaluate its solution quality or objective value *fi*

Choose a nest among *n* (say, *j*) randomly

**if** (*fi < fj* ),

Replace *j* by the new solution *i*

**end**

A fraction (*pa*) of worse nests are abandoned

New nests/solutions are built/generated

Keep best solutions (or nests with quality solutions)

Rank the solutions and find the current best

Update *t ← t* + 1

**end while**

Postprocess results and visualization

**end**

As a further approximation, this last assumption can be approximated by replacing a fraction *pa* of the *n* host nests with new nests (with new random solutions). For a maximization problem, the quality or fitness of a solution can simply be proportional to the value of objective function.

This algorithm uses a balanced combination of a local random walk and the global explorative random walk, controlled by a switching parameter *pa*. The local random walk can be written as

⊗ ⊗ )

where are two different solutions selected randomly by random permutation, *H(u)* is a Heaviside function, is a random number drawn from a uniform distribution, and *s* is the step size. Here, ⊗ means the entry-wise product of two vectors.

The global random walk is carried out by using Levy Flights:

where,

Here alpha *>* 0 is the step size scaling factor, which should be related to the scales of the problem of interest. In most cases, we can use *α* = *O(L/*10*)*, where *L* is the scale characteristic of the problem of interest, whereas in some cases *α* = *O(L/*100*)* can be more effective and avoid flying too far. Obviously, the *α* value in these two updating equations can be different, thus leading to two different parameters, *α*1 and *α*2. Here we use *α*1 = *α*2 = *α* for simplicity.

There are several parameters in CS. Apart from the population size *n*, there are switching probability *pa*, step-size scaling factor *α*, and the Lévy exponent *λ*. However, the key parameters are *pa* and *n* because we can take *α* and *λ* as constants. By varying their values, we found that we can set *λ* = 1*.*5 and *α* = 0*.*01 for most problems.

**2.4 Levy Flights Implementation**

From the implementation point of view, the generation of random numbers with Lévy

flights consists of two steps: the choice of a random direction and the generation of steps that obey the chosen Lévy distribution. The generation of a direction should be drawn from a uniform distribution, whereas the generation of steps is quite tricky. There are a few ways of achieving this, but one of themost efficient and yet straightforward ways is to use the so-called Mantegna algorithm for a symmetric Lévy stable distribution.

However, it is not trivial to generate pseudo-random step sizes that correctly obey this Lévy distribution. In Mantegna’s algorithm, the step size *s* can be computed using two Gaussian distributions *U* and *V* via the following transformation:

Where, *U* ∼ *N(*0*, σ*2*), V* ∼ *N(*0*,* 1*).*

Here *U* ∼ *(*0*, σ*2*)* means that the samples are drawn from a Gaussian normal distribution with a zero mean and a variance of *σ*2. The variance can be calculated by:

This distribution (for *s*) obeys the expected Lévy distribution for |*s*| ≥ |*s*0|m where *s*0

is the smallest step. In principle, |*s*0| >>0, but in reality *s*0 can be taken as a sensible value such as *s*0 = 0*.*1 to 1.

These formulas look complicated, but the Гfunction is just a constant for a given *λ*. For example, when *λ* = 1, we have Г *(*1 + *λ)* = 1*,* Г *((*1 + *λ)/*2*)* = 1 and

=1

It has been proved, mathematically that the Mantegna algorithm can produce the random samples that obey the required distribution correctly.

**2.5 Choice of Parameters**

There are several parameters in Cuckoo Search. Apart from the population size *n*, there are switching probability *pa*, step-size scaling factor *α*, and the Lévy exponent *λ*. However, the key parameters are *pa* and *n* because we can take *α* and *λ* as constants. By varying their values, it is found that we can set *λ* = 1*.*5 and *α* = 0*.*01 for most problems.

For the key parameters, vary the number of host nests (or the population size *n*) and the probability *pa*. We have used *n* = 5*,* 10*,* 15*,* 20*,* 30*,* 40*,* 50*,* 100*,* 150*,* 250*,* 500 and *pa* = 0*,* 0*.*01*,* 0*.*05*,* 0*.*1*,* 0*.*15*,* 0*.*2*,* 0*.*25*,* 0*.*3*,* 0*.*4*,* 0*.*5. It is found that *n* = 15 to 40 and *pa* = 0*.*25 are sufficient for most optimization problems. Results and analysis also imply that the convergence rate, to some extent, is not sensitive to the parameters used. This means that the fine adjustment is not needed for any given problems.

Let us look at a simple example. One of the many test functions we have used is the

bivariate Michalewicz function

where *m* = 10 and *(x, y)* ∈ [0*,* 5] × [0*,* 5]. This function has a global minimum *f*∗ ≈ −1*.*8013 at *(*2*.*20319*,* 1*.*57049*)*. This global optimum can easily be found using CS, and the results are shown in Figure, where the final locations of the nests are also

marked with in the figure. Here we have used *n* = 15 nests, *α* = 1, and *pa* = 0*.*25.

From the figure, we can see that as the optimum is approaching, most nests aggregate

toward the global optimum.We also notice that the nests are also distributed at different (local) optima in the case of multimodal functions. This means that CS can find all the optima simultaneously if the number of nests are much higher than the number of local optima. This advantage may become more significant when we’re dealing with multimodal and multi-objective optimization problems.

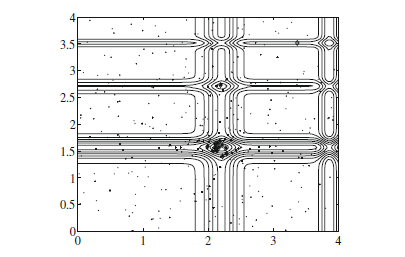
****

Fig. 2.2 Method of Optimization

**2.6 An Example**

**2.6.1 Functions**

The Cuckoo Search Algorithm was used to optimise two functions. The funtions are:

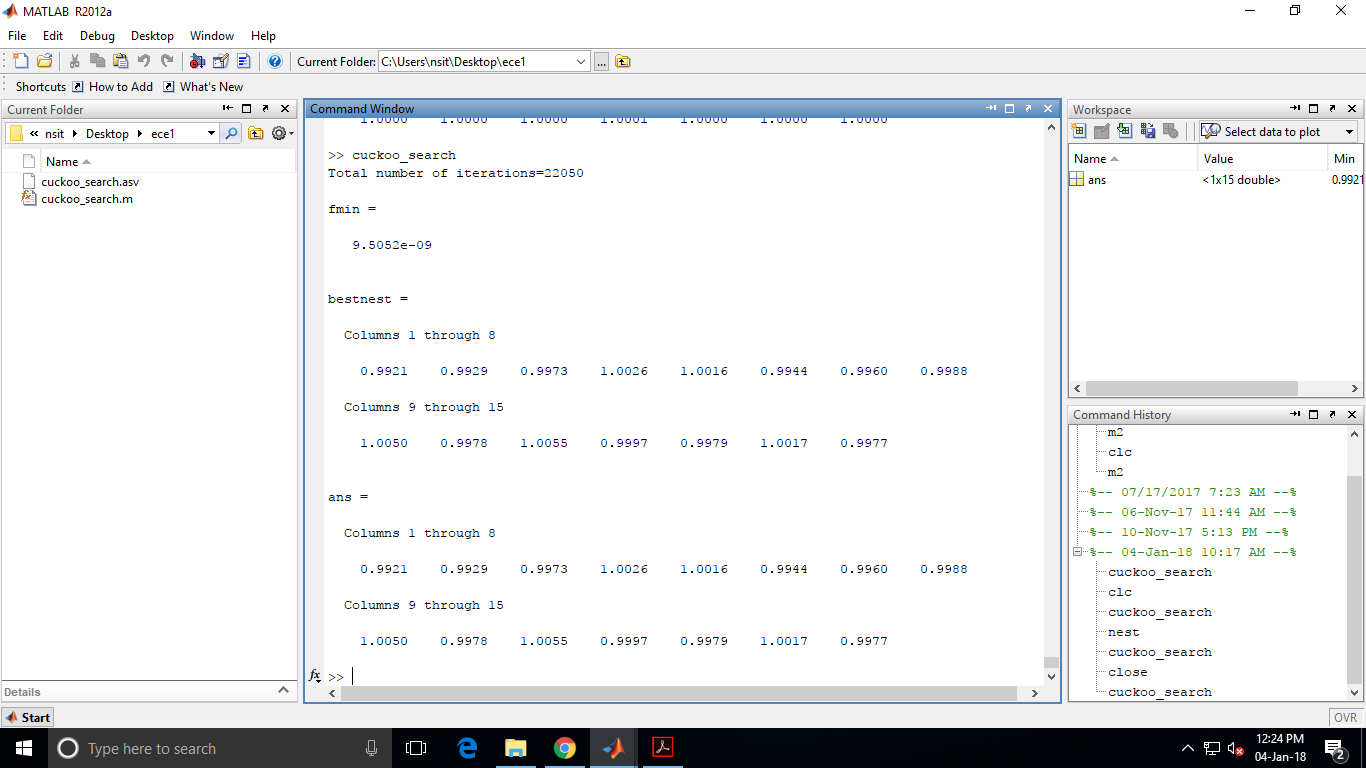
1. **Schumer Steiglitz Function**

The global minimum is located at x4 =(0,…..0) with f(x4) =0.

1. **Sphere Function**

Subject to 0≤xi≤10. The global minimum is located x\* = (0,….,0) with f(x\*)=0.

