Modeling of Nonlinear *I-V* Characteristics for Submicron GaAs MESFETs

A thesis submitted to the Department of Electrical and Electronic Engineering of BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, Dhaka for partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING

by

Md. Moniruzzaman

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY

May 2002

This thesis entitled "Modeling of Nonlinear *I-V* Characteristics for Submicron GaAs MESFETs" submitted by Md. Moniruzzaman, Roll No. 9606201F, Session 1995-96-97 to the Department of Electrical & Electronic Engineering, BUET, Dhaka has been accepted as satisfactory in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING.

BOARD OF EXAMINERS

1. Islam 07/05/2002

Dr. Md. Shafiqul Islam
Associate Professor
Department of EEE
BUET, Dhaka-1000.

Chairman
(Supervisor)

2. 07/05/02

Dr. Md. Nasim Ahmed Dewan
Assistant Professor
Department of EEE
BUET, Dhaka-1000.

Member

3. JAlam 7.05.02

Dr. Mohammad Jahangir Alam
Assistant Professor
Department of EEE
BUET, Dhaka-1000.

Member

4.

Dr. M. M. Shahidul Hassan
Professor and Head
Department of EEE
BUET, Dhaka-1000.

Member
(Ex-Officio)

5. M. Ali Asgar

Dr. M. Ali Asgar
Professor
Department of Physics
BUET, Dhaka-1000.

Member
(External)

CANDIDATE'S DECLARATION

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Signature of the candidate

07/05/2002

(Md. Moniruzzaman)

TABLE OF CONTENTS

Acknowledgments

I would like to thank my supervisor of the thesis and teacher Dr. Md. Shafiqul Islam who has directed all of these works. I thank him from the bottom of my heart for his guidance, support and enhancement. Without his contribution I could never be successful in completing the research. I have found him beside whenever I faced difficulties and he always help me in furnishing with valuable information and advice, when I was in problem.

I also thank all of my friends and well-wishers whose support and inspiration kept me going through this thesis.

Above all, I am grateful to Almighty Allah who has blessed me with consciousness and all the ability to complete the thesis.

Abstract

The objective of this research is to develop a non-linear current-voltage (*I-V*) characteristics model for sub-micron Gallium Arsenide Metal Semiconductor Field Effect Transistors (GaAs MESFETs). For simulation purposes, data is generated on a PC by an improved algorithm and empirical constants are estimated by computing Mean Square Error (MSE) values from the experimental and simulated characteristics. The developed model was compared with the Kacprzak-Materka model [1]. For this comparison, an improved mean square error (MSE) technique was employed instead of root mean square (RMS) technique. In this respect, the Kacprzak-Materka model [1] was modified incorporating the effect of drain bias, V_{DS} and the effect of gate bias, V_{GS} on output conductance, g_d in the saturation region, which is suitable for nonlinear small signal circuit design.

As the gate lengths of GaAs MESFETs enter into the submicron regime, short channel effects occur. One of the most prominent short-channel effects is threshold voltage shift (ΔV_T). The value of ΔV_T is found to be directly proportional to active channel thickness (*W*) and inversely proportional to gate length (L_G). The values of ΔV_T for submicron MESFETs was included in the developed model.

The model that was proposed here needs six parameters; two of them were acquired from simple terminal measurement and other four parameters were extracted by the global curve-fitting technique of a measured family of drain current-voltage characteristics. A computer algorithm was developed to determine the empirical constants of the model. An improved MSE technique was employed to determine the accuracy of the model. Accuracy of the developed model was verified for different dimensions of MESFETs in the sub-micrometer regime. The model can easily be implemented in programs of computer aided analysis and design of circuit with sub-micron GaAs MESFETs.

LIST OF SYMBOLS

Symbol	Used to designate	Unit or value
L_G	Typical gate length	μm
V_{BO}	Built in potential	V
V_{GS}	Gate-to-source voltage	V
V_{DS}	Drain-to-source voltage	V
d	Depth of the depletion	μm
W	Total depth of the N region	μm
N	Channel doping density	cm^{-3}
I_{DS}	Drain current	mA/mm
I_{CH}	Channel current	mA/mm
R_s	Bulk resistance of the N-layer in the source-gate region	Ω
R_d	Bulk resistance of the N-layer in the gate-drain region	Ω
C_{gd}	Gate to drain charge storage capacitance	F
v_{sat}	Saturation velocity of electron	cm/s
E_s	Critical electric field of saturation	V/cm
ϕ_B	Schottky barrier height	V
ϕ_S	Surface potential	V
g_d	Output conductance	mS/mm
g_m	Transconductance	mS/mm
g_{mo}	Intrinsic transconductance	mS/mm
I_{DSS}	Saturation current at $V_{GS} = 0\ V$	mA/mm
V_T	Threshold voltage	V
ΔV_T	Shift in threshold voltage	V
q	Electronic charge	$1.6 \times 10^{-19}\ C$
ε_s	Permitivity of GaAs	$13.2 \times 8.85 \times 10^{-12}\ F/m$
τ	Time delay in the transconductance	s
μ	Drift mobility	$cm^2 / V\text{-}s$
$\alpha, \beta, \gamma, \delta$	Empirical constants.	

LIST OF FIGURES

Number *Page*

LIST OF TABLES

Number *Page*

CHAPTER I

Chapter One

INTRODUCTION

The history of semiconductor switching devices begins with the bipolar junction transistor (BJT), which is a combination of two *P-N* junctions. When a certain voltage which is equal to the potential barrier is applied across its base-emitter junction keeping the collector-base junction reverse biased, current starts to flow either from emitter to collector (for *P-N-P* transistor) or from collector to emitter (for *N-P-N* transistor), driving the devices *ON* state. By controlling the base current, it can control the amplified output. Thus BJT works both as a switching device and as an amplifier. It may be mentioned that, BJT is a current driven device. Its switching speed is not very fast also the amplification is not very high.

As science advances, design and performance of transistor is also advanced. Instead of current driven transistors, voltage driven transistors are introduced for switching purposes. They are fast, work in higher frequencies and amplify more. These are commonly called FETs (Field Effect Transistors). Instead of base, emitter and collector, they contain gate, source and drain along with a substrate. The device is controlled by gate voltage.

Since its invention, the field effect transistors (FETs) became the superior rival to its counterparts, the bipolar junction transistors (BJTs). Some of the advantages of FET over BJT are as follows:

- ❖ Exhibits a higher input impedance (typically many mega ohms), since the control (or gate) voltage is applied to a reverse-biased junction or Schottky barrier or across an insulator.

- ❖ Simpler to fabricate and occupies less space in ICs, hence it is specially suitable for integration of many devices on a single chip.

- Well suited for controlled switching between ON state and OFF state and is therefore useful in digital circuits.

- Less noisy.

Although the speed of the present BJT is faster than the FET devices, this situation is likely to change in the future. As we know that the speed of the bipolar device depends on its base width, which is at its minimum level. So further increase of speed of bipolar devices cannot be achieved. But the speed of the FET device depends on its gate length and a very reduced gate length (less than 0.1 μm) can be achieved for this device. As a result FET devices with a speed faster than bipolar devices can be realized in practice.

Among various types of FET (JFET, MOSFET, MESFET), MESFET is used extensively in communication and amplifying purposes.

1.1 Basic Difference Between Si-FET and GaAs FET

In principle, the physics of Silicon junction FET is very similar to GaAs (Gallium Arsenide) FET. Some of the differences are that in GaAs one usually deals with Schottky-barrier junction instead of p-n junction, and also in GaAs the conducting channel is confined on one side by a space-charge region and on the other side by a semi-insulating region. In Silicon the channel is usually, but not always, constricted from both sides by space charge regions formed around the gate p-n junctions. Moreover, in GaAs the electron velocity saturates near rather low electric field of *3 to 4 kV/cm* whereas Silicon obeys ohmic behavior over a range approximately *10 kV/cm*. Thus in GaAs the saturation of drain current with increasing drain-to-source voltage is caused by carrier-velocity saturation, whereas in Silicon it is channel pinch-off that causes the drain current to saturate.

1.2 Metal-Semiconductor Field Effect Transistor

MESFET is derived from the structure of the gate electrode, which is a metal semiconductor Schottky barrier contact. Thus GaAs MESFET stands for **Gallium** Arsenide **ME**tal-**S**emiconductor **F**ield **E**ffect **T**ransistor. MESFET consists of a conducting channel positioned between the source and the drain contact region as shown in Fig. 1.1. The Schottky metal gate controls the carriers that flow from source to drain. The control of the channel is obtained by varying the depletion layer width underneath the metal contact which modulates the thickness of the conducting channel and thereby the current.

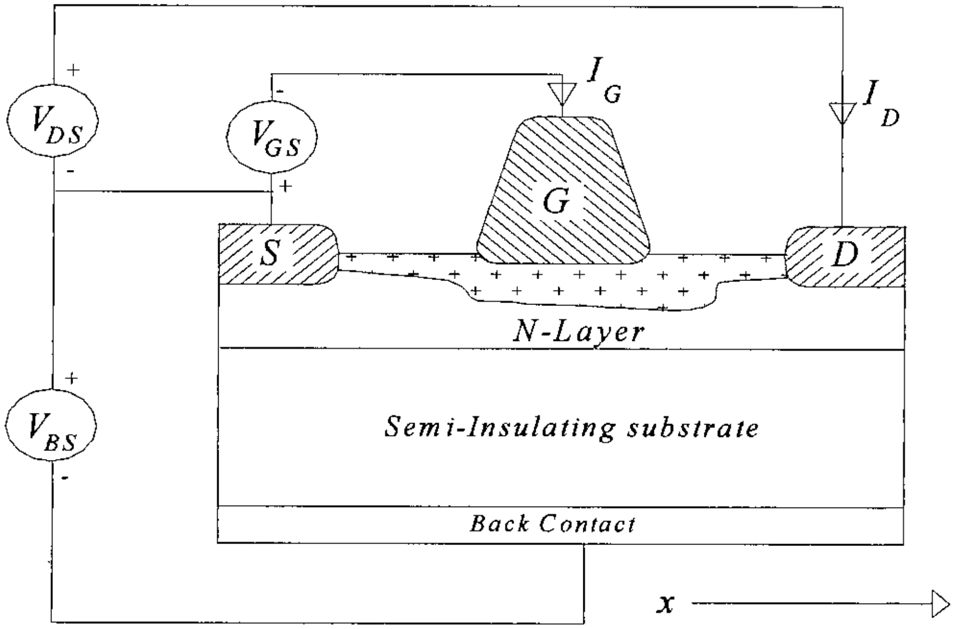


Figure 1.1: Cross-sectional view of MESFET.

1.3 Advantages of MESFET

The key advantage of MESFET is the higher mobility of the carriers in the channel as compared to the MOSFET. Since the carriers located in the inversion layer of a MOSFET have a wave function, which extends into the oxide, their mobility - also referred to as surface mobility - is less than half of the mobility of bulk material. As the depletion region separates the carriers from the surface in MESFET, their mobility is close to that of bulk material. The higher mobility leads to a higher current, transconductance and transit frequency for the device. The higher transit frequency of the MESFET makes it particularly of interest for microwave circuits.

Moreover, GaAs MESFET has large energy band gap (1.42 eV). Hence it can withstand higher operating temperatures, have lower thermal generation and have lower leakage currents than their Si counter-parts. Finally, GaAs MESFETs are much more insensitive to cosmic radiation than Si-MOSFETs, which makes them attractive for use in microwave satellite communication systems.

However, low defect density GaAs wafers are difficult to manufacture, which causes problems for device designers.

1.4 Applications of MESFETs

MESFET shows better performance in-

- **Wireless communication application:** Since the gate length is very short ($L_G \leq 1$ µm), in the radio frequency range; specifically, base station cellular applications, satellite and radar systems and personal communication systems and networks for infra-facility use.

- **Power switching application:** Specifically, high power specialty industrial ICs, high power switching for lighting and switch mode power supplies for PCs and other consumer applications.

- **Amplification and Switching application:** GaAs MESFET performs the basic functions of amplification and switching as required for low-noise amplifiers, broadband amplifiers, power amplifiers and microwave switches.

1.5 Organization of the Thesis

The thesis is organized into six chapters. The advantages of GaAs technology over Silicon are presented in chapter 1. Advantages of MESFETs and their applications are also presented in this introductory chapter.

Chapter 2 provides the mechanism of current saturation and the current-voltage characteristics of GaAs MESFET. It also describes the mechanism of current conduction through MESFET with respect to various bias effects and also describes the effect of substrate current.

Chapter 3 discusses the previous works in modeling of *I-V* characteristics of GaAs MESFET. Comparison among the models has also been done in this chapter.

Chapter 4 describes the steps of improvement of Materka model and the validity of its various steps comparing with the published experimental results.

Chapter 5 discusses the algorithms and flow-chart of various proposed step of improvement of Materka model for extracting the empirical parameters needed.

The last chapter of this thesis contains discussions about the range of empirical constants, range of drain-source voltage and conclusions of the research with a new reduced time algorithm. This chapter also contains suggestions for future research.

CHAPTER II

Chapter Two

OPERATION OF GaAs MESFET

2.1 Mechanism of Current Saturation

A schematic diagram of a GaAs MESFET in cross-section is given in Fig. 2.1. As MESFETs are normally used to amplify or switch signals in the microwave frequency range (several GHz upwards), the gate length has to be short: $L_G \leq 1\ \mu m$ is required. The gate itself is a rectifying metal-semiconductor contact in the form of an Schottky barrier.