* + 1. **Results**

1. **Schumer Steiglitz Funtions**

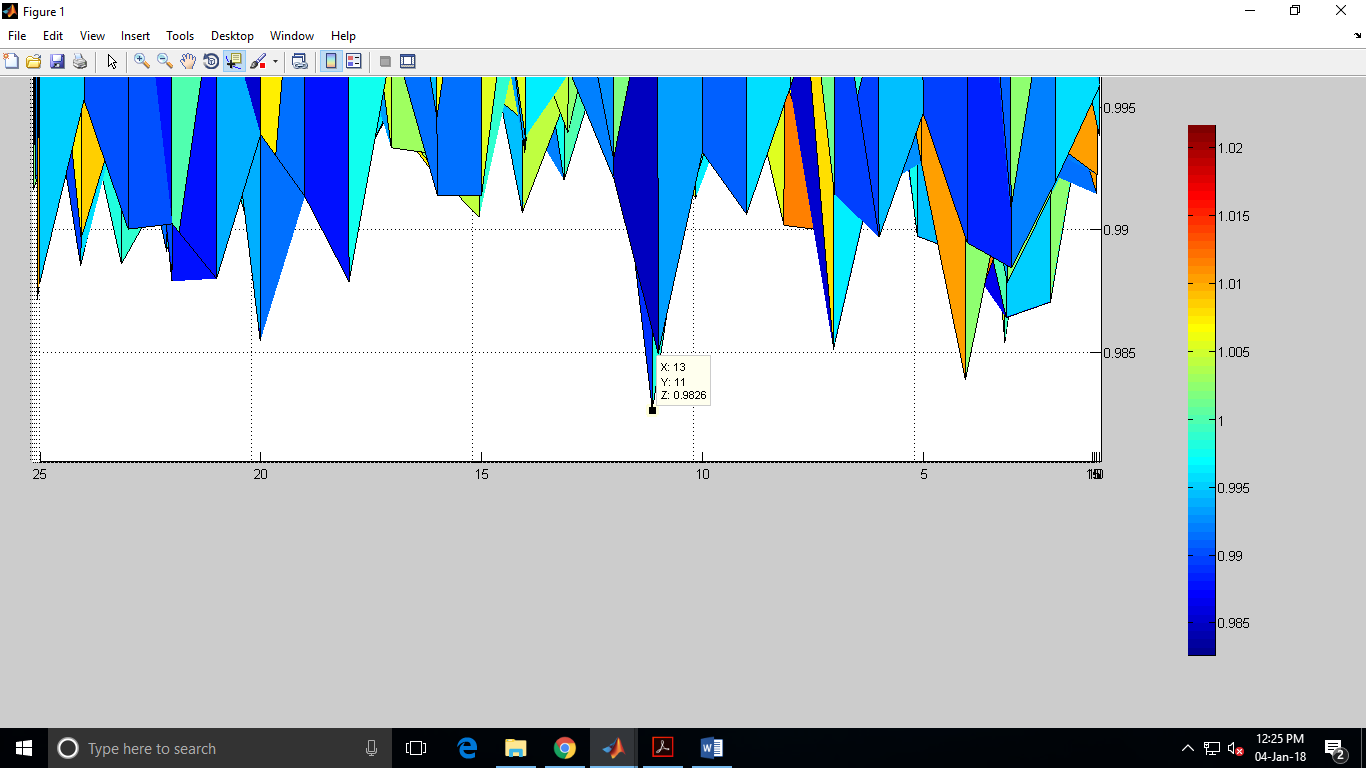
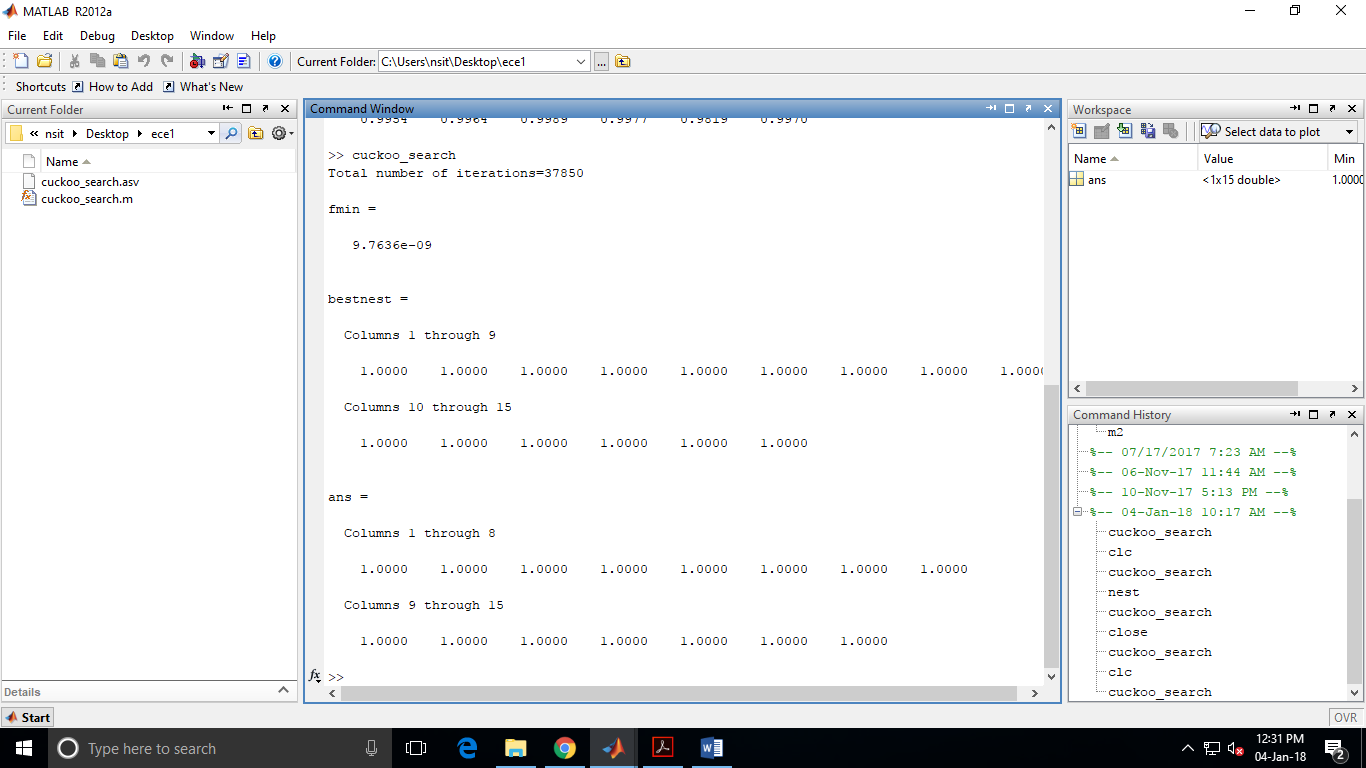


Fig. 2.3 Schumer Steiglitz Funtion

1. **Sphere Function**



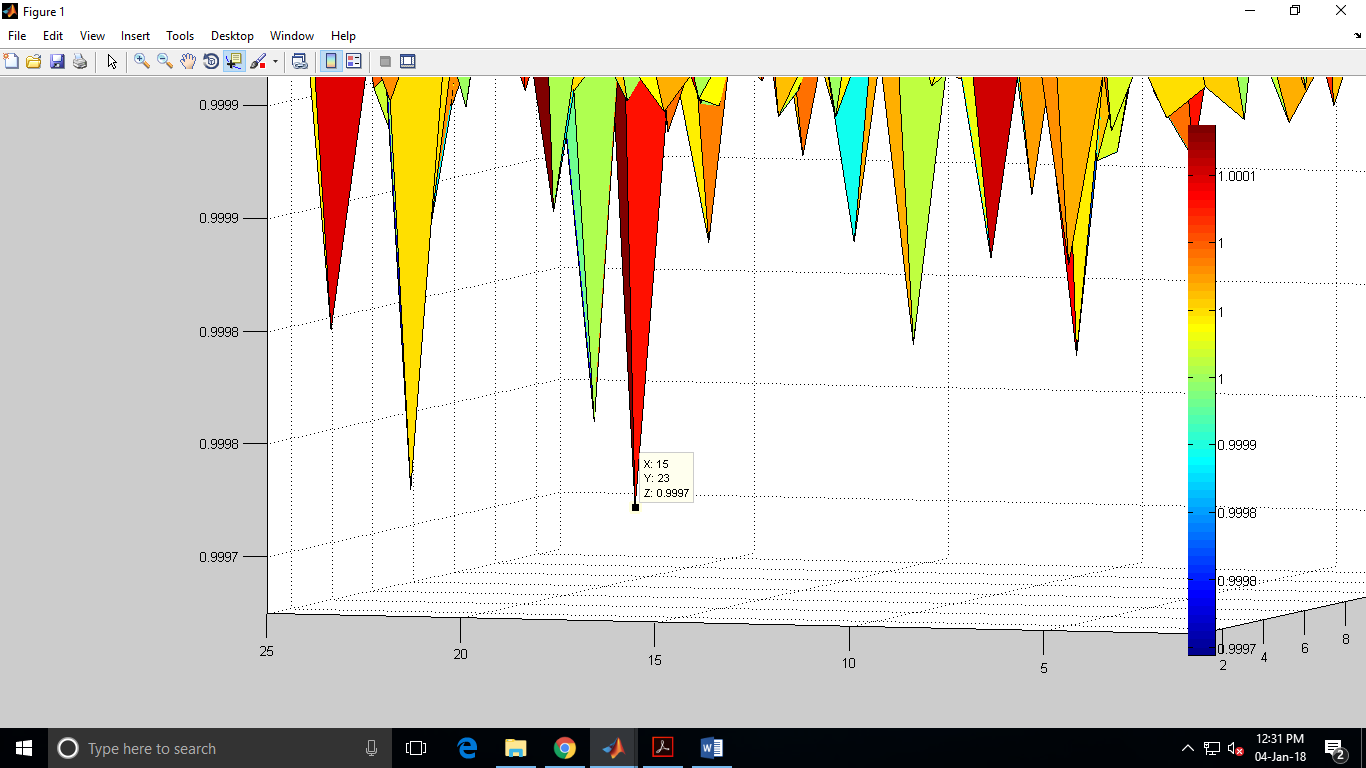


Fig.2.4 Sphere Function

**CHAPTER 3**

**CONVENTIONAL CMOS OTA DESIGN**

**3.1 Introduction**

A two stage Operational Transconductance Amplifier is a widely used analog building block. The operational transconductance amplifier (OTA) is an [amplifier](https://en.wikipedia.org/wiki/Amplifier) whose differential input voltage produces an output [current](https://en.wikipedia.org/wiki/Electric_current). Thus, it is a voltage controlled current source (VCCS).

The OTA circuit presented here is in CMOS technology and it allows circuit parameters to be univocally related to amplifier performance.

The design procedure is based on the following parameters: phase margin, slew rate , gain bandwidth product, load capacitance, input common mode range.

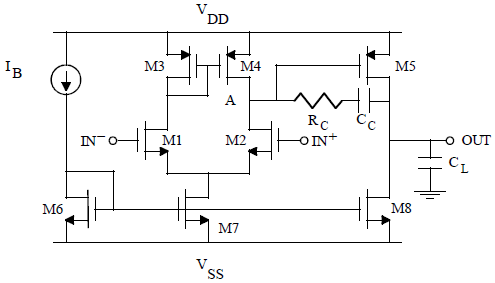


Fig. 3.1 : CMOS OTA Schematic

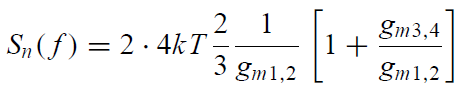
This is the simplest two stage CMOS OTA circuit. The first stage is a differential transconductance amplifier. The second (output) stage is a common source amplifier (formed by M5 and M8). M1 and M2 form a differential pair, M3 and M4 form a simple PMOS current mirror and M6 and M7 form a current repeater if (W/L)6 = (W/L)7 or a current amplifier if the two aspect ratios are not equal. Cc is the Miller compensation capacitance and Rc is used as a nulling resistor for the right half plane zero.

IB: Biasing current

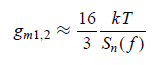
Vdd: +1.8V Vss: -1.8V

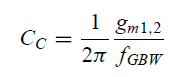
**3.2** The design procedure of the OTA is as follows:

The procedure starts from the noise requirement. Neglecting flicker noise which contributes at low frequencies, the input noise voltage spectral density of the OTA is given by

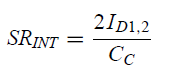


The noise analysis is carried out only for the input differential amplifier stage. To minimize noise, we assume *gm*3*,*4 *< gm*1*,*2 (which is easily met) and calculate the transconductance gain of transistors M1 and M2 from



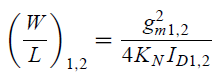
The value of Gain Bandwidth Product and gm1,2 allows us to calculate the value of the compensating capacitor Cc

The slew rate performance of the amplifier depends on the slews on both the output node of the differential stage and the output node of the second stage (i.e., theoutput of the OTA), to which we will refer as internal and external slew rate, respectively. These slew rate terms are related to the quiescent currents *ID*1*,*2 and *ID*8 according to



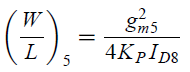
From the basic drain current equation of the Mosfet, we can write that,

C:\Users\Abhishek\Pictures\Screenshots\Screenshot (17).png

Finally the aspect ratio of M1-M2 will be

Where Kn is technology dependent parameter whose value is calculated from TSMC018 library.

C:\Users\Abhishek\Pictures\Screenshots\Screenshot (6).pngThe Transconductance gain of M5 resuts in

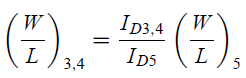
Where Mφ is the Phase Margin of the system, CL is the load capacitance and Fgbw is the gain bandwidth product. The aspect ratio of M5 is calculated as,

Rc and Cc are compensating resistors and capacitors respectively. They compensate for the right half plane zero caused by the forward path to the output.