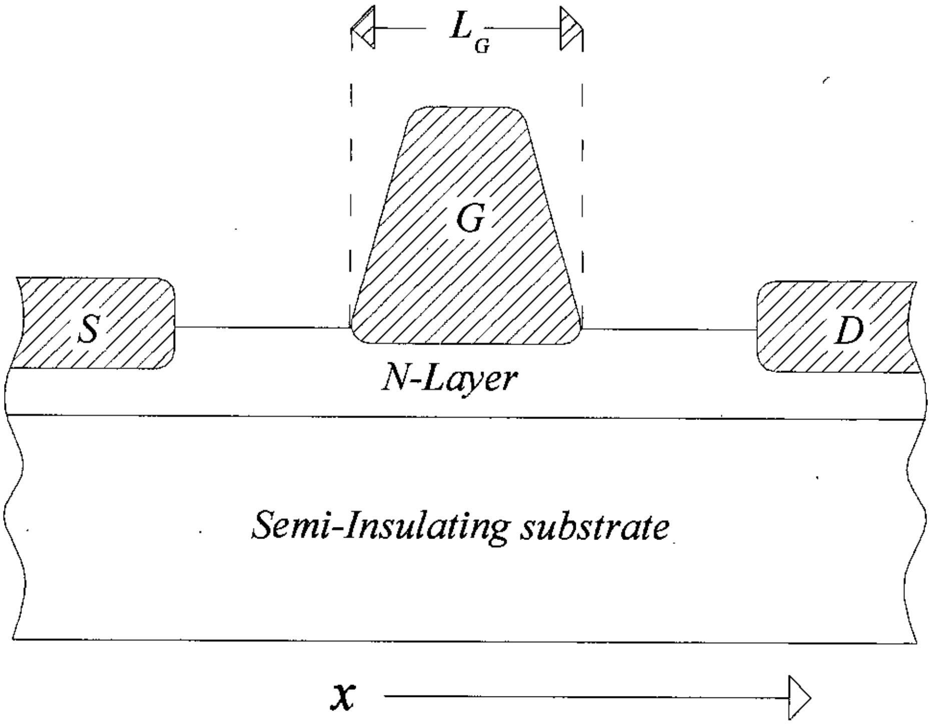
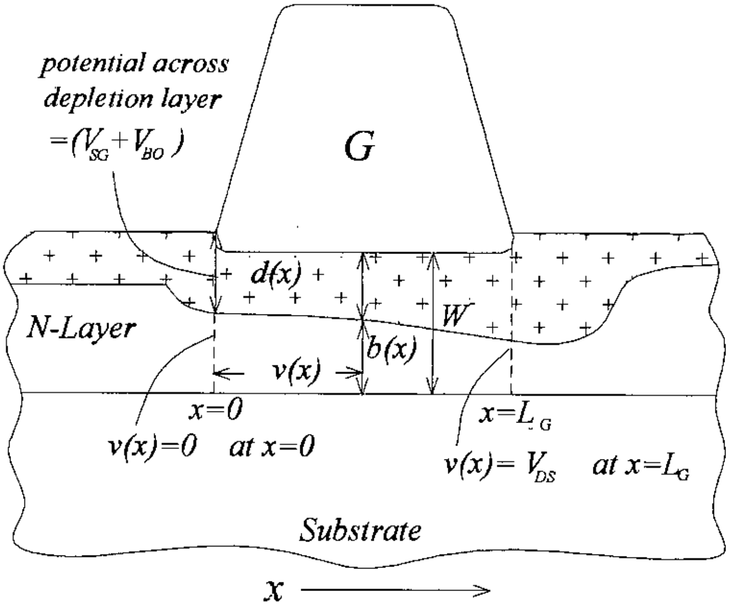


Figure 2.1: Schematic diagram of GaAs MESFET with short gate for microwave operation [2].

There is an *N*-layer, typically doped in the range of $10^{17}\ cm^{-3}$ to $10^{18}\ cm^{-3}$, with a thickness under the gate of *0.05* to *0.3 micron*. Here it is assumed that this layer is to be uniformly doped, grown by epitaxy rather than diffused. Neglecting any depletion at the junction of the *N*-layer with the semi-insulating substrate, it may be assumed that this interface to be a "hard wall". Metallized with aluminum the *N*-layer forms the gate contact. The source (*S*) and the drain (*D*) are formed by diffusing N^+ wells, which when metallized, form ohmic contact to the N^+-layer.

In normal operating condition a voltage is applied between the source and gate with the gate negative with respect to the source. Because the junction itself is a rectifying metal semiconductor contact in the form of an Schottky barrier, almost all the junction potential is dropped across the depletion layer, which is formed in the *N*-layer as shown in Fig. 2.2. For the doping concentrations indicated, this amounts to approximately *0.8 V*. This "*built in potential*" here will be written as V_{BO}. If the source and drain are connected together and grounded, and a negative voltage is applied to the gate, say $V_{GS} = -1\ V$, the depletion layer expands. The total potential change in going from the top of the depletion layer to the bottom, that is from the gate to *N*-layer interface to the edge at the depth *d* in Fig. 2.2, is now



potential across depletion layer =(V_{SG}+V_{BO})
G
d(x)
N-Layer
W
b(x)
v(x)
x=0
v(x)=0 at x=0
x=L_G
v(x)= V_{DS} at x=L_G
Substrate
x

Figure 2.2: GaAs MESFET in cross section showing approximate shape of depletion region with normal operating bias applied [2].

$V_{BO} - V_{GS} = 0.8\ V - (-1)\ V = 1.8\ V$

To save carrying unnecessary minus signs, the voltage applied externally between the source and the gate is specified by V_{SG}, i.e. the potential of the source relative to the gate. For the example at hand, $V_{SG} = +\ 1\ V$, then voltage supported by the depletion layer is now written as

$V_{BO} + V_{SG} = 0.8\ V + 1\ V = 1.8\ V$

By integrating Poisson's equation twice, the potential across the depletion is given in terms of the materials properties and the depletion depth d by the equation

$$V_{across\ depletion} = \frac{q \cdot N \cdot d^2}{2\,\varepsilon_S} \tag{2.1}$$

Where ε_s is the permitivity of GaAs ($\varepsilon_s = 13\ \varepsilon_o$). In the present case,

$V_{across\ depletion} = (V_{BO} + V_{SG})$

If a large enough external voltage is applied to deplete the *N*-channel layer fully, so $d = W$, the channel is said to be "*pinched-off*". The value of V_{SG} required to bring about pinch-off is called the *pinch-off voltage*, or the *threshold voltage*. Thus

$$V_T = \frac{q \cdot N \cdot W^2}{2\,\varepsilon_S} - V_{BO} \tag{2.2}$$

Table 2.1: Example data for GaAs MESFET [2]

$N = 3\times10^{17}\,cm^{-3}$	$L_G = 1\ \mu m$	$\mu_0 = 2500\ cm^2 / V\text{-}s$	$V_{BO} = 0.8\ V$
$W = 0.15\ \mu m$	$Z_G = 300\ \mu m$	$v_{sat} = 10^7\ cm\ s^{-1}$	$V_T = 3.9\ V$

For the device under consideration the values of its parameter are listed in Table 2.1.

And so

$$V_T = \frac{1.6\times10^{-19}C\cdot3\times10^{23}m^{-3}\cdot(0.15\times10^{-6})^2m^2}{2\times1.15^{-10}C\,V^{-1}\,m^{-1}} - 0.8V$$

i.e. $V_T = 3.9\ V$

When the threshold voltage is applied between the gate and the source, there is no conducting path any longer between the source and the drain, and so it is expected no current to flow between S and D in this limit; the whole of the conducting channel has disappeared under the entire length of the gate.

In normal operation, a voltage is applied between the source and drain, with the drain positive with respect to the source; an x-directed or longitudinal electric field is created in the channel. Except when the channel is pinched-off, a current of electron then passes down the N-layer from the source, S to the drain, D. In the device physicist's terms, the electrons drift in the longitudinal (i.e. x-directed) electric field that the drain-source voltage V_{DS} establishes.

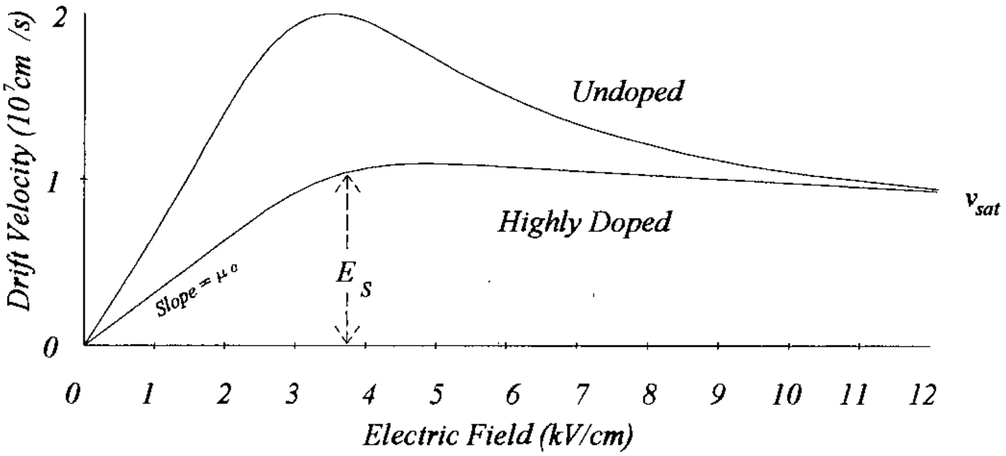


Figure 2.3: Drift velocity versus electric field for undoped GaAs, contrasted with case for typically highly doped MESFET layer [2].

The drift velocity versus electric field curve for GaAs is shown in Fig. 2.3. When an electric field E is applied to the semiconductor, the electrons experience a force $F = - q\ E$, and so acquire a component of momentum in the field direction (actually exactly anti-parallel to the field as expressed by the minus sign). Electrons however undergo collisions with the crystal lattice of GaAs atoms, which are continually in motion due to the thermal energy they possess.

The displacement of lattice atoms about their mean positions occurs in a wave-like manner with quantized states, giving rise to lattice waves referred to as phonons. The result of an electron-phonon collision generally is that the component of momentum in the field direction that the electron has acquired is partly destroyed; the momentum tends to be randomized into arbitrary directions. So the electron velocity can be, or will be, somewhat larger than component of velocity along the field direction.

The drift velocity, v is obtained by finding the average, over all mobile electrons, of the velocity component along the field direction. Referring to Fig. 2.3, at relative low values of field E, the average component of the electron's velocity along the field direction increases proportionally to E. The drift mobility, μ, is defined as the constant of proportionality, thus

$$v = \mu \cdot E \qquad (2.3)$$

At high enough field strength, $E \geq 4\ kV\ cm^{-1}$ in GaAs, the drift velocity saturates at a value of about $10^{7}\ cm\ s^{-1}$ for highly doped GaAs and $2 \times 10^{7}\ cm\ s^{-1}$ for undoped GaAs. Here it is taken that $E_S \approx 4\ kV\ cm^{-1}$ as the boundary point between the constant drift mobility regime and the saturated drift velocity regime.

Now considering the probability that the electric field strength in the x - direction in the MESFET channel of Fig. 2.1 could become large enough to cause the electrons moving down the channel to reach v_{sat}. Assuming somewhat simplistically that the field under the gate is uniform along x, the voltage drop in moving from the left end of the gate to the right end, a distance typically $1\ \mu m$, would be

$$V_{DS} = \int_{0}^{L_G} E_x dx \approx E_x L_G$$

Since E_x must have reached the value E_s for the electrons to travel at the saturated drift velocity, the drain-source voltage required is

$$V_{DS} \geq 4\ kV\ cm^{-1} \times 1 \times 10^{-4}\ cm = 0.4\ V$$

Thus, if the transistor ever operated only at a drain voltage less than $0.4\ V$, the electrons in a GaAs MESFET always move in the constant mobility regime in Fig. 2.3.

With the above current saturation mechanism, low-level derivation of Current-Voltage (I_{DS} *versus* V_{DS}) Characteristic of the GaAs MESFET can be derived. It is found that [2]

$$I_{DS} = \left(qN\mu_o W \frac{Z_G}{L_G} \right) \cdot \left[V_{DS} - \frac{2}{3}(V_T + V_{BO}) \cdot \left\{ \left(\frac{V_{BO} + V_{SG} + V_{DS}}{V_T + V_{BO}} \right)^{3/2} - \left(\frac{V_{BO} + V_{SG}}{V_T + V_{BO}} \right)^{3/2} \right\} \right] \tag{2.4}$$

This is the equation, which defines the GaAs MESFET relationship between drain current, I_{DS}, drain-to-source voltage, V_{DS}, and source-to-gate voltage V_{SG}. Its limit of validity is when the channel becomes pinched-off at the drain end of the gate.

Because the voltage across the depletion layer at $x = L_G$ is equal to ($V_{SG} + V_{DS}$), the range of validity of the equation (2.4) is

$$V_{SG} + V_{DS} \leq V_T$$

or

$$V_{SG} \leq V_T - V_{DS}$$

$$\leq 3.9\ V - 0.4\ V \qquad \text{i.e. } 3.5\ V$$

In another words, the range is specified by the conditions

$$V_{SG} \geq 3.5\ V \text{ for all } V_{DS}$$

$$V_{DS} \leq 0.4\ V \text{ for } V_{SG} < 3.5\ V$$

For $V_{DS} > 0.4\ V$ when $V_{SG} < 3.5\ V$, the current saturates due to electron drift velocity saturation in the channel. Because the undepleted part of the channel is tapered (Fig. 2.2), being slightly narrower at $x = L_G$, the electric field in the channel, E_x, is slightly larger at this point. Consequently, velocity saturation occurs first at the drain-end of the gate.