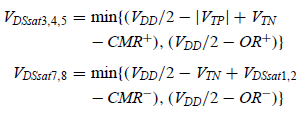
As far as transistors M3 and M4 is concerned, they contribute to the systematic offset and *CMRR*, besides affecting noise performance according to equation (1). In order to improve both offset and CMRR, accurate matching must be guaranteed by both a proper layout design and symmetrical bias conditions. This means the same drain-source voltages, other than the same aspect ratios. Consequently, we must set

C:\Users\Abhishek\Pictures\Screenshots\Screenshot (9).png

The above equation gives,



Other equations for calculation include,





**3.3 Calculations**

The default values of parameters for the above calculations are:

kT= 4.11x10-11 J

Ib=18uA

Input White Noise=10nV/

GBP=10Mhz

M(Φ)= Phase Margin=600

Slew Rate= 10V/μs

Cox=8.4183 mF/m2

μn= 259.534\*10-4 m2/Vs

μp =109.976\*10-4 m2/Vs

gm1,2= (16/3)\*4.14\*10-21/(100\*10-18)

=2.208\*10-4 Amp/V

Cc= (1/2\*π)\*(2.208\*10-4)/10\*106)

=3.5pF

Id1,2= (10/(10-6\*2))\*3.5\*10-12

=17.5uA

Id8= SR\*(Cc+Cl)=(10/10-6)\*10-12)\*(7.5)

=75uA=Id5

**(W/L)1,2=** (2.208\*10-4)^2/(4\*109.2\*17.5)

=**6.364**

Mφ= 90-arctan(107/fsp)=60

Fsp =17.32\*106 Hz

Gm5= 2\*π\*4\*17.32\*106

=435.29\*10-6 Amp/V

**(W/L)5=**  (435.29\*10-6)^2/(4\*46.3\*75)

**=13.65**

Rc= 1/(435.29\*10-6)

=2.29KΩ

**(W/L)3,4=(**17.5/75)\*13.65

**=3.17**

**(W/L)7=** 35/(109.2\*(.36875)2)

**=2.3747**

**(W/L)6=** 2\*2.3747

**=4.74**

**3.5 Observations And Results**

The circuit is simulated using LT Spice –Tool and AC, DC and Transient Analysis of the circuit is performed to verify the consistency of the simulations with the calculations.

**Table 3.1Target Parameters**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Desired Value** |
| **1.** | **DC Gain** | **>60dB** |
| **2.** | **Input White Noise** | **10nV/√ Hz** |
| **3.** | **Slew Rate: SR** | **10V/us** |
| **4.** | **Phase Margin: Mφ** | **60** |
| **5.** | **Gain Bandwidth Product** | **10MHz** |

**DC Analysis**

We performed the DC analysis of the circuit. DC analysis is performed to find the biasing point of the circuit and also we can know the limits of the power supply.

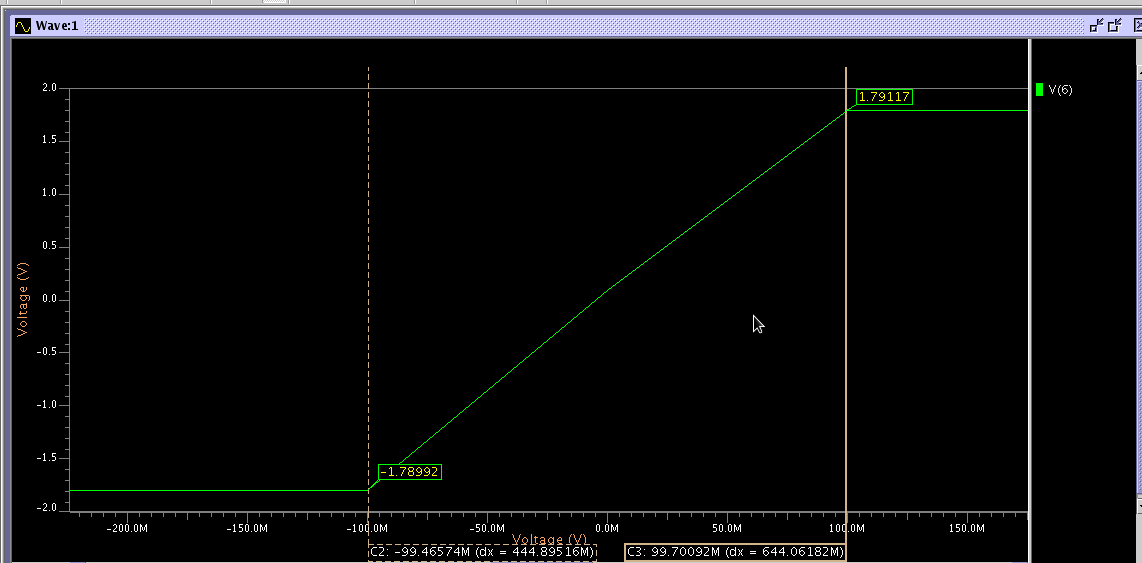
The operating range of the circuit can be found out from the above figure

Fig. 3.2 DC Analysis of Circuit

**AC Analysis**

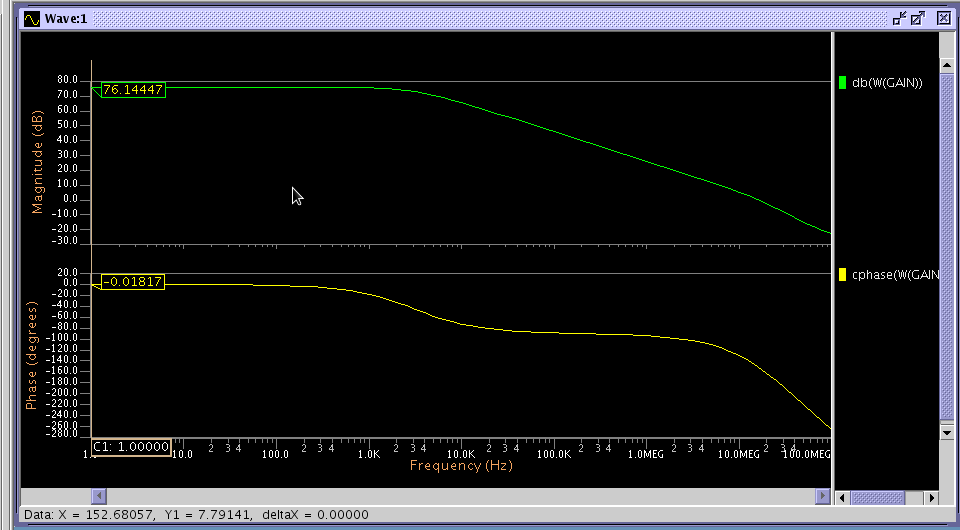
AC analysis is done to figure out the frequency response of the circuit. It helps in finding gain of the circuit and the phase margin.

Fig 3.3 Ac Analysis of Circuit

**Transient Analysis**

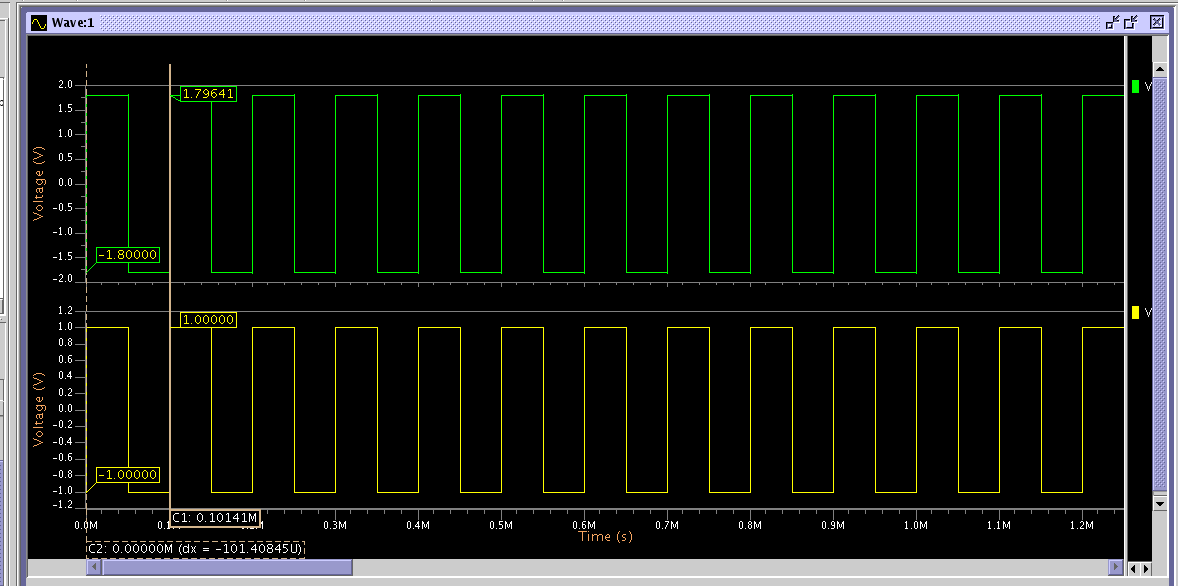
It is a time domain analysis which tells the behaviour of the circuit when time varying signals are applied to it.

Fig 3.4 Transient Analysis of Circuit.

**Transient Analysis** (done to find Slew Rate)

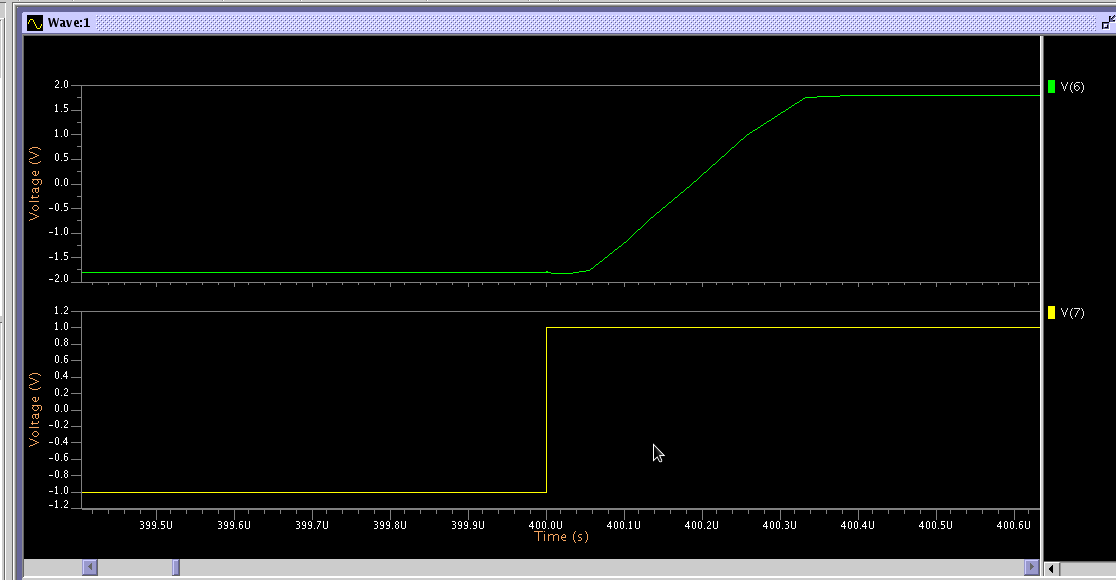
Slew Rate is defined as the rate of change of output with input. Here a step signal is applied to the input and its response is found out. From that output response Slew Rate is determined.

Fig 3.5 Transient Analysis for Slew Rate

**Table3.2 Observed Parameters**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Paramater** | **Observed Value** |
| **1.** | **DC Gain** | **76.14dB** |
| **2.** | **Input White Noise** | **12nV/√ Hz** |
| **3.** | **Slew Rate: SR** | **11.368V/us** |
| **4.** | **Phase Margin: Mφ** | **58.757** |
| **5.** | **Gain Bandwidth Product** | **8.823MHz** |

**Table 3.3 for Aspect Ratios of the Mosfets**

|  |  |
| --- | --- |
| **MOSFET** | **Aspect Ratio** |
| M1, M2 | 6.364/1 |
| M3, M4 | 3.17/1 |
| M5 | 13.65/1 |
| M6 | 2.3747/1 |
| M7 | 4.548/1 |
| M8 | 5.0883/1 |

**CHAPTER 4**

**INVERSION COEFFICIENT THEORY**

**4.1 INVERSION COEFFICIENT**

Inversion Co-efficient (I.C) is denoted by the following formula:

I.C = Id/It(W/L) (4.1)

Where Id is the drain current of the MOSFET

It is the technology current, which is fixed for a particular MOS technology, i.e, channel length and has different values for PMOS and NMOS

(W/L) is the aspect ratio of the MOS under consideration

Using inversion coefficient as a parameter, three regions of operation are defined for a MOSFET namely, the weak inversion, moderate inversion and strong inversion regions

***(a*)** **In weak inversion:**

Weak inversion occurs for MOSFETs operating at sufficiently low effective gate– source voltages (< −72 mV) where the gate–source voltage, Vgs is below the threshold voltage, Vt, by at least 72 mV for a typical bulk CMOS process at room temperature. In this region, the channel is weakly inverted and drain diffusion current dominates. MOS drain current in weak inversion is proportional to the exponential of the effective gate–source voltage.

**(b)** **In strong inversion:**

Without velocity saturation effects:

Strong inversion occurs for MOSFETs operating at sufficiently high effective gate– source voltages where the gate–source voltage is above the threshold voltage by at least 225 mV for a typical bulk CMOS process at room temperature. The channel is strongly inverted and drift current dominates. Strong inversion drain current, excluding small-geometry effects like velocity saturation and vertical field mobility reduction, is proportional to the square of the effective gate–source voltage.

**(c)** **In moderate inversion:**

Between weak and strong inversion, there is a transition region known as moderate inversion where both diffusion and drift current are significant

Table 4.1: Range of inversion coefficient for the three inversion regions is:

|  |  |  |
| --- | --- | --- |
| **Weak Inversion** | **Moderate Inversion** | **Strong Inversion** |
| <0.1 | 0.1-10 | >100 |

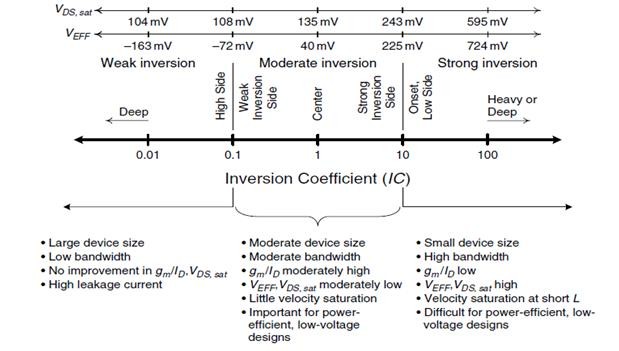


Fig 4.1 Inversion Coefficient Range

**4.2 DATA FOR CALCULATIONS**

**Io = 2\*no\*μo\*Cox\*v 2** (4.2)

Where,

Io = Technology Current

no=Substrate factor

μo = Mobility

Cox = Gate oxide capacitance = εsio2/tox

vt = Thermal voltage = 25.8 mV

εsio2 = Permittivity = εrsio2 εo

tox = Gate oxide thicknes

Various parameters for NMOS

● no = 2.4494296 (library file –tsmc018)

● μo = 259.5304169\*10-4 m2/(Vs)(library file –tsmc018)

● tox = 4.1 \*10-9 m (library file –tsmc018)

● εsio2 = 3.9\*8.85\*10-12 F/m

● Cox = 3.9\*8.85\*10-12 / 4.1\*10-9 = 8.4183 mF/m2

Solving for Io using the above equation

**Io = 0.715 nA**

Various parameters for PMOS

● no = 2 (library file –tsmc018)

● μo = 109.9762539\*10-4 m2/(Vs)(library file –tsmc018)

● tox = 4.1 \*10-9 m (library file –tsmc018)

● εsio2 = 3.9\*8.85\*10-12F/m

● Cox = 3.9\*8.85\*10-12 / 4.1\*10-9 = 8.4183 mF/m2

Solving for Io using the above equation

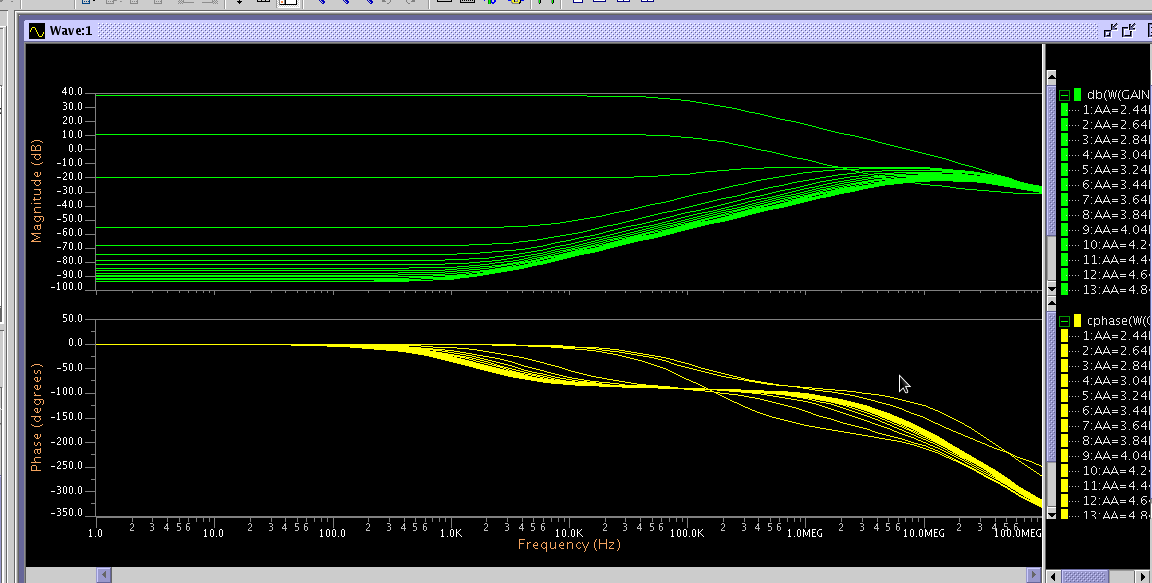
**Io = 0.24746 nA**

The length of each mosfet is 1 μm.

**4.2.1 Transistor width ranges**

**(a) M1 AND M2 (NMOS)**

These mosfets affect the gain of the circuit. MOSFETs have high gm in weak inversion as depicted in figure 4.2. Thus, in order to have high gain these MOSFETs should be in weak inversion. The affect of M1 and M2 on the gain and phase margin of the circuit is depicted as follows.

Fig 4.2 Effect of width of M1 and M2 on Gain.

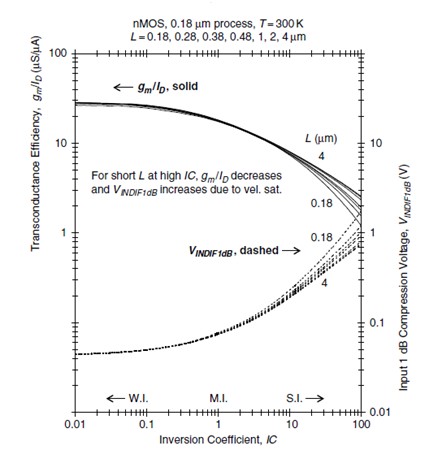
We see that if the widths of M1 and M2 are in a particular range then only the is good. Otherwise the gain falls of 3dB/dec.

Fig 4.3 Gm v/s IC

Therefore, the value of I.C. for weak region of operation taken is between 0.05 and 0.10. Using the above equation and solving for (W/L) we get the lower bound when I.C. is 0.10 and upper bound when I.C. is 0.05

Id = 15 uA

I.C.=0.10 I.C.=0.05

0.10=(15/0.715)\*(1/(W/L))

0.05=(15/0.715)\*(1/(W/L))

(W/L)L = 209.7 (W/L)U = 419.58

**W= [209.7-419.89] μm**

**(b) M3 and M4 (PMOS)**

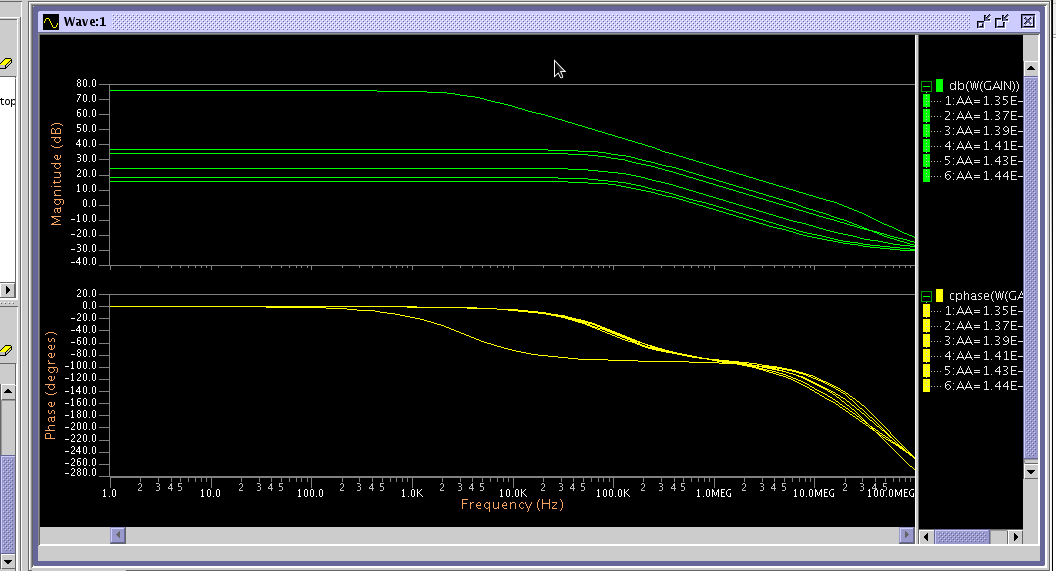
The given current mirror affect the bandwidth of the circuit and are therefore in strong region.

Fig 4.4 Effect of width of M3 M4 on gain.

For strong inversion the value of I.C. is taken from 10 to 50.

Similarly solving for widths we get the range of M3 and M4 as follows:

**W=[ 1.212-606 ] μm**

**(c) M5 (PMOS)**

It forms a common source amplifier which provides additional gain to the circuit. So we can say that the gain of the circuit will be affected drastically by the width of M5. So keeping M5 is Moderate inversion region where IC is in the range 0.1 to 10. The width of M5 would be in the range **W5=[30.27 – 3030.27]um.**

**(d) M6 and M7 and M8 (NMOS)**

They are basically driver transistors. They form a set of current mirrors and repeaters. So they can be kept in Moderate inversion region with IC ranging from 0.1 to 10. Therefore there width ranges are as follows:

**W6=[2.396-513.5]um**

**W7=[8.095-809.5]um**

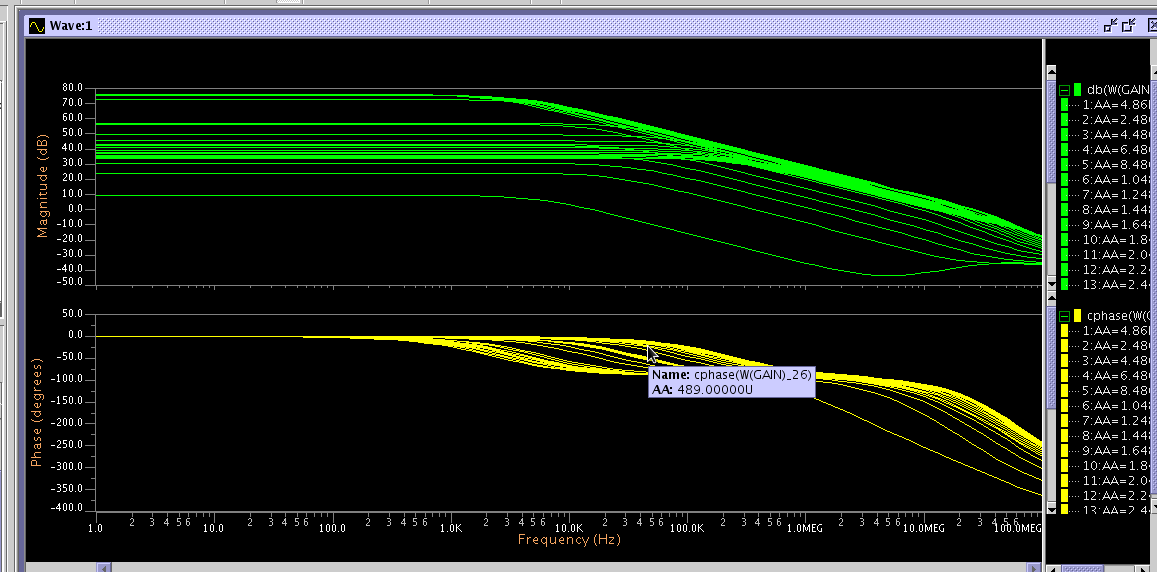
** W8=[10.489-1048.9]um**

Fig 4.5 Effect of width W7 on gain

**CHAPTER 5**

**RESULTS AND CONCLUSIONS**

**5.1 INTEGRATION FLOWCHART**

MATLAB

CODE

.spi

(netlist)

.cir

Width.txt

.aex

(outputs)

Bash Script

O/P file for Bandwidth

O/P file for Gain

Fig 5.1 Flowchart of Code

**•**The MATLAB code in the beginning of each iteration, stores suitable values of the 5 widths in the width.txt file, keeping in mind the bounds and the result provided by the previous iteration. For the first iteration it provides the file with random values in the bounds.