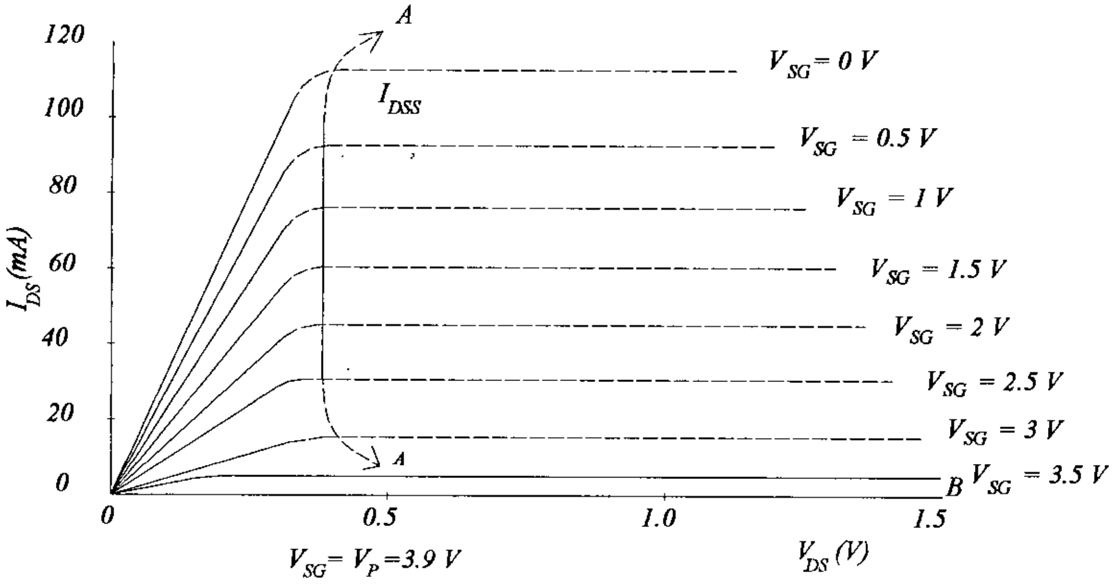


Figure 2.4: Calculated drain characteristics of short gate GaAs MESFET: A: region where current saturation is determined by velocity saturation: B: region where current saturation is governed by pinch-off at drain end of gate [2].

Equation (2.4) is plotted over its range of validity as the solid part of the curves in Fig. 2.4. Table 2.1 summarizes the data for the example MESFET. For each V_{SG}, the two regions above and below $V_{DS} = 0.4\ V$ have been connected a "hand smoothed" curve to give a reasonable transition.

2.2 Internal Mechanism of Current Conduction

The general shape of the space charge layer in a velocity saturated MESFET for normal biasing is shown in Fig. 2.5. Under the un-metallized source-gate and gate-drain regions, there are depletions to a depth W_S, such that $N\,W_S = D_S$, where D_S is the occupied surface-state density. Associated with this depletion is a band-bending giving rise to a surface potential ϕ_S as

$$\frac{\varphi_S}{q} = \frac{q \cdot N \cdot (W_S)^2}{2\,\varepsilon_S} = \frac{q \cdot (D_S)^2}{2\,\varepsilon_S \cdot N} \tag{2.5}$$

Where, as before, a uniform channel doping N has been assumed. It is important to note that D_S may be spatially non-uniform, with different values in the two regions.

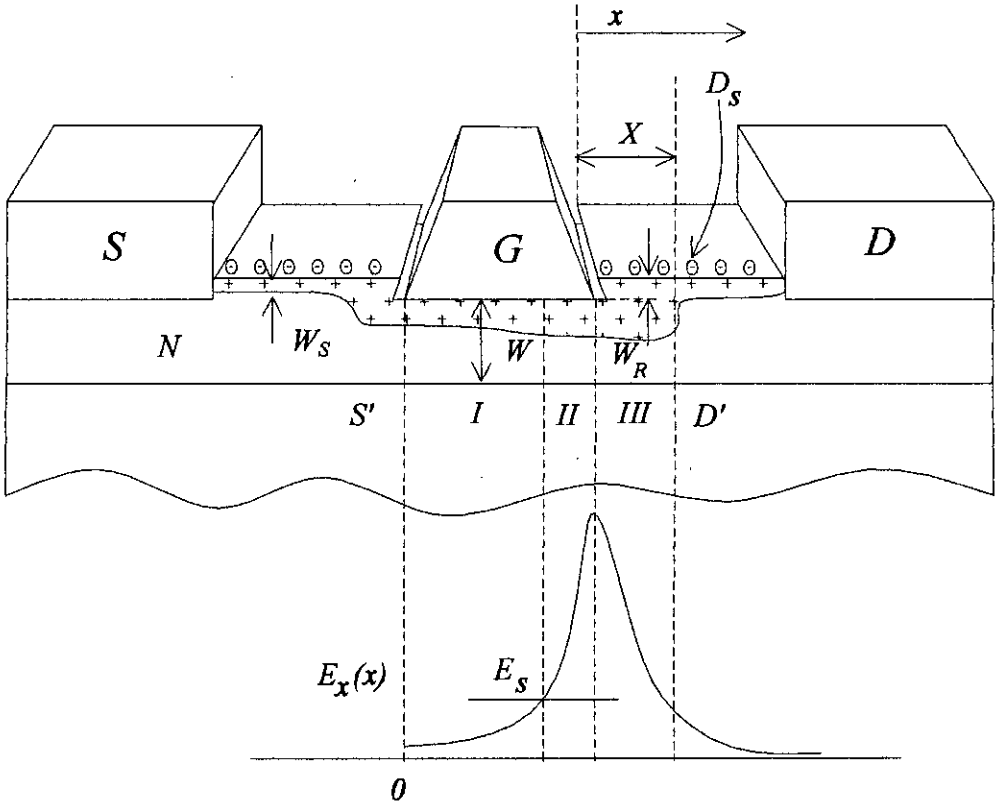


Figure 2.5: Depletion layer and channel field in an operating GaAs MESFET [2].

2.2.1 Effect of High Gate-Drain Bias

Under high gate-drain bias conditions, φ_S may rise to well above the equilibrium surface potential. The active channel region comprises the zones marked *I*, *II* and *III* in Fig. 2.5. In Fig. 2.5, *S'* and *D'* are referring to the virtual source and drain electrodes. The importance of these virtual electrodes is that their potentials differs from the applied potentials due to voltage drops across the source-gate and gate-drain regions where there is bulk resistance R_s and R_d respectively. It is primarily the potentials between these virtual electrodes ($V_{S'G}$ and $V_{D'S'}$), which govern the equivalent circuit elements of the intrinsic MESFET, and for that reason it is important that they may be distinguished from the applied voltages at the contacts, V_{SG} and V_{DS}.

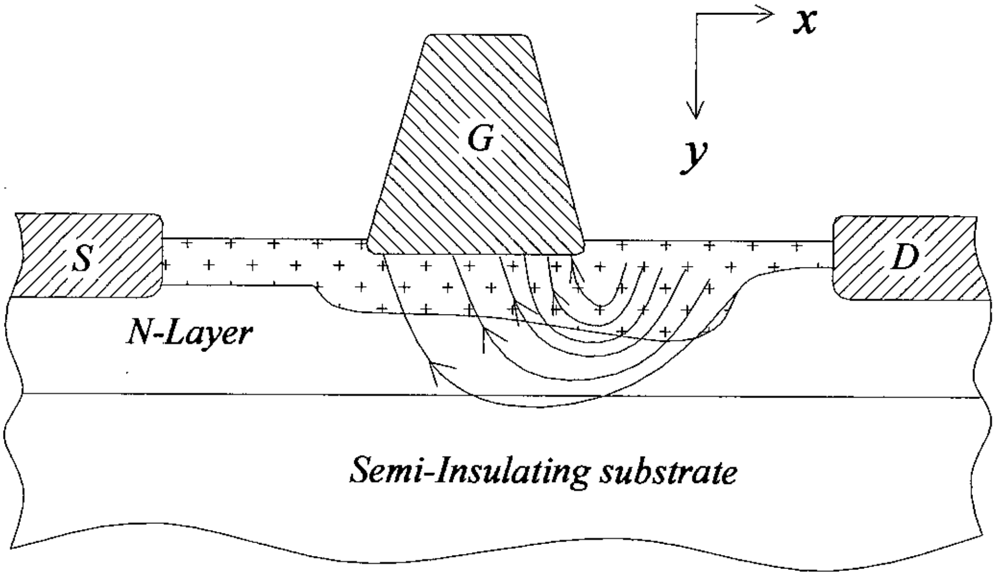


Figure 2.6: Illustrating y-directed fields in the channel of a MESFET [2].

In region *I*, the motion of the channel electrons in the longitudinal channel field is governed by the part of the velocity-field characteristics of Fig. 2.5, for which $0 < E \leq E_S$. Over this region, an increase in electric field strength results in an increase in electron drift velocity, i.e. the channel essentially behaves resistively. At the end of the region *I*, the electric field has reached the value E_S at which the electron drift velocity saturates and becomes independent of *E*. This condition persists up to the end of region *III*, at which point the *x*-directed channel field has dropped to values below

E_S once more. The distinction between regions *II* and *III* is that the first is in the shadow of the gate, whereas the second constitutes the space-charge layer extension X into the gate-drain space. Region *III* is fundamentally necessary to a satisfactory description of MESFET behavior, for it has a direct bearing on the values of gate-drain capacitance, C_{gd}, output conductance, g_d and time delay in the transconductance, τ. Finally, but importantly, electric field arising from ionic charge in region *III* gives raises to transverse fields in the region *I* and *II*, as illustrated in Fig. 2.6. Transverse fields in region *I* degrade the longitudinal mobility there.

2.2.2 Effect of Gate Voltage

If the gate voltage is increased in the negative sense while keeping the gate-drain voltage constant, the depletion layer changes as shown in Fig. 2.7. In region *I*

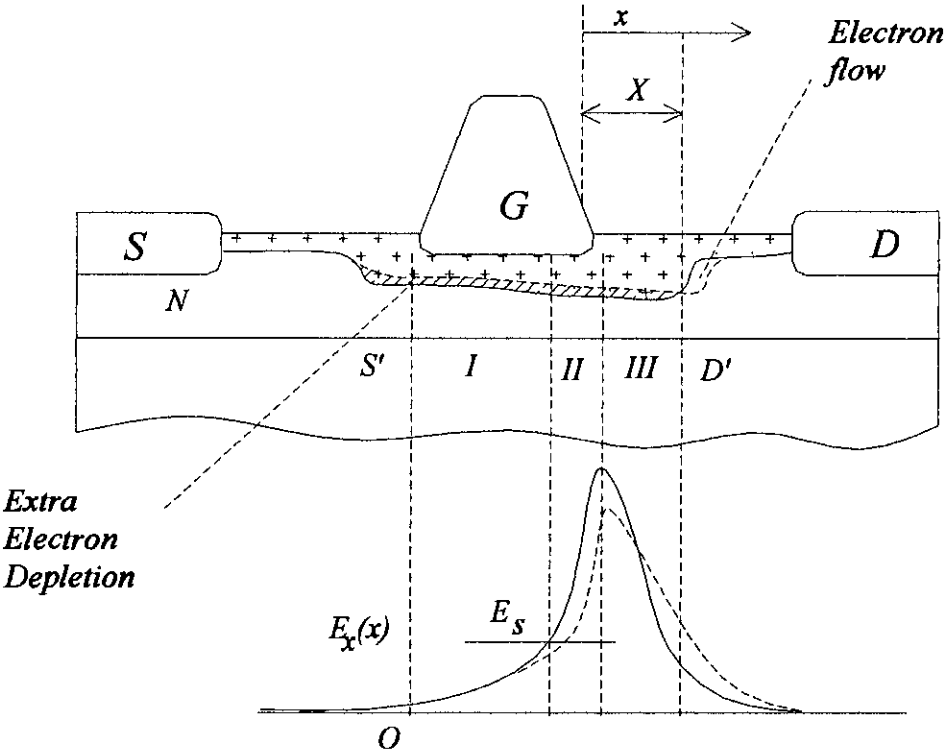


Figure 2.7: Showing effect of depletion layer of increasing negative bias on the gate while maintaining V_{DS} constant [2].

the depletion layer expands; the electrons which deplete from the shaded region are transported first through the resistive region, region *I*, then through region *II* and *III* and finally to the drain. Extra positive ionic charge is thus uncovered in the depletion edge in region *I* in response to the increase in gate voltage, i.e. the depletion layer exhibits charge storage properties which can be described in electrical terms by a capacitance.

Neglecting the effects of transverse fields for a moment, a linear extrapolation from low fields with a typical as-doped mobility of $3000\ cm^{2}/V\text{-}s$ intersects the value $v_{sat} = 10^{7}\ cm\ s^{-1}$ at a field of $3.3\ kV\ cm^{-1}$ (Fig. 2.8). So it can be written,

$$E_S \approx \frac{v_{sat}}{\mu_o}$$

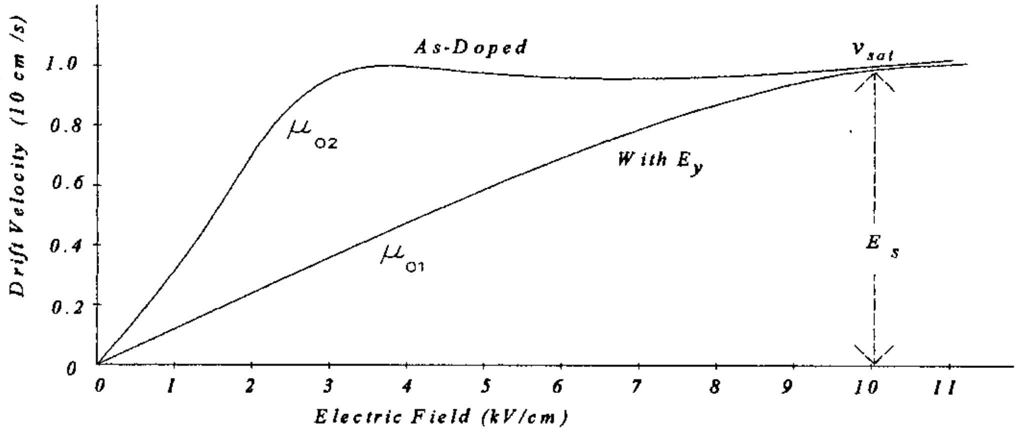


Figure 2.8: Effect of transverse fields in the channel on the effective velocity-field characteristic in the longitudinal direction [2].