•Subsequently the code runs the .cir file using the run LT command of the OS. The .cir file is linked to both the width.txt file as well as the .spi file which contain the widths of the mosfets being used and the netlist respectively.

•The .cir file outputs the Gain and Bandwidth of the circuit using the .extract command in the .aex file.

•.aex file is then opened by the Bash script which was implemented by the MATLAB code on completion of the .cir command

•Bash transfers the numerical values of Gain and Bandwidth keeping in mind their units into two separate Gain.txt and Bandwidth.txt files.

•These files are then opened by MATLAB to calculate the Gain Bandwidth Product, which is then fed into the algorithm for better results and to run subsequent iterations.

**5.2 FLOWCHART OF THE AUTOMATED OPTIMIZATION METHODOLOGY**

Perform DC, AC, Transient Analysis

Analyze parameters like Gain, BW, Phase Margin

Analyze each Transistor & impact of it on each Parameter

Fix the region of each transistor in terms of IC

Set L=1u

Work out the value of W and Io(technology current)

Select the algorithm (Cuckoo Search)

Specify the objective function

Optimize the objective for (maximum) subject to given constraint

Get the Final Widths (optimized)

Fig 5.2 Flowchart of Optimisation

**5.3 Objective Function and Constraints**

The input to the algorithm is an array of MOSFET widths and the output is Gain and Phase margin of the OTA circuit.

The constraints of optimisations are:

50< Gain <80

40< Phase Margin<70

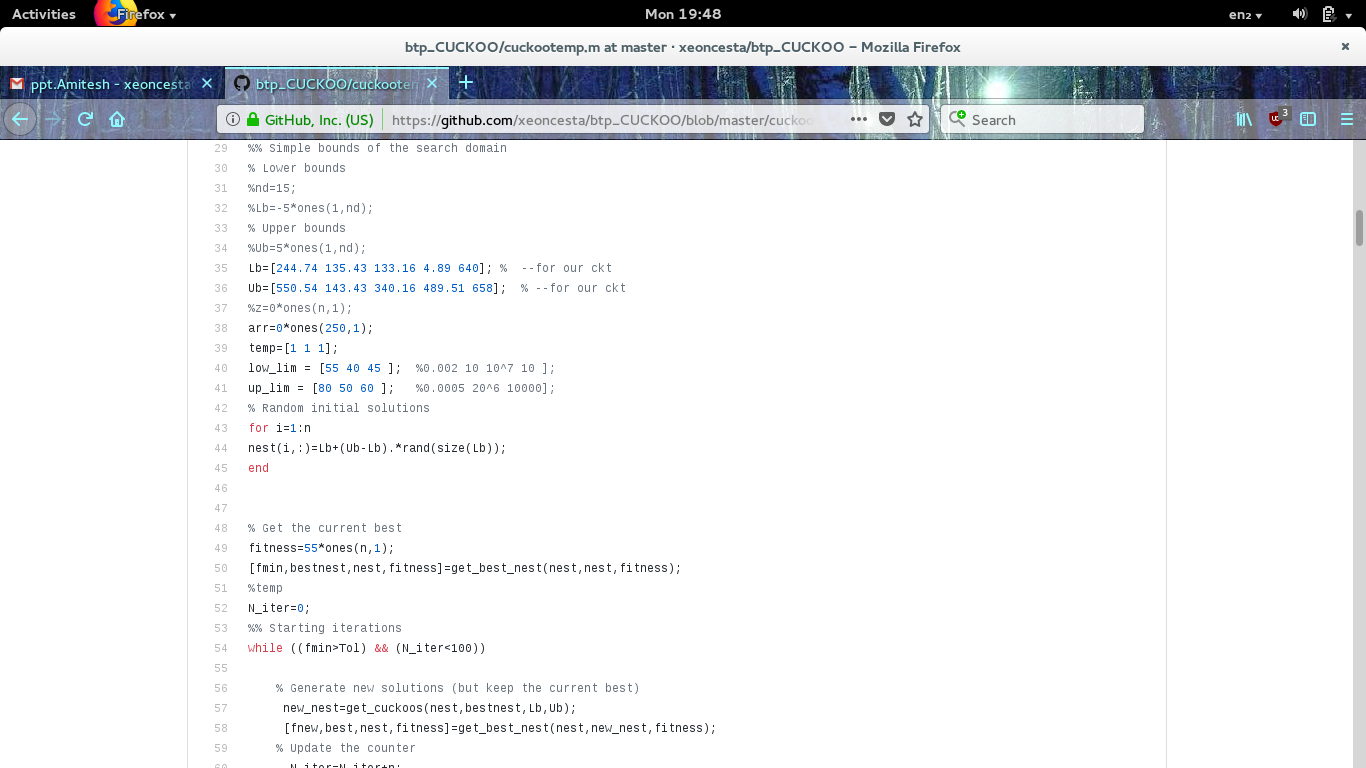
****

Fig 5.3 Objective Function and Constraints

**Set 1 of Readings**

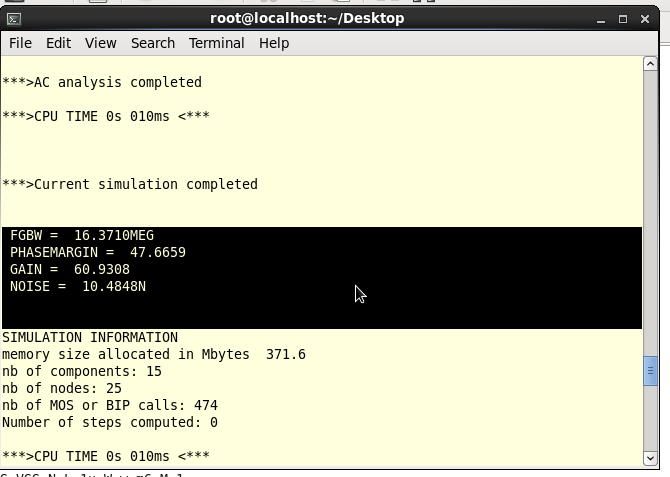


Fig 5.4 Set 1 of Gain and Phase Margin

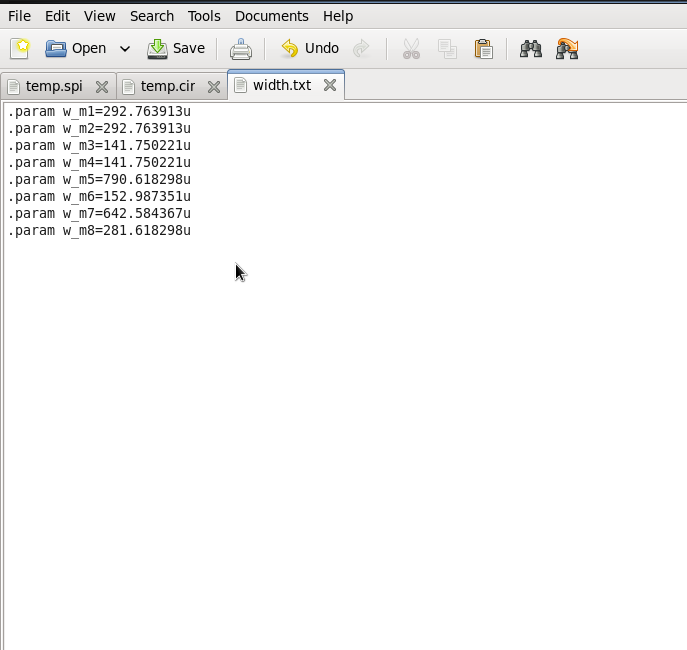


Fig 5.5 Set 1 of MOSFET widths

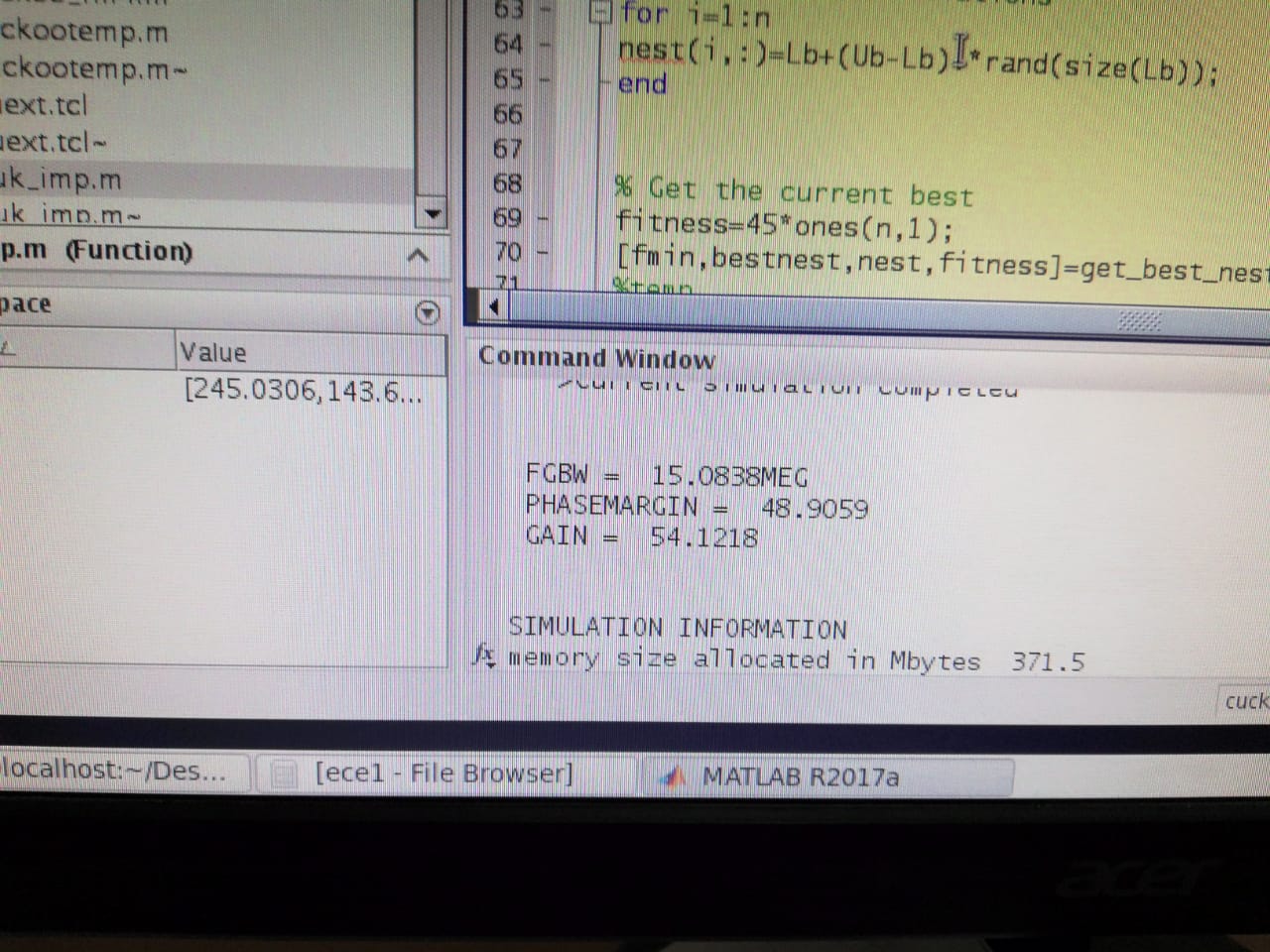
**Set 2 of Readings**

Fig 5.6 Set 2 of Gain and Phase Margin

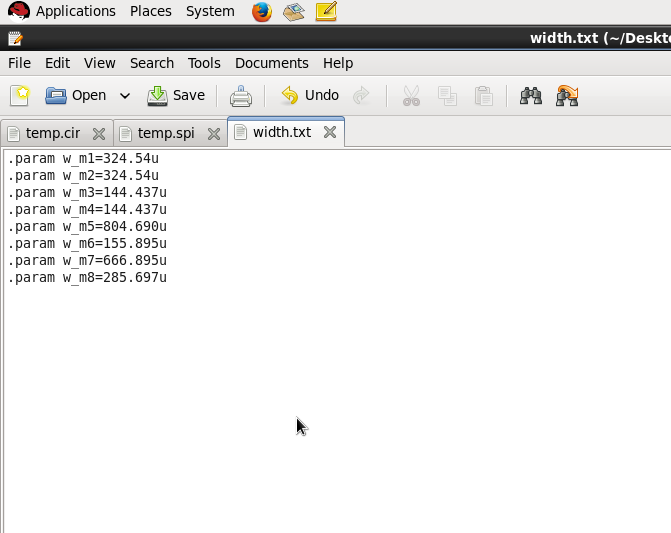


Fig 5.7 Set 2 of MOSFET widths

**Set 3 of Readings**

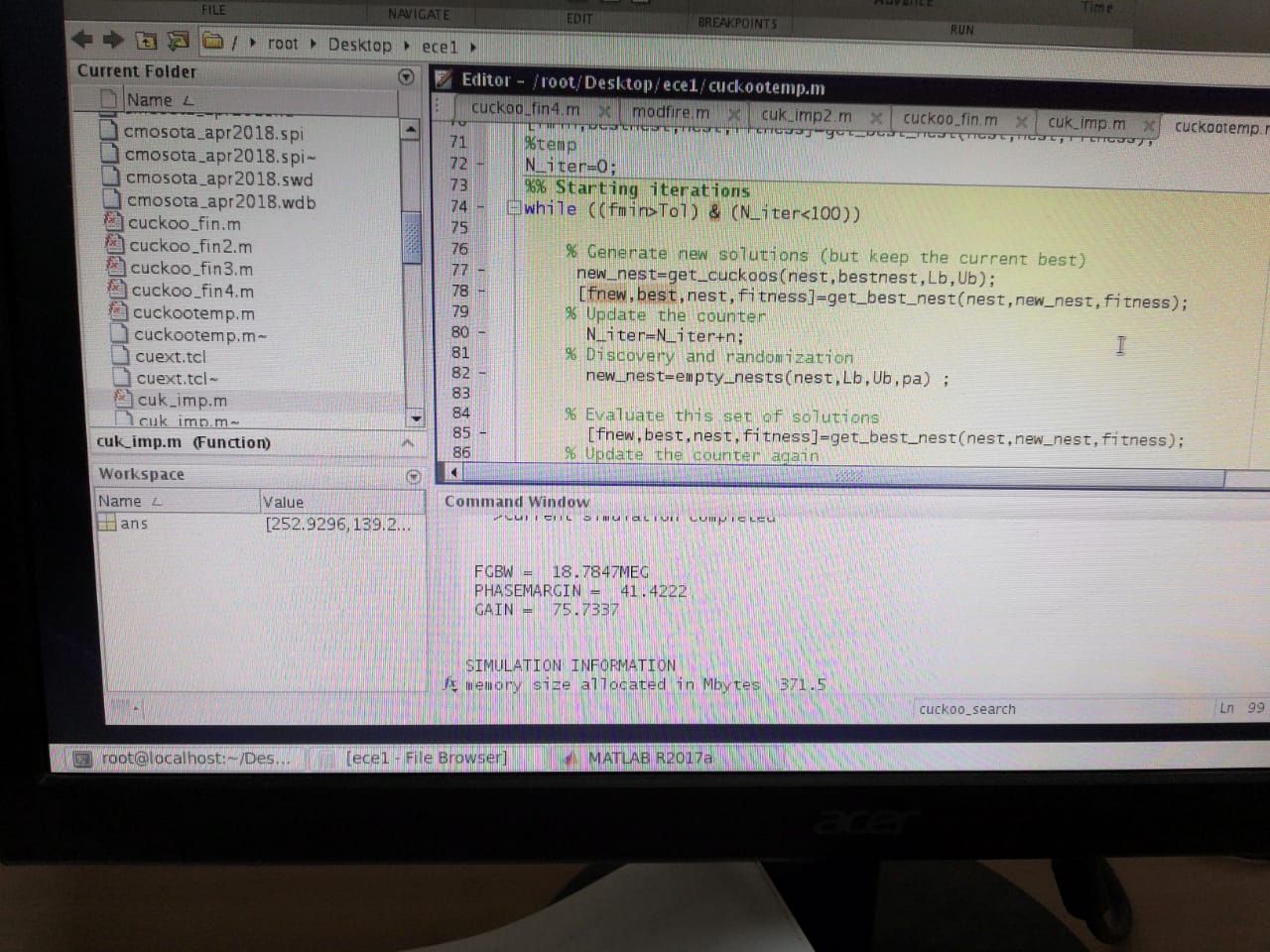


Fig 5.8 Set 3 of Gain and Phase Margin

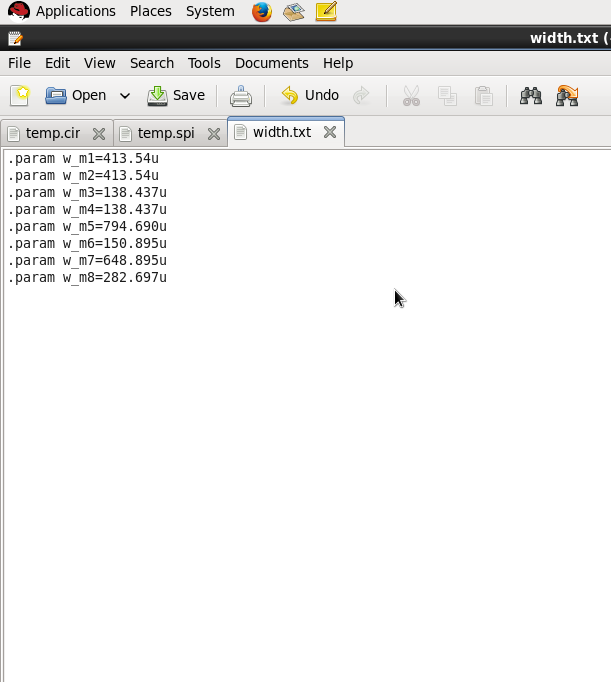


Fig 5.9 Set 3 of MOSFET widths

**TABLE 5.1 FOR PARAMETERS AND MOSFETs WIDTHS**

|  |  |  |  |
| --- | --- | --- | --- |
| **ITERATIONS** | **500** | **1500** | **2500** |
| **Best Gain(dB)** | **75.73** | **54.12** | **60.93** |
| **Best GBP(MEG)** | **18.78** | **15.08** | **16.37** |
| **Phase Margin** | **41.42** | **48.90** | **47.66** |
| **M1 M2(um)** | **413.54** | **324.54** | **292.76** |
| **M3 M4(um)** | **138.43** | **144.43** | **141.75** |
| **M5(um)** | **794.60** | **804.69** | **790.61** |
| **M6(um)** | **150.89** | **155.89** | **152.98** |
| **M7(um)** | **648.89** | **666.89** | **642.58** |
| **M8(um)** | **282.69** | **285.67** | **285.69** |

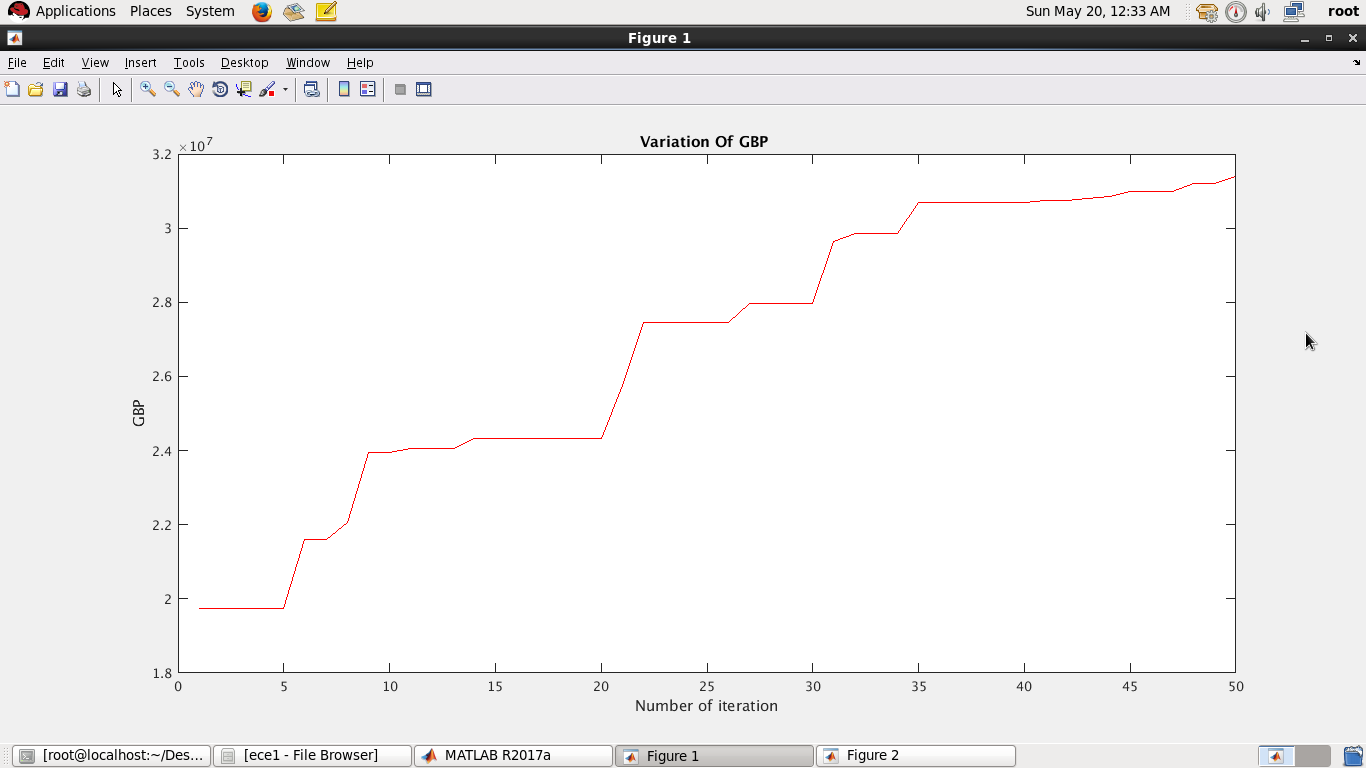
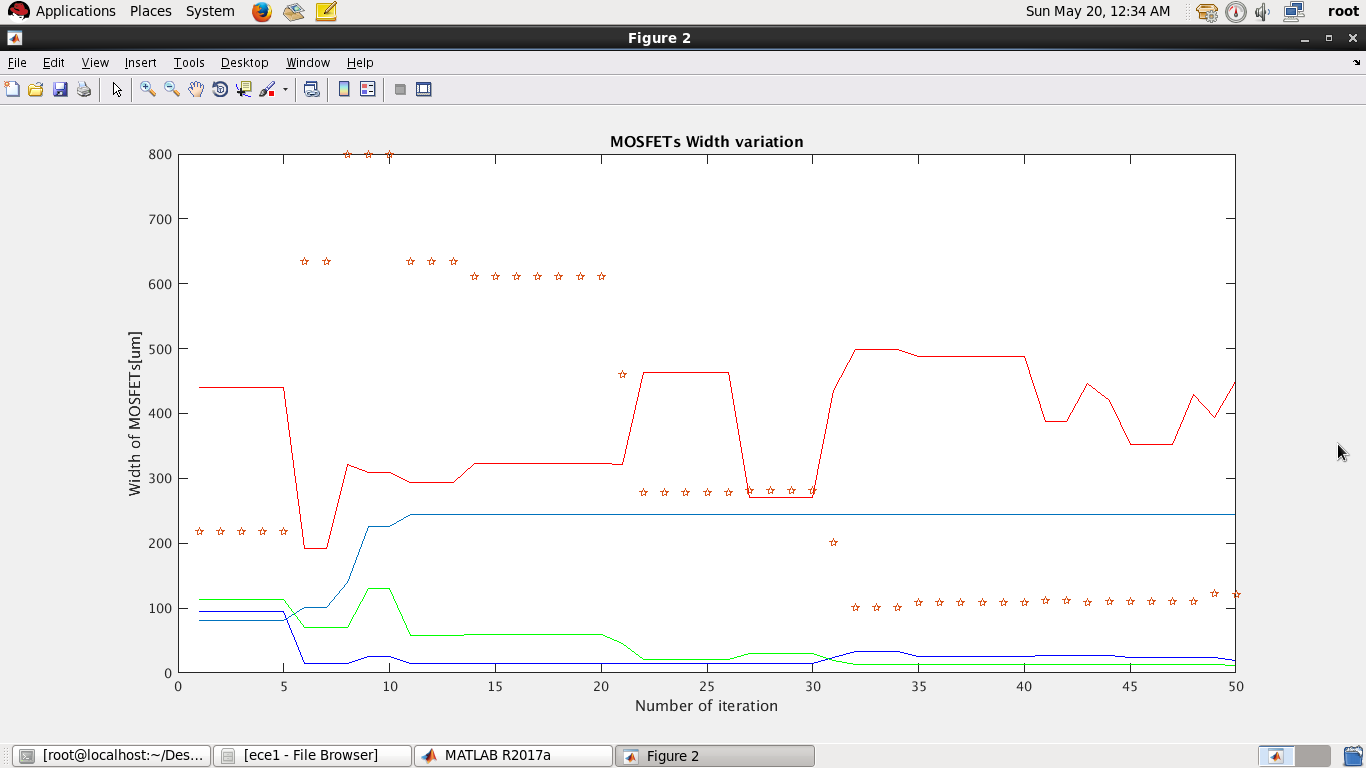