Taking account of mobility degradation due to transverse fields in Fig. 2.8 and applying the rule of thumb, mobility μ_o is reduced by E_y to about one-third of its as-doped value, the previous estimate of E_S is modified to

$$E_S \approx \frac{3\,v_{sat}}{\mu_o}$$

2.2.3 Effect of Drain-to-source Voltage

If the gate-source voltage is held constant and the drain-source (or drain-gate voltage) is increased positively, the depletion extension *X* increases slightly, as in Fig. 2.9. Electrons deplete from the extreme edge of the space-charge layer to uncover more positive ionic charge, resulting in charge storage, which can be represented by a capacitance,

$$C_{gd} \approx \partial Q / \partial V_{D'G}$$

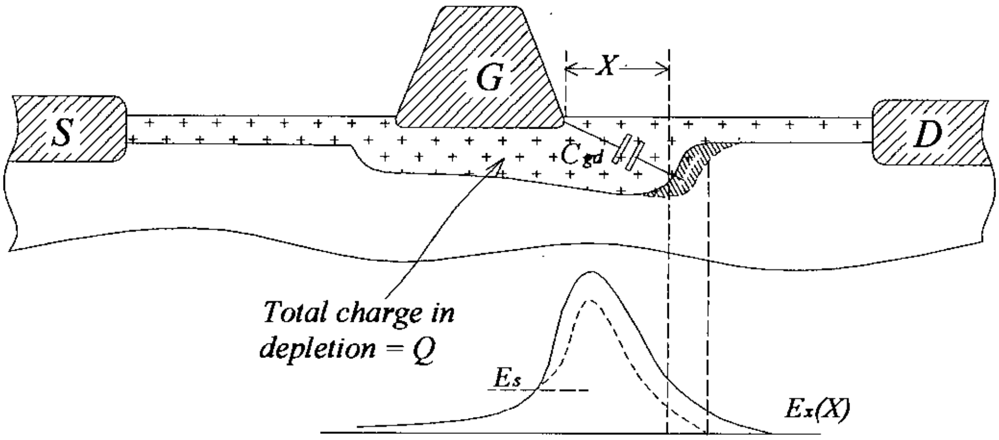


Figure 2.9: Illustrating increased extension of depletion layer for a positive increase in drain-source bias while maintaining the gate source bias constant [2].

2.3 Substrate Current and Its Effects

In Fig. 2.10 shows an operating FET in cross-section. There is a sharp rise in field in the channel across region *II* such that the peak field near the drain-end of the gate may reach several hundred $kV\ cm^{-1}$. (It is a result of solving the two-dimensional Poisson's equation in a velocity-saturated FET channel that the sharp rise in field occurs even without electron accumulation in the channel; quite simply, it comes about from field changing vector direction from being largely *x*-directed to *y*-directed, as shown in Fig. 2.10.) Electrons are heated by these intense fields, and so the electron occupancy of available energy states in the rising conduction band, towards the buffer or substrate, increases. In other words, the channel of current carrying electrons tends to widen and spread into the substrate (or buffer), giving rise to a component of current I_{SUBS} in the substrate. The electric field strength along the initial part of the path shown for I_{SUBS} in Fig. 2.10 is easily large enough to keep the electrons moving at the scattered-limited velocity, v_{sat}, so that I_{SUBS} may be worked out by calculating the sheet concentration of hot electrons injected into the substrate and multiplying by $q\ v_{sat}\ Z_G$. Such electrons are eventually collected by the drain together with the channel electron current I_{CH}. Thus,

$$I_{DS} = I_{CH} + I_{SUBS} = I_{CH}\left(1 + \frac{I_{SUBS}}{I_{CH}}\right) \qquad (2.6)$$

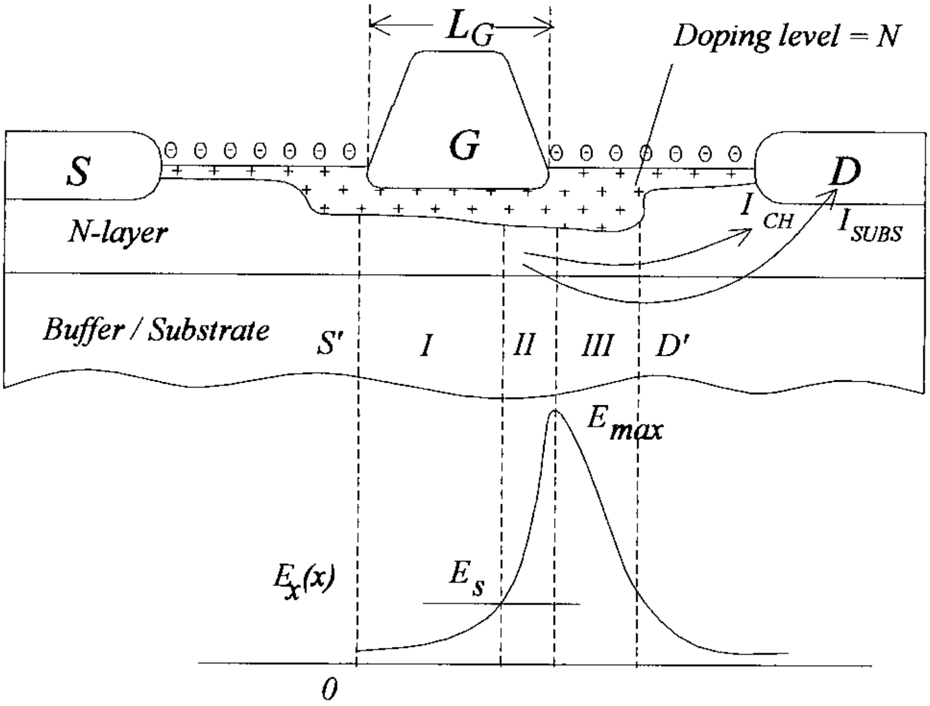


Figure 2.10: Current paths in layer and through buffer or substrate [2].

An approximate analytic theory for the ratio I_{SUBS} / I_{CH} yields the results shown in Fig. 2.11. The interfacial barrier of energy, which is effective in confining the electrons to the channel, is ϕ_B, which, corresponds to the energy rise in the potential barrier in one mean free path for a hot electron. ϕ_B may have any value up to the full height of the potential barrier itself, which is about *0.7 eV*.

Substrate current has a number of very interesting and very important consequences for MESFET behavior, both at dc and under microwave conditions:

(a) It may constitute several tens of percent of the total drain current.

(b) It is a physical mechanism which contributes to the formation of the knee in the I_{DS} (V_{DS}) characteristic at which current saturation sets in. (DC analyses of the GaAs MESFET based on channel transport alone yield knee voltages of typically several tenths of a volt, and assign the origin of the knee to the onset of velocity saturation in the channel. In sharp contrast, the present theory yields a knee voltage of typically *2 V* and assigns it to saturation of the numbers of electrons that are scattered into the substrate.),

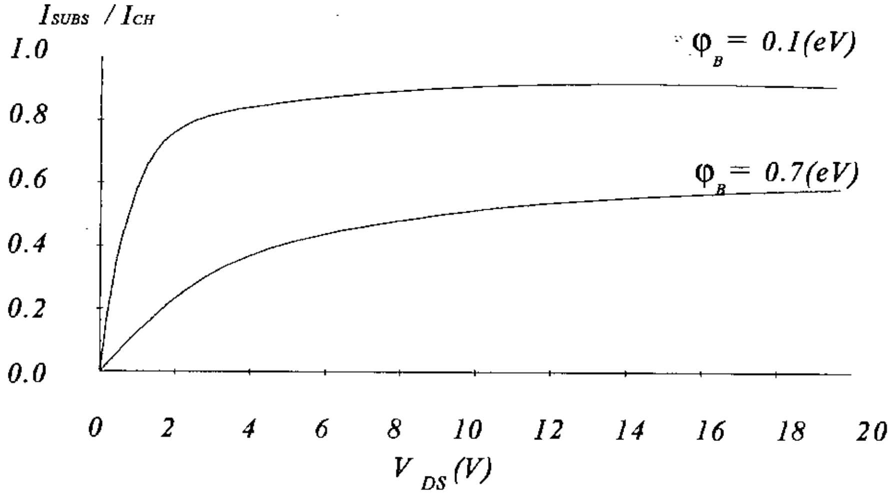


Figure 2.11: Calculated ratios of substrate current to channel current for 0.5 μm gate MESFET with shallow profile tail (ϕ_B = 0.1 eV) and steep profile tail (ϕ_B = 0.7 eV) [2].

(c) The slope of the total current I_{DS} with respect to V_{DS} defines the output conductance g_d as

$$g_d = \frac{\partial I_{DS}}{\partial V_{D'S'}}$$

Although analyses neglecting substrate current yield a finite value for g_d due to shortening of region *I* as $V_{D'S'}$ is increased, the magnitudes forecast are often far smaller than those observed in practice, i.e. values of output resistance r_d = (I/g_d) are calculated to be somewhat larger than those typically found in practice. Basing the calculation of g_d upon the substrate component of drain current yields values, which are reasonably well in keeping with practical values.

(d) The transconductance is strictly defined as

$$g_{mo} = \frac{\partial I_{DS}}{\partial V_{S'G}} = \left(\frac{\partial}{\partial V_{S'G}}\right)\left[I_{CH}\left(1+\frac{I_{SUBS}}{I_{CH}}\right)\right]$$

Providing $\partial I_{SUBS} / \partial V_{S'G}$ is not zero, g_{mo} will be changed by the presence of substrate current. In the present formalism (Fig. 2.10), the substrate current is taken to consist of hot electrons scattered into the substrate after heating by the high electric field in the channel. The carriers are scattered into the substrate at about the end of region *II* (i.e. underneath the drain-edge of the gate), in which event they have already undergone modulation by the gate field. In other words, substrate current is modulated, as well as the channel current, by the applied gate voltage or signal. Hence substrate current affects g_{mo}.

CHAPTER III

Chapter Three

MODELING OF GaAs MESFETs

3.1 Modeling and Its Importance

Device fabrication is a very complicated and expensive process. Once a device is designed and fabricated without modeling, it may not function properly as expected. So the total expenses of time and money will be in vain.

Representation of characteristics of a device by numerical expressions is said to be modeling. Usually devices with non-linear characteristics are modeled. The *I-V* characteristics of MESFETs are non-linear. The numerical model contains several parameters, which are dependent on the device geometry and internal configuration. If the generalized model is known for a certain device, then by changing the parameters, we can obtain the *I-V* characteristics as necessary. Now during fabrication the device geometry and doping density of different layers are maintained to have the same device parameters.

The software used to simulate circuit operation needs simple numerical model for non-linear devices. Beside the numerical model, certain empirical constants are also provided. These empirical constants differ from each other for different devices. Therefore simple and compact modeling is necessary for the ease of circuit simulation.

3.2 Modeling of MESFET I-V Characteristics

Now a days, in the high-speed and high frequency fields, the uses of Gallium Arsenide Metal Semiconductor Field Effect Transistors (GaAs MESFETs) are increasing. For this, researches on GaAs MESFETs are continuing to make it operate in higher frequencies with lower noise figures. In order to achieve a higher operating frequency, a short gate length is essential for MESFETs. However, several important so-called short-channel effects are observed in devices with short gate lengths. These effects limit the performances of the device by reducing transconductance (g_m) and by increasing drain-to-source current (I_{DS}) in the saturation region. It also results in a decrease in the feedback and input capacitances, and short transit time.

In device fabrication, one usually starts with a computer simulation of the device or circuit to be made. But it is difficult to model dc current-voltage characteristics of a GaAs MESFET. For the purpose of large signal computer analysis of GaAs integrated circuits microwave amplifiers and oscillators are difficult because these devices are extremely complex internally. Simple analytical model cannot accurately describe its behavior under all conditions. On the other hand, a detailed two-dimensional numerical models with the rigorous field dependent characteristics of carrier velocity in the channel, although more accurate, is not suitable for use with circuit analysis programs because of their complexity and numerous parameters. By means of numerical methods, the actual two-dimensional channel field distribution and space charge transition region can be calculated. Thus, while attention has to be paid to the identification of physical mechanisms for the understanding of short channel microwave MESFETs, a general accepted dc model is required. It should be simple and should predict device characteristics regardless of size or for a given device size the model should predict the characteristics for all variables, which can be changed during fabrication. Above all, the simulated characteristics should match the measured characteristics with a high degree of accuracy.

Criteria of a non-linear model required for the simulation of GaAs MESFET dc characteristics have been discussed in [3, 4]. According to Curtice [4] the criteria are as follows: -

a) Accurate approximation to the drain current control characteristics

b) Inclusion of transition-time effects

c) Accurate evaluation of gate capacitance

d) Evaluation of "non-electronic" drain-gate and source-gate capacitances

e) Evaluation of circuit parasitic

GaAs MESFET is a nonlinear device. Different researchers also did the modeling of this device before. The complex mathematical models are introduced at first, which are almost accurate. But in order to support simulation purpose, the necessity of compact but efficient modeling is felt. To do this, various numerical models were introduced; which were used to simulate the GaAs MESFET dc characteristics and also claimed to satisfy the required criteria [3, 4]. There are four commonly used models in circuit design. These are the Curtice model [4, 5], the Statz model [6], the Kacprzak-Materka model [1] for large signal simulation of GaAs MESFET devices and the Rodriguez model [7].

3.2.1 Walter R. Curtice Model

In Curtice model [4] transit time and other effects are included with the previously developed Shur model [8]. In 1978 Shur [8] proposed a simple analytical model for the simulation of GaAs MESFET. He has used Shockley's equations and the assumption that current saturation occurs due to the formation of stationary Gunn-domain at the drain side of the gate when the average electric field under the gate equals the domain sustaining field, E_s given as: v_{sat} / μ (v_{sat} is the saturation electron drift velocity, μ is the low field mobility). In this model electron transit time effects under the gate have been omitted.

The MESFET model proposed by Curtice [4] is slightly different than that of proposed by Shur [8], where transit time effects, τ, under the gate was included. He proposed a large-signal GaAs MESFET model, which consists primarily a non-linear voltage-controlled current source $I\ (V_{GS},\ V_{DS},\ \tau)$. He started with this large-signal model but different analytical dc expression for this voltage controlled current source.

The expressions used in SPICE 2 are derived from the FET model of Shichman and Hodges [9] and are (for $V_{DS} > 0$)

$$I\,(V_{GS}, V_{DS}) = \begin{cases} \mathit{0}, & V_{GS} - V_T < 0 \\ \beta \cdot (V_{GS} - V_T)^2 (1 + \lambda \cdot V_{DS}), & \qquad (3.1) \\ & 0 \le V_{GS} - V_T \le V_{DS} \\ \beta \cdot V_{DS} [2(V_{GS} - V_T) - V_{DS}] \cdot (1 + \lambda \cdot V_{DS}), & \qquad (3.2) \\ & 0 \le V_{DS} \le V_{GS} - V_T \end{cases}$$

Where,

β= Transconductance parameter,

λ = Simulates V_{DS} more effectively for the finite conductance,

Curtice proved that the use of the hyperbolic tangent function greatly improves the usefulness of the equation below saturation. The following analytical function is proposed for description of the current source in GaAs MESFET:

$$I\,(V_{GS}, V_{DS}) = \begin{cases} 0, & V_{GS} - V_T < 0 \\ \beta \cdot (V_{GS} - V_T)^2 (1 + \lambda \cdot V_{DS}) \tanh(\alpha \cdot V_{DS}), & \qquad (3.3) \\ \quad 0 \le V_{GS} - V_T \le V_{DS} & \\ \beta \cdot V_{DS} [2(V_{GS} - V_T) - V_{DS}] \cdot (1 + \lambda \cdot V_{DS}), & \qquad (3.4) \\ \quad 0 \le V_{DS} \le V_{GS} - V_T & \end{cases}$$

Where,

β= Transconductance parameter,

λ = Simulates V_{DS} more effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

3.2.2 Hermann Statz Model

Statz pointed out that the proposed analytical dc expression (equation 3.3) for the voltage controlled current source by Curtice [4] has a number of deficiencies. First of all, Curtice proposed that the drain current saturates at the same drain-to source voltage irrespective of the gate-to-source voltage. This is different from conventional models, because the critical field E_s in the channel is reached at approximately the same voltage $V_{DS} = E_s \times L_G$, where L_G is the channel or gate length. Statz proved that the dc expression given by equation (3.3) is a good representation of the current for a given V_{GS}. However, the behavior of I (V_{GS}, V_{DS}) as a function of V_{GS} is only poorly represented, especially if the pinch-off voltage of the transistor is large. Statz had shown that, except for V_{GS} near the pinch-off voltage, the saturated drain current I_{DS} is proportional to the height of the undepleted channel region near the source end. This is because the reduction in the channel height between the channel entrance and the point where the carrier velocity saturates is usually a negligible fraction of the height at the entrance. Thus the current may be approximately calculated by assuming that all carriers at the channel opening are moving at their saturated velocity. For constant channel doping, the saturated current I_{DS} should then vary approximately as

$$I_{DS} = v_{sat} \cdot Z_G \cdot \sqrt{(2 \cdot \varepsilon_s \cdot q \cdot N)}\left(\sqrt{(-V_T + V_{BO})} - \sqrt{(-V_{GS} + V_{BO})}\right) \qquad (3.5)$$

Where Z_G is the channel width, v_{sat} is the saturated electron velocity, ε_s is the dielectric constant, q is the electronic charge, N is the channel donor density, V_T is the threshold or pinch-off voltage, and V_{BO} is the built-in-potential of the gate junctions. Note that V_{GS} and V_T are normally negative. The first term in equation (3.5) is proportional to the height of the space-charge region at the threshold voltage, and thus is proportional to the thickness of the doped region under the gate. The second term in equation (3.5) is proportional to the height of the space charge region when the gate-to-source voltage V_{GS} is applied. Thus equation (3.5) is indeed proportional to the height of the undepleted channel.

Equation (3.5) is obtained by assuming that all carriers in the channel opening move at their saturated velocity. The approximation for the current in equation (3.5) breaks down when the voltage drop from the entrance of the channel to the point of velocity saturation of the carriers is comparable to the voltage difference V_{GS} - V_T. Under these conditions the assumption of constant channel height breaks down. In the limit of gate voltages near the pinch-off point, the proposed equation is different from equation (3.5) and is given by.

$$I_{DS} \cong \beta \cdot (V_{GS} - V_T)^2 \qquad (3.6)$$

Unfortunately, equation (3.6) is valid only near V_{GS} - V_T = *0*. Elsewhere the behavior is better described by equation (3.5).

To smoothly connect a law like equation (3.6) for small V_{GS} - V_T to an expression like equation (3.5) for larger V_{GS} - V_T, Statz chose the empirical expression

$$I_{DS} = \frac{\beta \cdot (V_{GS} - V_T)^2}{1 + b(V_{GS} - V_T)} \qquad (3.7)$$

For small values of V_{GS} - V_T, the expression is indeed quadratic while for larger values, I_{DS} becomes almost linear with V_{GS} - V_T. The value of b of the bare transistor is a measure of the doping profile extending into the insulating substrate and thus depends on the fabrication process. Negligence of b cannot be tolerated in most circuit simulations.

Statz also found that the *tanh* function in equation (3.3) consumes considerable computer time. He then further approximated the *tanh* function below saturation by a simple polynomial p of the form

$$p = 1 - \left(1 - \frac{\alpha \cdot V_{DS}}{n}\right)^n \qquad (3.8)$$

with n = 2 or 3.

In the saturated region ($V_{DS} > n/\alpha$), the *tanh* function is replaced by unity. Statz found consistently that the polynomial with $n = 3$ give the best fit. In summary Statz modified the previous model with the following dc equations:

$$I_{DS} = \frac{\beta \cdot (V_{GS} - V_T)^2}{1 + b(V_{GS} - V_T)} \cdot \left[1 - \left(1 - \frac{\alpha \cdot V_{DS}}{3}\right)^3\right] \cdot (1 + \lambda \cdot V_{DS})$$

For $0 < V_{DS} < 3 / \alpha$ (3.9)

$$I_{DS} = \frac{\beta \cdot (V_{GS} - V_T)^2}{1 + b(V_{GS} - V_T)} \cdot (1 + \lambda \cdot V_{DS})$$

For $V_{DS} \geq 3 / \alpha$ (3.10)

3.2.3 Kacprzak-Materka Model

It is found by two-dimensional analysis of a short channel GaAs MESFET [10] that the carrier drift velocity reduction and saturation due to electron upper valley scattering results in charge accumulation at the drain side of the channel, giving rise to a large field increase in this region. The Gunn effect enhances this field, which is interpreted as a formation of a stationary Gunn domain in the channel [11]. Theoretical analysis of this effect in a structure without a substrate showed [12] that three operation modes of dc characteristics exist: normal junction-gate FET operation (the so-called "pentode-like" operation), stable negative resistance in saturation, and Gunn oscillation due to the propagation of the high field domain. The operation mode depends on the doping density level of the conductive layer, the channel's geometry, length and thickness, and the bias condition.

On the other hand, dc current voltage characteristics measured for typical commercial GaAs MESFETs, e.g., 2SK138, 2SK273, 2SK279, 2N6680 and others, are stable for all bias conditions and do not exhibit a decrease of drain current in the saturation region. This discrepancy between theory and experiment may be understood by remembering that apparently gross assumptions of abrupt transition from channel to substrate and zero substrate conductivity are not valid for a non-ideal device. In a real transistor the active n-type epitaxial layer is grown on a semi-insulating substrate. Since the conductivity of the substrate is not zero, injection of electrons into the substrate takes place. If the channel is thin, the number of injected electrons is not negligible when compared to the number of electrons in the residual channel. Moreover, large drain voltage beyond saturation gives rise to holes generation by impact ionization within the high field domain and injection of these carriers into the substrate. The injected holes and electrons form the leakage current flowing in the substrate from source to drain and shunting the main path of drain current in the channel. As a result of this effect, it is understood that the increase of drain voltage beyond saturation gives rise to an increase in the leakage component of drain current, which compensates for the decreasing channel current due to the existence of the dipole domain and to a shift of the effective pinch-off potential V_T which is drain voltage V_{DS} dependent.

They showed that the measured I_{DS} - V_{DS} dc characteristics of an n-channel GaAs MESFET of type 2SK273 (Mitsubishi, with channel length 0.7 μm, and channel width 400 μm) are pentode-like. For the same device they also showed the threshold voltage shifts with the change of drain-to-source voltage V_{DS}. Hence Kacprzak-Materka proposed a model with the simple modification of pinch-off potential in the well known Taki formula [13] of pentode-like characteristics and is as follows:

$$I_{DS} = I_{DSS}\left(1 - \frac{V_{GS}}{V_T + \gamma \cdot V_{DS}}\right)^2 \tanh\left(\frac{\alpha \cdot V_{DS}}{V_{GS} - V_T - \gamma \cdot V_{DS}}\right) \qquad (3.11)$$

Where I_{DSS} is the drain saturation current at $V_{GS} = 0\ V$, the parameter α was not chosen to represent physical effects but to provide the best average fit to the experimental characteristics in the triode and pentode regions, γ simulates the effective threshold voltage displacement as a function of V_{DS} and V_T is the threshold voltage. Therefore the Kacprzak-Materka model is a *four* parameters dc GaAs MESFET model for non-linear circuit. The four parameters I_{DSS}, V_T, γ and α must be defined by a global curve-fitting technique with computer optimization program. The proposed dc model describes the current-voltage characteristics of a transistor over the drain-source voltage V_{DS} from zero to avalanche breakdown.

3.2.4 Rodriguez - Tellez Model

Rodriguez started with the model proposed by the Shichman and Hodges [9], which were described by the equations (3.1) and (3.2). He also mentioned that Curtice [4] proved that the use of the hyperbolic function greatly improves the usefulness of the equation below saturation (equation 3.3).

Rodriguez proved that though the accuracy of the Curtice model [4] was improved compared to Shichman and Hodges [9], but there was still a marked difference between the measured and the computed *I-V* characteristics of the device. As before optimization process reduced these errors but this effort was in many cases found to produce little improvement in relation to the CPU time. Interestingly enough, the accuracy of the model was found to be particularly poor when high λ devices were considered. It was also observed that the accuracy of the model deteriorated as the size of the device decreased. This was found to result from the fact that the maximum current handling capability of the device is determined to a large extent by its size, and the accuracy of the Curtice model decreases at low bias levels.

To determine the reasons for the poor accuracy of the model, Rodriguez et al. decided to investigate the bias dependency of the model parameters (β, V_T, λ, R_d and R_s). This study considered the device operation over widely varying bias conditions. Most of the model parameter values were found to vary with bias. In this respect, β and V_T showed the smallest variations with bias changes, whereas λ showed a much larger variation. R_d and R_s were found to be bias independent. After much experimentation, they found that the largest improvement in the accuracy of the Curtice model was achieved by modeling the bias dependency of V_T. Various expressions for simulating this bias dependency were tried against the measured response of several hundred devices of differing designs and sizes. The best compromise between accuracy, ease of parameter extraction and CPU time was achieved with the expression

$$I_{DS} = \beta \cdot (V_{GS} - V_T + \gamma \cdot V_{DS})^2 (1 + \lambda \cdot V_{DS}) \tanh(\alpha \cdot V_{DS}), \qquad (3.12)$$

$$0 < V_{GS} - V_T < V_{DS}$$

Where,

β = Transconductance parameter,

λ = Simulates V_{DS} more effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

γ = Bias dependency of the pinch-off voltage on V_{DS}.

3.3 Comparison among Four Models

Comparison of the above-described four models by Rodriguez et al. [14] shows that Kacprzak-Materka model [1] appears to be the most accurate both in linear region and in the saturation region.

Rodriguez et al. carried out experiment using measured and computed graphs for *6 × 150 μm* and *1 × 25 μm* gate-width devices. These devices utilize a *0.5 μm* gate-length and *-2 V* pinch-off processes. These were selected so that the effect of varying gate-length, gate-width, number of fingers and pinch-off voltage on model accuracy could be observed. Therefore the results presented were quite general and applicable to a wide range of device designs and processing conditions.

From the experiment Rodriguez et al. showed that the accuracy of the Curtice model is somewhat poor overall and it deteriorates considerably as the drain current is reduced. For each model, the computed results were achieved after extensive optimization with the Simplex algorithm [15]. Also, all the parameters defining I_{DS} were included in the optimization process. The initial estimation of the parameters of each model was performed with the procedures described by Rodriguez [7].

They also showed that the Statz model [6] is marginally better than the Curtice model [4] in the saturation region and under high current conditions. As with the Curtice model, the accuracy decreases in the saturation region as the drain current is reduced. In the linear region this model is slightly worse than the Curtice model. This is to be expected because of the replacement of the *tanh* $(\alpha\ V_{DS})$ term in the Curtice model with the $(1 - (1 - \alpha\ V_{DS} / 3))^{3}$ approximation in the Statz model. Though Statz model reduced the CPU time considerably over its rivals but it is a discontinuous model, which is more difficult to optimize against the measured characteristics of the device. Therefore, the Statz model does not appear to offer any appreciable advantage over the Curtice model.