Fig 5.10 GBP Variation with number of iterartions

 Fig 5.11 Mosfet Width variation with number of iteration

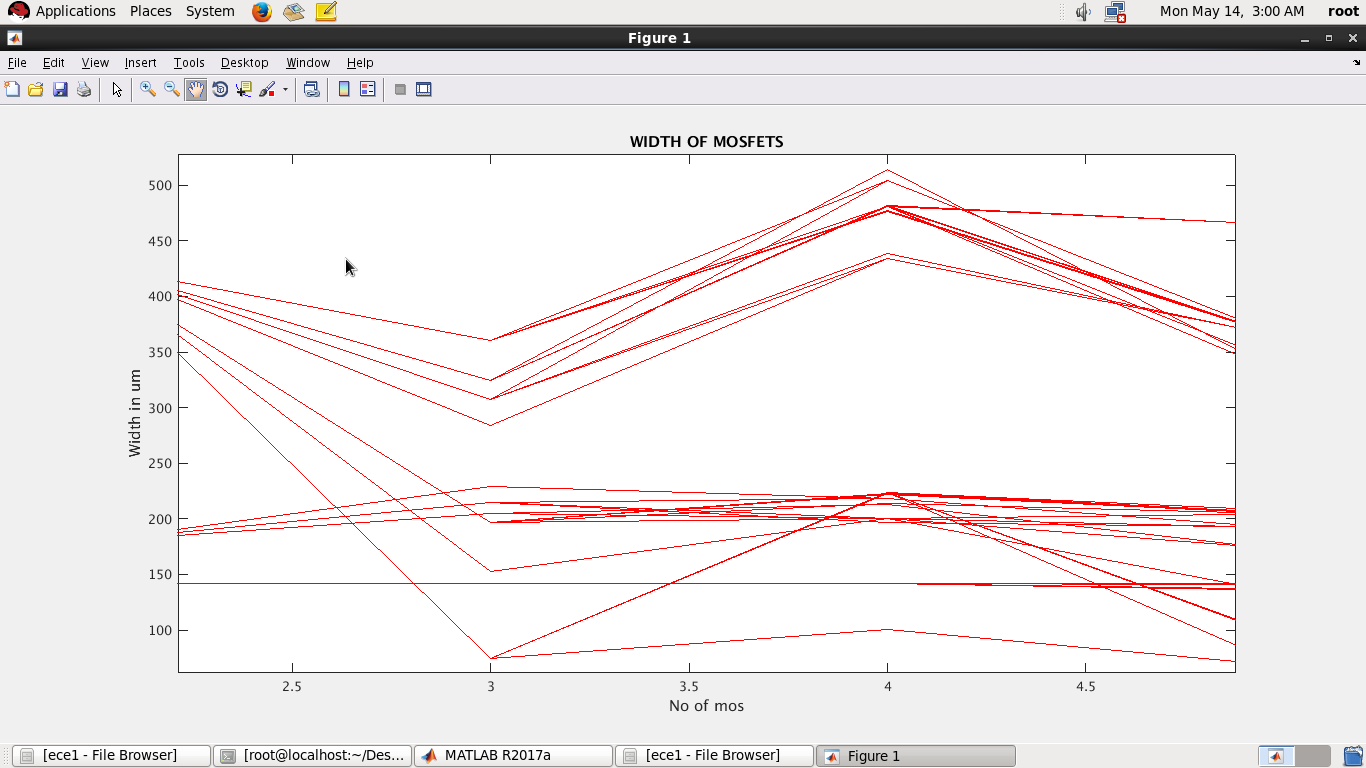


Fig 5.11 Mosfets values at the end of each iteration

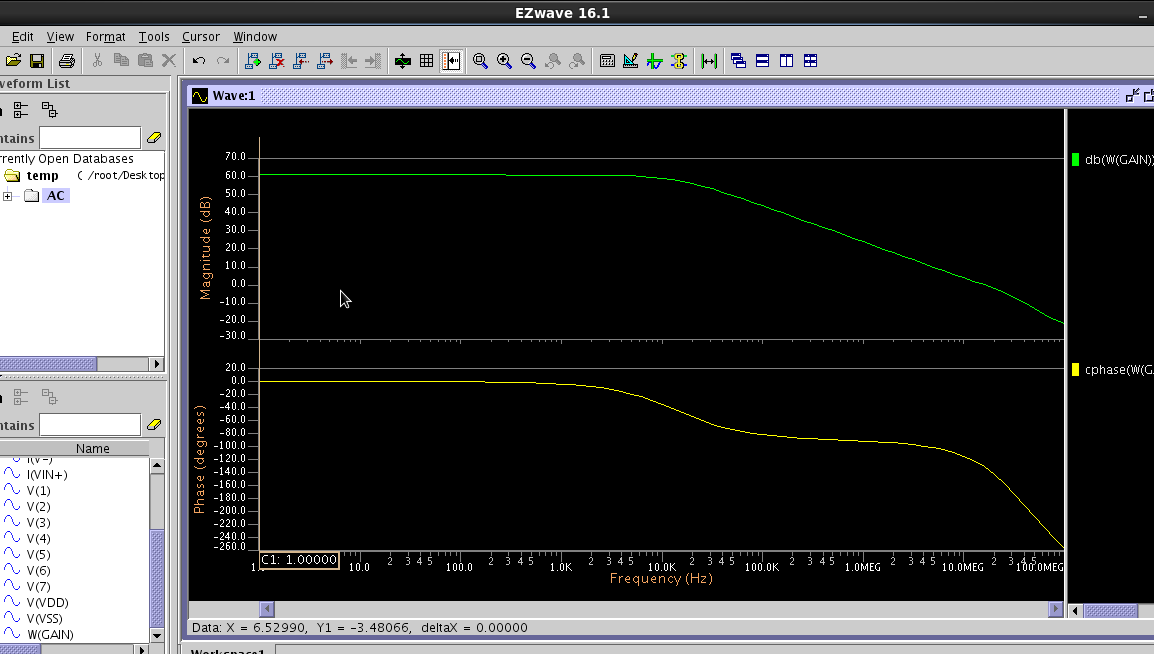


Fig 5.12 AC Analysis for 2500 iterations

**REFERENCES**

[1] David M. Binkley, *Tradeoffs and Optimization in Analog CMOS Design,* West

Sussex, England John Wiley & Sons Ltd

[2] Allen Holberg, et al. *CMOS Analog Circuit Design*, 2nd Ed. , Oxford University

Press, 2002

[3] Xin-She Yang, *Nature-Inspired Optimization Algorithms*, School of Science and

Technology Middlesex University London, First Edition 2014

[4] Xin-She Yang, *Fireﬂy Algorithm: Recent Advances and Applications,* School of

Science and Technology Middlesex University London, First Edition 2014

[5] MO Yuan-bin, *Optimal Choice of Parameters for Firefly Algorithm,* College of

Science, Guangxi University for Nationalities, 2013

[6] Jianhai YU, “A Design Method in CMOS Operational Amplifier Optimization based on Genetic Algorithm”, *WSEAS Transactions on Circuits and Systems*, vol.

[7] Taiwan Semiconductor Manufacturing Company, *TSMC018 Library*

[8] SPICE, *LT Reference Manual*

[9] SPICE, *LT User Guide*

**APPENDIX**

**A.1 LT-Spice Circuit Simulation Software**

The LT™ analog simulator is the core component of a comprehensive suite of analog and mixed-signal simulation tools. LT offers a unique partitioning scheme allowing the use of different algorithms on differing portions of design. It allows the user a flexible control of simulation accuracy using a wide range of device model libraries, and gives a high accuracy yield in combination with high speed and high performance.

The following is a list of the major product features of LT:

● LT is the core technology allowing to address RF simulation (LT RF) and mixed-signal (ADVance MS and ADVance MS Mach)

● Simulation of very large circuits (up to around 300,000 transistors) in time and frequency domains

● 3 to 10 gain in simulation speed over other commercial SPICE simulators, while maintaining same accuracy

● Three complementary transient simulation algorithms (OSR, Newton, IEM)

● Flexible user control of simulation accuracy

● Unique transient noise algorithm

● Advanced analysis options such as pole-zero, enhanced Monte-Carlo analysis

● Reliability simulation

● Extensive device model libraries including leading MOS, bipolar and MESFET

transistor models such as the tsmc018, tcmc09 and HICUM

● IBIS (I/O Buffer Information Specification) model support

● Integration into SPICE IC flow, consisting of Design Architect IC for schematic capture, IC station for the layout side, and Calibre/CalibrexRC for DRC/LVS and extraction. This flow provides a complete, front-to-back design and verification environment for analog, mixed-signal and RF.

**Commands used in LT Spice**

● **.include**

○ It is used to include text code other than the what is included in the .cir file

○ Can be used to include the .mod, .txt, .spi, etc files

● **.plot**

○ Used to save plots or the .wdb files for the required circuit

● **.extract**

○ Can be used to extract extra information onto the .aex file to be used for further analysis

○ Phase Margin, Bandwidth and Gain can be extracted by using it.

● **.param**

○ Can be used for parametric analysis

**ANALYSIS**

Can be one of the following:

● DC

Specifies that the plots are required for a DC analysis. Provides compatibility with

SPICE.

● AC

Specifies that the plots are required for an AC analysis. Provides compatibility with SPICE.