They also showed that, the Materka model [1] offers real advantages in accuracy over the two previous models. In the saturation region, the improved accuracy extends from high to low drain currents. This arises from the inclusion of the γ parameter, which simulates the bias dependency of V_T. It is noticed that under high current conditions, the accuracy of the model worsens. This deterioration increases as the device size decreases. The improved accuracy over the two previous models in the linear region is also noticeable. This is due to the more complicated expression employed to simulate the linear region (the *tanh* term), which is now V_{GS} and V_{DS} dependent.

In the saturation region, the Rodriguez model [7] is significantly better than the Curtice or Statz models and comparable with the Materka model. The improvements in accuracy over Curtice or Statz model are most noticeable in the low current region. In the linear region, the accuracy of this model is worse than Materka model.

The advantage in the linear region was retained by the Materka model for all devices considered by Rodriguez et al. [14]. Generally speaking, when these four models are applied and optimized against data, which includes only the linear region, the Statz model produces the lowest accuracy. This is followed by the Curtice model, which is on average *40 %* more accurate than Statz. The Rodriguez model is slightly better, producing a *50 %* increase in accuracy over Statz. The Materka model produces the best results showing on average a *62 %* improvement in accuracy over Statz.

The performance of the Materka model worsens as the device size is decreased, as proved from the device of size *1 × 25 μm*. However, under low drain current conditions, the model accuracy is still quite good, but as the current increases the errors become more significant. The accuracy of the Curtice or Statz models was found to worsen as the device size decreased. They also proved that for the same device, applying the Rodriguez model results in an excellent agreement with the measured data.

When the four models above were applied to the *I-V* characteristics of the device acquired under *r.f.* conditions (i.e. g_m), the same results were observed. For the large device the Materka model showed the best agreement, closely followed by the Rodriguez model. This order then reversed when the process was applied to the small *1 × 25 μm* device. In all cases, the Curtice and the Statz models fared the worst with very little difference between the two.

CHAPTER IV

Chapter Four

MODEL DEVELOPMENT

4.1 Objectives

The current-voltage (*I-V*) characteristics curve of GaAs MESFET is non-linear in nature. To represent this non-linear characteristic, detailed two-dimensional numerical models with rigorous field dependent characteristics of carrier velocity in the channel have been designed. Clearly, a complex model requiring extensive process-related information will yield good accuracy but this information is rarely available to the everyday user. Such models, because of their complexity or large number of parameters, also place a heavy burden on circuit simulation time. For this reasons, the circuit designer often compromises the requirement to take into account the problems relating to the acquisition of the parameters of the model and the complexity of the model itself. So designing of a generally accepted simpler numerical model is necessary. It should be simple, compact and contain the minimum number of variables. It should also predict device characteristics regardless of size or for a given device size the model should predict the characteristics for all the variables, which can be changed during fabrication. Above all, the simulated characteristics should match the measured characteristics with a high degree of accuracy.

Several numerical models have been designed for this purpose such as Curtice model [4, 5], the Statz model [6], the Kacprzak-Materka model [1] for large signal simulation of GaAs FET devices and the Rodriguez model [7]. In every new numerical model, the degree of accuracy of the simulated characteristics comparing with the measured characteristics has increased, which have discussed in the previous chapter. The goal of this research was either to design a new such model or to modify any of the previous models so that the degree of accuracy could climb higher in the sub-micrometer regime.

Though the Kacprzak-Materka model [1] is much-improved edition of numerical model among the four models as discussed in the previous chapter considering the degree of accuracy, yet there is a room for improvement. *Hence in this research, the Kacprzak-Materka model [1] is taken to be the base model for further improvement.*

In the process of designing a more accurate numerical model for sub-micron GaAs MESFET, designing a computer algorithm is needed so that it can calculate the values of empirical constants of the device more precisely than the previously used computer algorithms. The previous algorithms, (for example the algorithm [15] used to calculate empirical constants for Rodriguez's model [7]) were not much accurate. To increase the degree of accuracy in finding the values of empirical constants, necessity of a new algorithm rose and the second goal of this research is to design a new algorithm to calculate empirical constants as accurately as possible. Hence in a nutshell, the two objectives of this research are as follows:

a) To modify the previous Kacprzak-Materka model for sub-micron MESFETs.

b) To design a new algorithm to calculate empirical constants.

4.2 First Step Improvement of Materka Model

Rodriguez et al. [14] showed that the advantages of the Materka model over Curtice or Statz increase as the device size increases. This is due to the fact that output conductance of the devices becoming progressively more negative as their size increases. On the other hand, when the device size decreases the output conductance of the devices becoming progressively more positive. For these devices the Materka model is worse than Curtice or Statz model.

One of the reasons of deficiencies is that, in Materka model the effects of substrate current as discussed in section 2.3 has been neglected. Practically, there is a direct relation of substrate current with the drain-to-source voltage V_{DS}.

Another reasons of deficiencies is that, in Materka model the effects of transverse fields in the gate region due to the electric field arising from the ionic charges undepleted by the extra drain-to-source voltage, V_{DS}; as discussed in section 2.2.3 has been ignored. If the gate-source voltage is kept constant and the drain-source voltage is increased positively, the depletion extension X increases slightly, as shown in Fig. 2.9. Electrons deplete from the extreme edge of the space-charge layer to uncover more positive ionic charge. The electric field lines originated by the positive ionic charges will have its maximum strength near the drain side of the gate electrode. It is assumed that negative charges are induced in the gate metal, because of these field lines, resulting in a reduction in the gate biasing. This decreases the gate depletion, which corresponds to greater active channel thickness under the gate and thereby increasing channel current, I_{CH} flowing from source to drain electrodes.

Hence from the above discussions, it is clear that there is a direct relation between the drain-to-source voltages with the drain-to-source current (both channel current and substrate current). But Kacprzak-Materka proposed their model with the simple modification of pinch-off potential with the drain-to-source voltage in the well-known Taki formula [13] of pentode-like characteristics. This deficiency of the Materka model could easily be overcome by multiplying the existing expression for I_{DS} with a $(I + \lambda\ V_{DS})$ term. This change would be valid for devices exhibiting a positive or negative output conductance. Hence the first step of improvement of Materka model is proposed as follows:

$$I_{DS} = I_{DSS}\left(1 - \frac{V_{GS}}{V_T + \gamma \cdot V_{DS}}\right)^2 \tanh\left(\frac{\alpha \cdot V_{DS}}{V_{GS} - V_T - \gamma \cdot V_{DS}}\right)\left(1 + \lambda \cdot V_{DS}\right) \qquad (4.1)$$

Where,

λ = Simulates V_{DS} effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

γ = Bias dependency of the pinch-off voltage on V_{DS}.

To find out the validity of the ***First Step Improvement of Materka (FSIM) model*** two different GaAs MESFET having different device dimensions are tested with both Kacprzak-Materka model and the new *FSIM* model to achieve the characteristics as shown in Figs. 4.1 and 4.2. In these figures, the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using Kacprzak-Materka model and solid lines show the simulated characteristics using *FSIM* model.

The magnitudes of transconductance, $g_m = \left(\dfrac{\partial I_{DS}}{\partial V_{GS}}\right)_{V_{DS}=const.}$ and output conductance, $g_d = \left(\dfrac{\partial I_{DS}}{\partial V_{DS}}\right)_{V_{GS}=const.}$ for Materka Model can be evaluated from Materka equation (3.11) and are expressed as follows:

Transconductance,

$$g_m = I_{DSS}\left[2\cdot\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\cdot\left(\frac{-1}{V_T+\gamma\cdot V_{DS}}\right)\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right.$$
$$\left.+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot(-V_{DS})}{(V_{GS}-V_T-\gamma\cdot V_{DS})^2}\right\}\right] \tag{4.2}$$

And output conductance,

$$g_d = I_{DSS}\left[2\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\frac{\gamma\cdot V_{GS}}{(V_T+\gamma\cdot V_{DS})^2}\cdot\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right.$$
$$\left.+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot(V_{GS}-V_T-\gamma\cdot V_{DS})+\alpha\cdot\gamma\cdot V_{DS}}{(V_{GS}-V_T-\gamma\cdot V_{DS})^2}\right\}\right] \tag{4.3}$$

The magnitudes of transconductance, g_m and output conductance, g_d for *FSIM* Model can be evaluated from *FSIM* equation (4.1) and are expressed as follows:

Transconductance,

$$= I_{DSS}\left[2\cdot\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\cdot\left(\frac{-1}{V_T+\gamma\cdot V_{DS}}\right)\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\left(1+\lambda\cdot V_{DS}\right)\right.$$
$$\left.+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot(-V_{DS})}{\left(V_{GS}-V_T-\gamma\cdot V_{DS}\right)^2}\right\}\cdot\left(1+\lambda\cdot V_{DS}\right)\right] \tag{4.4}$$

And output conductance,

$$g_d = I_{DSS}\left[2\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\frac{\gamma\cdot V_{GS}}{\left(V_T+\gamma\cdot V_{DS}\right)^2}\cdot\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\left(1+\lambda\cdot V_{DS}\right)\right.$$
$$+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot\left(V_{GS}-V_T-\gamma\cdot V_{DS}\right)+\alpha\cdot\gamma\cdot V_{DS}}{\left(V_{GS}-V_T-\gamma\cdot V_{DS}\right)^2}\right\}$$
$$\left.\cdot\left(1+\lambda\cdot V_{DS}\right)+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\lambda\right] \tag{4.5}$$

To find out the validity of *FSIM* model, the simulated and the observed transconductances, g_m are plotted against gate-to-source voltage (V_{GS}) both for Kacprzak-Materka model and the new *FSIM* model as shown in Figs. 4.3 and 4.4. In these figures, the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using Kacprzak-Materka model and solid lines show the simulated characteristics using *FSIM* model. Similarly, the simulated and the observed output conductance, g_d are plotted against drain-to-source voltage (V_{DS}) both for Kacprzak-Materka model and the new *FSIM* model as shown in Figs. 4.5 and 4.6.

From all of these plots it is observed that the simulated graphs and the observed characteristics show a good degree of agreement and the degree of accuracy for the *FSIM* model is higher than that of the Kacprzak-Materka model.

Table 4.1: Comparative data of the Kacprzak-Materka model and *FSIM* model.

Device	V_{GS}	Materka Model				FSIM Model			
		RMS Error	MSE	Avg. MSE	Avg. RMS Error	RMS Error	MSE	Avg. MSE	Avg. RMS Error
A-74-1	0.0 V	24.7499	27.0716	14.3157	9.6753	2.1609	3.7622	6.2136	0.2863
	-0.5 V	5.2928	11.6788			3.2194	5.9425		
	-1.0 V	4.8861	10.4625			4.2773	9.4571		
	-1.5 V	3.6949	6.5794			0.8174	5.8757		
	-2.0 V	1.4934	1.6364			4.425	4.4291		
A-64-2	0.0 V	9.0576	16.5041	12.4182	6.8718	3.4805	11.9543	8.9885	1.0148
	-0.7 V	14.2653	16.4652			8.1162	10.9881		
	-1.4 V	4.6825	7.4408			3.5378	5.6189		
	-2.1 V	3.3536	5.6385			1.4348	4.5999		
	-2.8 V	11.8075	11.8494			9.3023	9.3589		

Table 4.1 shows the measured and modeled output characteristics of the devices. The modified model shows good agreement with experimental characteristics both in low and high V_{GS} voltages. In device A-64-2, the maximum magnitude of the MSE is 11.85 for Kacprzak-Materka model at V_{GS} = -2.8 V, whereas its value is 9.36 for the proposed *FSIM* model. Device A-74-1 has relatively high conductance after saturation. Hence the Kacprzak-Materka model accuracy deteriorates, than the modified model especially in the non-linear region. At V_{GS} = 0 V the magnitude of the MSE for Materka model is 27.07, whereas for the same gate voltage it is only 3.76 for the FSIM model.

In general, it can be stated that with the proposed modification of the Kacprzak-Materka model it is possible to simulate a wide variety of GaAs MESFET devices with a reasonable accuracy.

I_{DSS} = 192 mA, V_T = - 2 V,

(FSIM Model) α = 3.2858, γ = - 0.0553, λ = 0.0807,

(Materka Model) α = 3.8387, γ = - 0.1614.

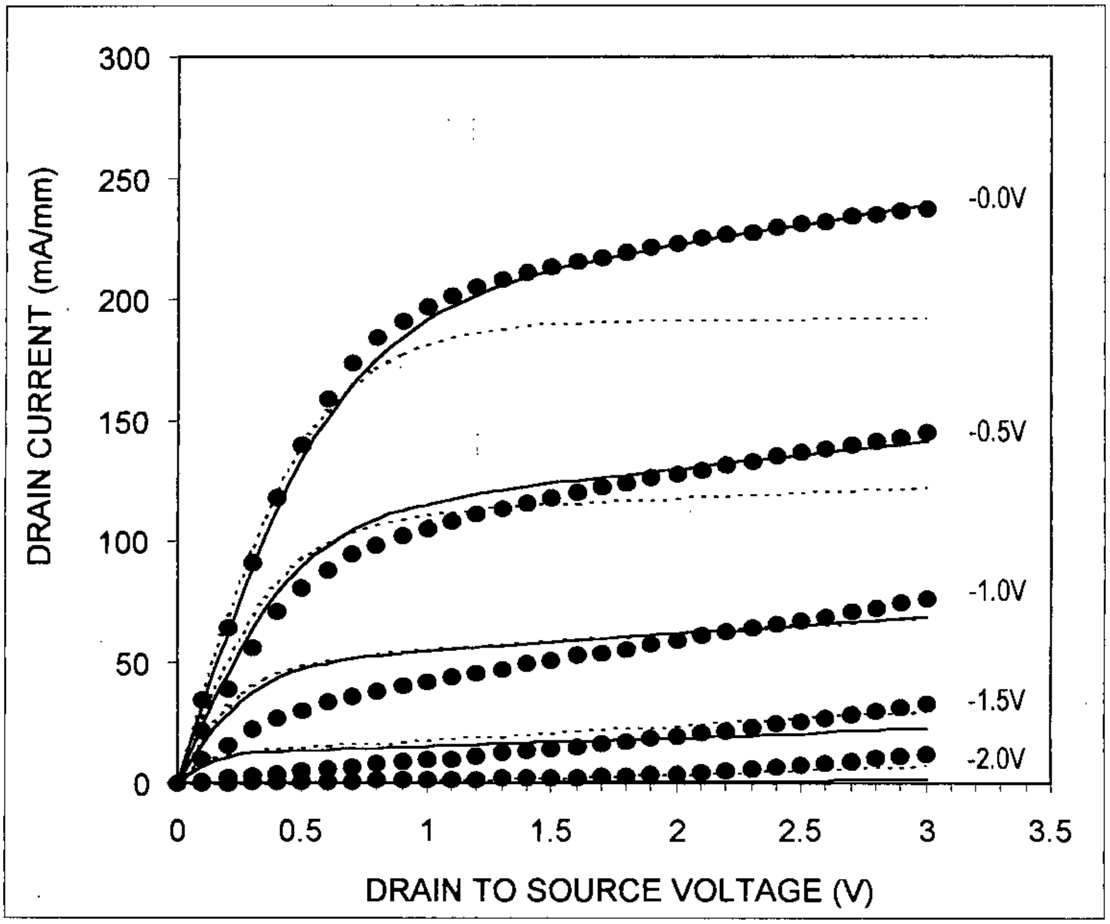


Figure 4.1: Observed and simulated output characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristics [19] and dash lines show the simulated characteristics using Kacprzak-Materka model and solid lines show the simulated characteristics using *FSIM* model.

$I_{DSS} = 350\ mA,\ V_T = -\ 2.8\ V,$

(*FSIM Model*) $\alpha = 3.5168,\ \gamma = -\ 0.3117,\ \lambda = 0.0302,$

(*Materka Model*) $\alpha = 3.8333,\ \gamma = -\ 0.3701.$

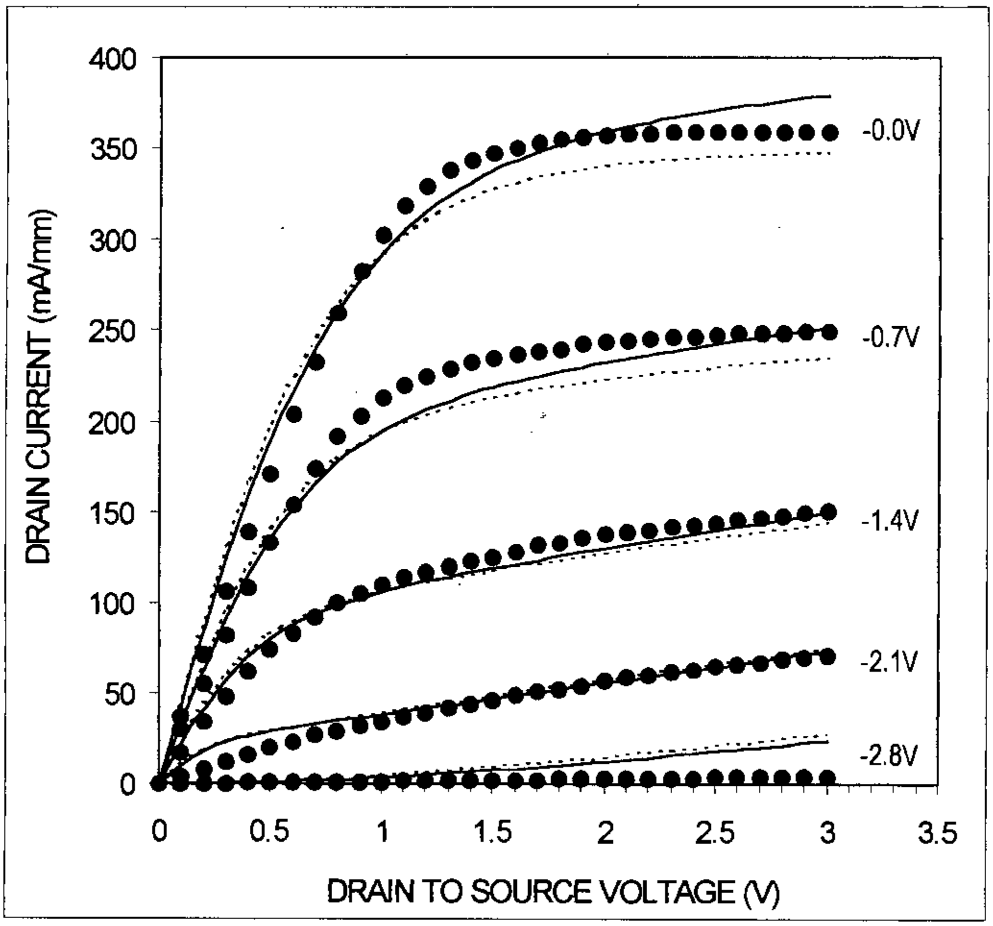


Figure 4.2: Observed and simulated output characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristics [18] and dash lines show the simulated characteristics using Kacprzak-Materka model and solid lines show the simulated characteristics using *FSIM* model.

$I_{DSS} = 192\ mA,\ V_T = -2\ V,\ V_{DS} = 2\ V,$

(FSIM Model) $\alpha = 3.2858,\ \gamma = -0.0553,\ \lambda = 0.0807,$

(Materka Model) $\alpha = 3.8387,\ \gamma = -0.1614.$

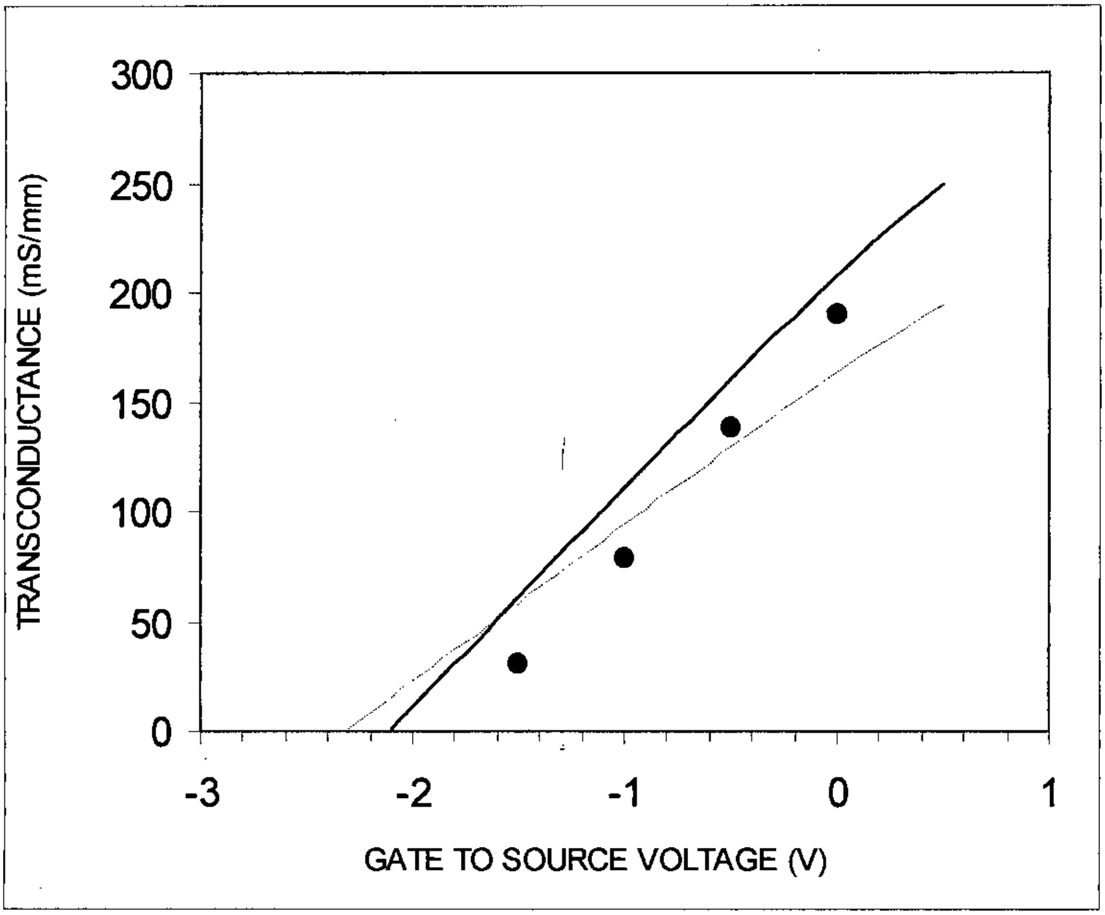


Figure 4.3: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured [19] characteristic and dash line shows the simulated characteristic using Kacprzak-Materka model and solid line shows the simulated characteristic using *FSIM* model.

$I_{DSS} = 350\ mA,\ V_T = -2.8\ V,\ V_{DS} = 2\ V,$

(FSIM Model) $\alpha = 3.5168,\ \gamma = -0.3117,\ \lambda = 0.0302,$

(Materka Model) $\alpha = 3.8333,\ \gamma = -0.3701.$

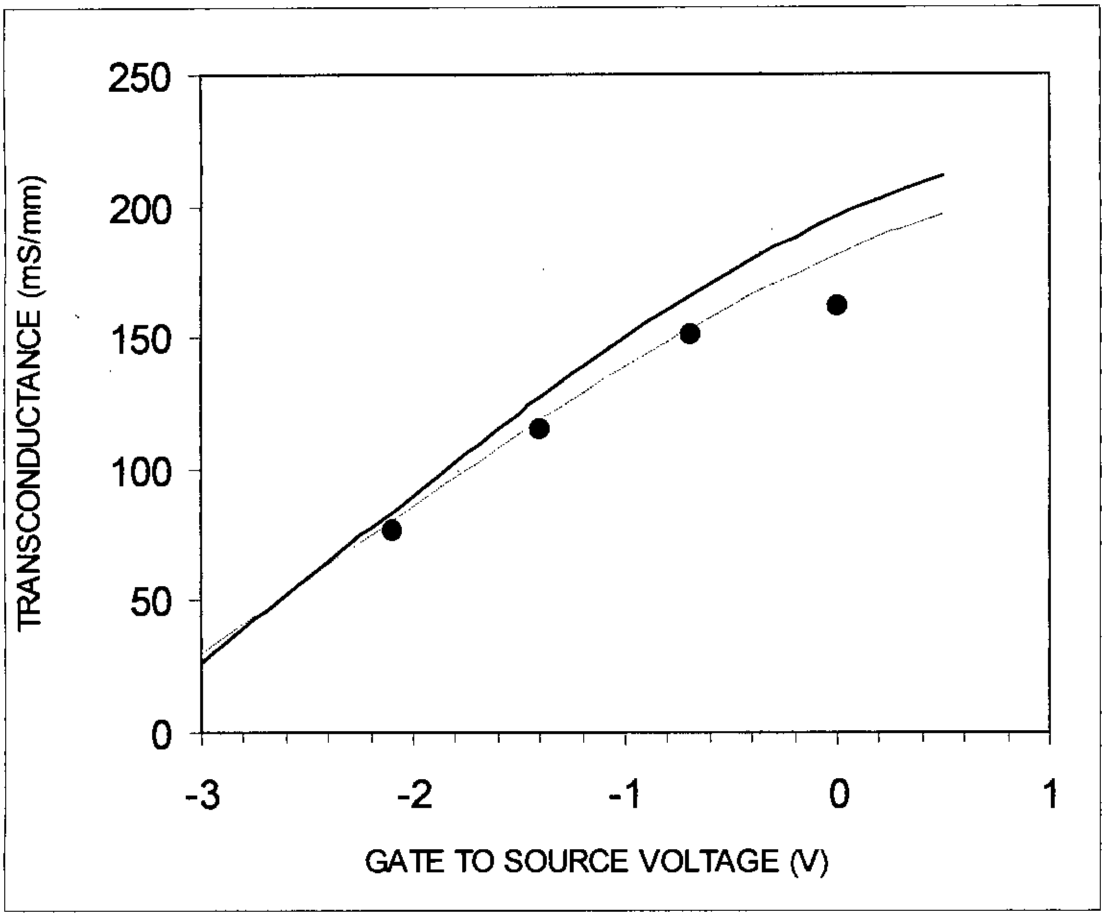


Figure 4.4: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using Kacprzak-Materka model and solid line shows the simulated characteristic using *FSIM* model.