● TRAN

Specifies that the plots are required for a transient analysis. Provides compatibility with SPICE.

**A.2 MATLAB**

MATLAB (**mat**rix **lab**oratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. Developed by Mathworks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages including C,C++,Java and Fortran. .OP

MATLAB is used as an interface between the LT-Spice and The Tool Command Language. It dumps the simulation results into a file using Bash and latter these results are used for the Optimization purpose using the Genetic Algorithm.

Commands used in MATLAB

● **FPRINTF** :-Write formatted data to file.

FPRINTF(FID,FORMAT,A,...) formats the data in the real part of array A (and in any additional array arguments), under control of the specified FORMAT string, and writes it to the file associated with file identifier FID. COUNT is the number of bytes

successfully written. FID is an integer file identifier obtained from FOPEN. It can also be

1 for standard output (the screen) or 2 for standard error. If FID is omitted, output goes to the screen.

● **FCLOSE**:- Close file.

FCLOSE(FID) closes the file associated with file identifier FID, which is an integer value obtained from an earlier call to FOPEN. FCLOSE returns 0 if successful or -1 if not. If FID does not represent an open file, or if it is equal to 0 (standard input), 1 (standard output), or 2 (standard error), FCLOSE throws an error.

● **DISP** :-Display array.

DISP(X) displays the array, without printing the array name. In all other ways it's the same as leaving the semicolon off an expression except that empty arrays don't display.

● **FSCANF** :-Read formatted data from file.

[A,COUNT] = FSCANF(FID,FORMAT,SIZE) reads data from the file specified by file identifier FID, converts it according to the specified FORMAT string, and returns it in matrix A. COUNT is an optional output argument that returns the number of elements successfully read.

● **FOPEN** :- Open file.

FID = FOPEN(FILENAME) opens the file FILENAME for read access. FILENAME is a string containing the name of the file to be opened.

● **CLEAR**: - Clear variables and functions from memory.

**A.3 Netlist for the circuit**

.INCLUDE "$ADK/technology/ic/models/tsmc018.mod"

.CONNECT GROUND 0

.global VDD VSS GROUND

V2 VSS GROUND DC -1.8V

V1 VDD GROUND DC 1.8V

VIN- 8 GROUND DC 0

VIN+ 7 GROUND DC 0

\*VIN+ 7 GROUND AC 1m

\*VIN+ 7 GROUND PULSE (-1 1 1FS 1FS 1FS 50US 100US)

IBIAS VDD 1 DC 35uA

CC 2 6 3.5p

CL 6 GROUND 4p

RC 3 2 2.3k

M8 6 1 VSS VSS N L=180n W=0.9159u M=1

M7 5 1 VSS VSS N L=180n W=0.4274u M=1

M6 1 1 VSS VSS N L=180n W=0.4274u M=1

M5 6 3 VDD VDD P L=180n W=2.4516u M=1

M4 3 4 VDD VDD P L=180n W=0.5706u M=1

M3 4 4 VDD VDD P L=180n W=0.5706u M=1

M2 3 7 5 5 N L=180n W=1.14552u M=1

M1 4 8 5 5 N L=180n W=1.14552u M=1

.OP

.END

**A.4 Bash Scripting**

**Bash** is a [Unix shell](https://en.wikipedia.org/wiki/Unix_shell) and [command language](https://en.wikipedia.org/wiki/Command_language) written by [Brian Fox](https://en.wikipedia.org/wiki/Brian_Fox_(computer_programmer)) for the [GNU Project](https://en.wikipedia.org/wiki/GNU_Project) as a [free software](https://en.wikipedia.org/wiki/Free_software) replacement for the [Bourne shell](https://en.wikipedia.org/wiki/Bourne_shell). First released in 1989, it has been distributed widely as the default [login](https://en.wikipedia.org/wiki/Login) shell for most [Linux](https://en.wikipedia.org/wiki/Linux) distributions and [Apple's](https://en.wikipedia.org/wiki/Apple_Inc.) [macOS](https://en.wikipedia.org/wiki/MacOS) (formerly OS X). A version [is also available for Windows 10](https://en.wikipedia.org/wiki/Windows_Subsystem_for_Linux).

Bash is a [command processor](https://en.wikipedia.org/wiki/Command-line_interpreter) that typically runs in a text window, where the user types commands that cause actions. Bash can also read and execute commands from a file, called a [shell script](https://en.wikipedia.org/wiki/Shell_script). Like all Unix shells, it supports filename [globbing](https://en.wikipedia.org/wiki/Glob_(programming)" \o "Glob (programming)) (wildcard matching), [piping](https://en.wikipedia.org/wiki/Pipeline_(Unix)), [here documents](https://en.wikipedia.org/wiki/Here_document), [command substitution](https://en.wikipedia.org/wiki/Command_substitution), [variables](https://en.wikipedia.org/wiki/Variable_(programming)), and [control structures](https://en.wikipedia.org/wiki/Control_flow) for [condition-testing](https://en.wikipedia.org/wiki/Conditional_(programming)) and [iteration](https://en.wikipedia.org/wiki/Iteration). The [keywords](https://en.wikipedia.org/wiki/Keyword_(computer_programming)), [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)) and other basic features of the [language](https://en.wikipedia.org/wiki/Language_(computer_science)) are all copied from [sh](https://en.wikipedia.org/wiki/Bourne_shell). Other features, e.g., [history](https://en.wikipedia.org/wiki/C_shell#History), are copied from [csh](https://en.wikipedia.org/wiki/C_shell" \o "C shell) and [ksh](https://en.wikipedia.org/wiki/Korn_shell" \o "Korn shell). Bash is a [POSIX](https://en.wikipedia.org/wiki/POSIX)-compliant shell, but with a number of extensions.

The shell's name is an acronym for Bourne-again shell, a [pun](https://en.wikipedia.org/wiki/Pun) on the name of the Bourne shell that it replacesand on the common term "[born again](https://en.wikipedia.org/wiki/Born_again)".

**A.5 TSMC018 Library**

\* DATE: Jun 15/04

\* LOT: T44E WAF: 3009

\* Temperature\_parameters=Default

.MODEL NOTCHEDROW C

.MODEL HR R

.MODEL N NMOS ( LEVEL = 53

+VERSION = 3.1 TNOM = 27 TOX = 4.1E-9

+XJ = 1E-7 NCH = 2.3549E17 VTH0 = 0.3725327

+K1 = 0.5933684 K2 = 2.050755E-3 K3 = 1E-3

+K3B = 4.5116437 W0 = 1E-7 NLX = 1.870758E-7

+DVT0W = 0 DVT1W = 0 DVT2W = 0

+DVT0 = 1.3621338 DVT1 = 0.3845146 DVT2 = 0.0577255

+U0 = 259.5304169 UA = -1.413292E-9 UB = 2.229959E-18

+UC = 4.525942E-11 VSAT = 9.411671E4 A0 = 1.7572867

+AGS = 0.3740333 B0 = -7.087476E-9 B1 = -1E-7

+KETA = -4.331915E-3 A1 = 0 A2 = 1

+RDSW = 111.886044 PRWG = 0.5 PRWB = -0.2

+WR = 1 WINT = 0 LINT = 1.701524E-8

+XL = 0 XW = -1E-8 DWG = -1.365589E-8

+DWB = 1.045599E-8 VOFF = -0.0927546 NFACTOR = 2.4494296

+CIT = 0 CDSC = 2.4E-4 CDSCD = 0

+CDSCB = 0 ETA0 = 3.175457E-3 ETAB = 3.494694E-5

+DSUB = 0.0175288 PCLM = 0.7273497 PDIBLC1 = 0.1886574

+PDIBLC2 = 2.617136E-3 PDIBLCB = -0.1 DROUT = 0.7779462

+PSCBE1 = 3.488238E10 PSCBE2 = 6.841553E-10 PVAG = 0.0162206

+DELTA = 0.01 RSH = 6.5 MOBMOD = 1

+PRT = 0 UTE = -1.5 KT1 = -0.11

+KT1L = 0 KT2 = 0.022 UA1 = 4.31E-9

+UB1 = -7.61E-18 UC1 = -5.6E-11 AT = 3.3E4

+WL = 0 WLN = 1 WW = 0

+WWN = 1 WWL = 0 LL = 0

+LLN = 1 LW = 0 LWN = 1

+LWL = 0 CAPMOD = 2 XPART = 0.5

+CGDO = 8.53E-10 CGSO = 8.53E-10 CGBO = 1E-12

+CJ = 9.513993E-4 PB = 0.8 MJ = 0.3773625

+CJSW = 2.600853E-10 PBSW = 0.8157101 MJSW = 0.1004233

+CJSWG = 3.3E-10 PBSWG = 0.8157101 MJSWG = 0.1004233

+CF = 0 PVTH0 = -8.863347E-4 PRDSW = -3.6877287

+PK2 = 3.730349E-4 WKETA = 6.284186E-3 LKETA = -0.0106193

+PU0 = 16.6114107 PUA = 6.572846E-11 PUB = 0

+PVSAT = 1.112243E3 PETA0 = 1.002968E-4 PKETA = -2.906037E-3 )

\*

.MODEL P PMOS ( LEVEL = 53

+VERSION = 3.1 TNOM = 27 TOX = 4.1E-9

+XJ = 1E-7 NCH = 4.1589E17 VTH0 = -0.3948389

+K1 = 0.5763529 K2 = 0.0289236 K3 = 0

+K3B = 13.8420955 W0 = 1E-6 NLX = 1.337719E-7

+DVT0W = 0 DVT1W = 0 DVT2W = 0

+DVT0 = 0.5281977 DVT1 = 0.2185978 DVT2 = 0.1

+U0 = 109.9762536 UA = 1.325075E-9 UB = 1.577494E-21

+UC = -1E-10 VSAT = 1.910164E5 A0 = 1.7233027

+AGS = 0.3631032 B0 = 2.336565E-7 B1 = 5.517259E-7

+KETA = 0.0217218 A1 = 0.3935816 A2 = 0.401311

+RDSW = 252.7123939 PRWG = 0.5 PRWB = 0.0158894

+WR = 1 WINT = 0 LINT = 2.718137E-8

+XL = 0 XW = -1E-8 DWG = -4.363993E-8

+DWB = 8.876273E-10 VOFF = -0.0942201 NFACTOR = 2

+CIT = 0 CDSC = 2.4E-4 CDSCD = 0

+CDSCB = 0 ETA0 = 0.2091053 ETAB = -0.1097233

+DSUB = 1.2513945 PCLM = 2.1999615 PDIBLC1 = 1.238047E-3

+PDIBLC2 = 0.0402861 PDIBLCB = -1E-3 DROUT = 0

+PSCBE1 = 1.034924E10 PSCBE2 = 2.991339E-9 PVAG = 15

+DELTA = 0.01 RSH = 7.5 MOBMOD = 1

+PRT = 0 UTE = -1.5 KT1 = -0.11

+KT1L = 0 KT2 = 0.022 UA1 = 4.31E-9

+UB1 = -7.61E-18 UC1 = -5.6E-11 AT = 3.3E4

+WL = 0 WLN = 1 WW = 0

+WWN = 1 WWL = 0 LL = 0

+LLN = 1 LW = 0 LWN = 1

+LWL = 0 CAPMOD = 2 XPART = 0.5

+CGDO = 6.28E-10 CGSO = 6.28E-10 CGBO = 1E-12

+CJ = 1.160855E-3 PB = 0.8484374 MJ = 0.4079216

+CJSW = 2.306564E-10 PBSW = 0.842712 MJSW = 0.3673317

+CJSWG = 4.22E-10 PBSWG = 0.842712 MJSWG = 0.3673317

+CF = 0 PVTH0 = 2.619929E-3 PRDSW = 1.0634509

+PK2 = 1.940657E-3 WKETA = 0.0355444 LKETA = -3.037019E-3

+PU0 = -1.0227548 PUA = -4.36707E-11 PUB = 1E-21

+PVSAT = -50 PETA0 = 1E-4 PKETA = -5.167295E-3 )

\*