$I_{DSS} = 192\ mA,\ V_T = -2\ V,\ V_{GS} = 0\ V,$

(FSIM Model) $\alpha = 3.2858,\ \gamma = -0.0553,\ \lambda = 0.0807,$

(Materka Model) $\alpha = 3.8387,\ \gamma = -0.1614.$

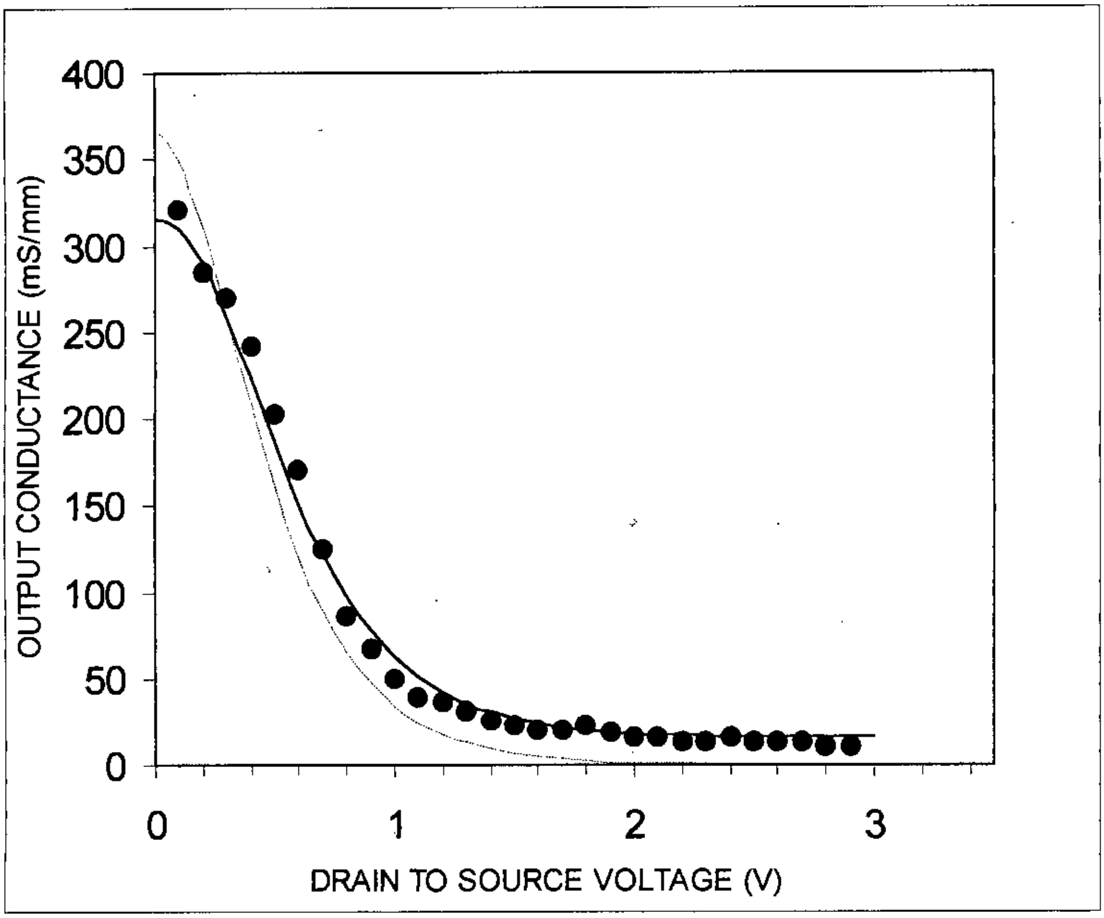


Figure 4.5: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristic [19] and dash line shows the simulated characteristic using Kacprzak-Materka model and solid line shows the simulated characteristic using *FSIM* model.

$I_{DSS} = 350\ mA,\ V_T = -2.8\ V,\ V_{GS} = 0\ V,$
(FSIM Model) $\alpha = 3.5168,\ \gamma = -0.3117,\ \lambda = 0.0302,$
(Materka Model) $\alpha = 3.8333,\ \gamma = -0.3701.$

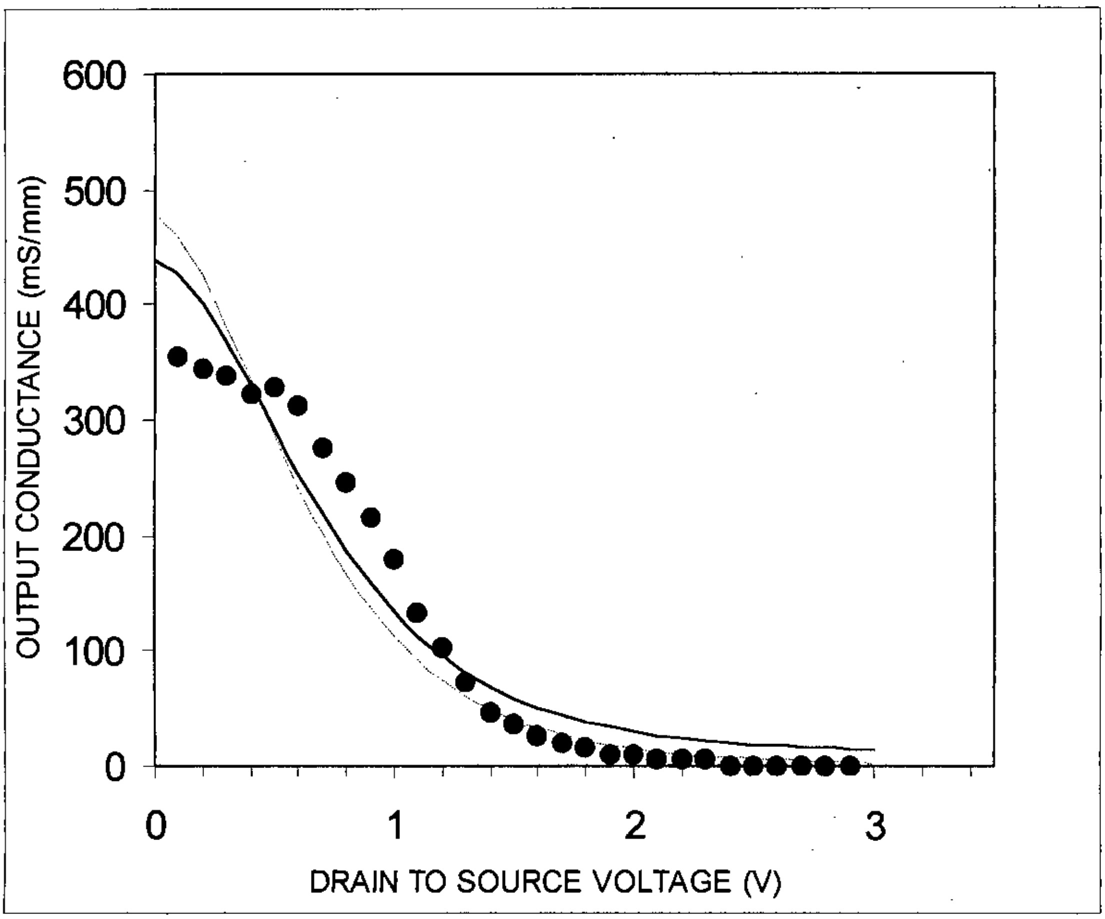


Figure 4.6: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using Kacprzak-Materka model and solid line shows the simulated characteristic using *FSIM* model.

4.3 Second Step Improvement of Materka Model

According to the Eastman and Shur [20] parasitic current and therefore output conductance, g_d, of sub-micron GaAs MESFET in the saturation region is independent of gate-to-source voltage, V_{GS}. But experimental results show that the output conductance in the saturation region depends not only on drain-to-source voltage, V_{DS}, but also on gate-to-source voltage, V_{GS} as reported by M. M. Ahmed [21]. In *FSIM* model, only variation of g_d with V_{DS} has been considered in the saturation region, by introducing $(1+\lambda V_{DS})$ term with the Kacprzak-Materka model, where λ is an empirical constant that simulates V_{DS} for the finite output conductance. But the effect of gate-to-source voltage V_{GS} on output conductance has been totally excluded. When V_{GS} is increased, the depletion layer under the gate expands, that means more positive charge is stored under the gate. The depleted electron due to positive charge storage is transported to the drain. As charge is stored under the gate, capacitive effect along with a resistive effect is observed in the gate shaded region. So the output conductance also varies with the variation of V_{GS} and this change of output conductance occurs both before and after saturation for submicron GaAs MESFET.

With the change of V_{GS}, the output conductance also changes due to another reason - for substrate current. For an applied voltage V_{DS} when the electron drift saturation occurred, a sharp rise in electric field is observed at the drain end under the gate. The peak field near the drain end of the gate may reach several hundred $kV\ cm^{-1}$. As a result of this high field, hot electrons are generated. The energized hot electrons are accelerated and may travel through the depleted to undepleted region to enter the substrate. For this electron flow, an extra current is produced known as substrate current. This substrate current increases the output conductance. Output conductance changes in this process with varying V_{GS} only when the current saturation has already occurred.

As parasitic current depends on both drain-to-source voltage (V_{DS}) and gate-to-source voltage (V_{GS}) and plays an important role when the GaAs MESFET operates at saturation, therefore, the output conductance not only depends on V_{DS} but also on V_{GS}. In order to add the above-mentioned effect, the term $(1+\lambda\ V_{DS})$ has been changed to $(1+\ \lambda\ V_{DS}+\ \beta\ V_{GS})$. Another empirical constant β is introduced. So in order to simulate drain-to-source current I_{DS} as a function of gate-to-source voltage

V_{GS} and drain-to-source voltage V_{DS} for sub-micron GaAs MESFETs, following relationship has been proposed:

$$I_{DS} = I_{DSS}\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2 \tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right) \tag{4.6}$$

Where,

λ = Simulates V_{DS} effectively for the finite conductance,

β = Simulates V_{GS} effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

γ = Bias dependency of the pinch-off voltage on V_{DS}.

To find out the validity of the ***Second Step Improvement of Materka (SSIM) model*** two different GaAs MESFET having different device dimensions are tested with both *FSIM* model and the new *SSIM* model to achieve the characteristics as shown in Figs. 4.7 and 4.8. In these figures, the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using *FSIM* model and solid lines show the simulated characteristics using *SSIM* model.

The magnitudes of transconductance, $g_m = \left(\frac{\partial I_{DS}}{\partial V_{GS}}\right)_{V_{DS}=const.}$ and output conductance, $g_d = \left(\frac{\partial I_{DS}}{\partial V_{DS}}\right)_{V_{GS}=const.}$ for *SSIM* Model can be evaluated from *SSIM* equation (4.6) and are expressed as follows:

Transconductance,

$$
\begin{aligned}
g_m = I_{DSS}\Bigg[& 2\cdot\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\cdot\left(\frac{-1}{V_T+\gamma\cdot V_{DS}}\right)\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right) \\
& +\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot(-V_{DS})}{(V_{GS}-V_T-\gamma\cdot V_{DS})^2}\right\}\cdot\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right) \\
& \cdot+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\beta\Bigg] \qquad (4.7)
\end{aligned}
$$

And output conductance,

$$
\begin{aligned}
g_d = I_{DSS}\Bigg[& 2\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)\frac{\gamma\cdot V_{GS}}{(V_T+\gamma\cdot V_{DS})^2}\cdot\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}\cdot}\right)\cdot\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right) \\
& +\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\right\}\cdot\left\{\frac{\alpha\cdot(V_{GS}-V_T-\gamma\cdot V_{DS})+\alpha\cdot\gamma\cdot V_{DS}}{(V_{GS}-V_T-\gamma\cdot V_{DS})^2}\right\} \\
& \cdot\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right)+\left(1-\frac{V_{GS}}{V_T+\gamma\cdot V_{DS}}\right)^2\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\gamma\cdot V_{DS}}\right)\cdot\lambda\Bigg] \qquad (4.8)
\end{aligned}
$$

To find out the validity of *SSIM* model, the simulated and the observed transconductances, g_m are plotted against gate-to-source voltage (V_{GS}) both for *FSIM* model and the new *SSIM* model as shown in Figs. 4.9 and 4.10. In these figures , the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using *FSIM* model and solid lines show the simulated characteristics using *SSIM* model.

Similarly, the simulated and the observed output conductance, g_d are plotted against drain-to-source voltage (V_{DS}) both for *FSIM* model and the new *SSIM* model as shown in Figs. 4.11 and 4.12.

From these plots it is observed that the simulated and the observed characteristics show a good degree of agreement. The degree of accuracy for the *SSIM* model is higher than that of the *FSIM* model (Figs. 4.7, 4.8, 4.11 & 4.12). Slight discrepancy is observed for the g_m plots (Figs. 4.9 & 4.10).

Table 4.2 shows the comparison between these two models in tabular form for the same devices. The SSIM model shows good agreement with experimental characteristics especially for high V_{GS} voltages. In device A-64-2, the maximum magnitude of the MSE is 11.95 for *FSIM* model at $V_{GS} = 0$ V, whereas MSE is 10.30 for the proposed *SSIM* model. Device A-74-1 has relatively high conductance after saturation. Hence the *FSIM* model accuracy deteriorates, compared to that of the *SSIM* model. At $V_{GS} = -1.0$ V the magnitude of the MSE for the FSIM model is 9.46, whereas for the same gate voltage it is only 1.68 for the *SSIM* model.

In general, it can be stated that with the proposed *SSIM* model it is possible to simulate a wide variety of GaAs MESFETs with a reasonable accuracy.

Table 4.2: Comparative data of the *FSIM* model and *SSIM* model.

Device	V_{GS}	**FSIM Model**				**SSIM Model**			
		RMS Error	MSE	Avg. MSE	Avg. RMS Error	RMS Error	MSE	Avg. MSE	Avg. RMS Error
A-74-1	0.0 V	2.1609	3.7622	6.2136	0.2863	0.7894	3.0293	2.3432	0.1603
	-0.5 V	3.2194	5.9425			2.7422	2.9244		
	-1.0 V	4.2773	9.4571			0.6025	1.684		
	-1.5 V	0.8174	5.8757			1.8716	2.5513		
	-2.0 V	4.425	4.4291			0.0093	0.617		
A-64-2	0.0 V	3.4805	11.9543	8.9885	1.0148	0.5549	10.3003	7.2207	0.5312
	-0.7 V	8.1162	10.9881			1.9807	6.232		
	-1.4 V	3.5378	5.6189			1.052	6.9696		
	-2.1 V	1.4348	4.5999			0.5261	6.9007		
	-2.8 V	9.3023	9.3589			4.3375	4.4232		

$I_{DSS} = 192\ mA,\ V_T = -2\ V,$

(SSIM Model) $\alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -0.3558,\ \lambda = 0.0878,$

(FSIM Model) $\alpha = 3.2858,\ \gamma = -0.0553,\ \lambda = 0.0807.$

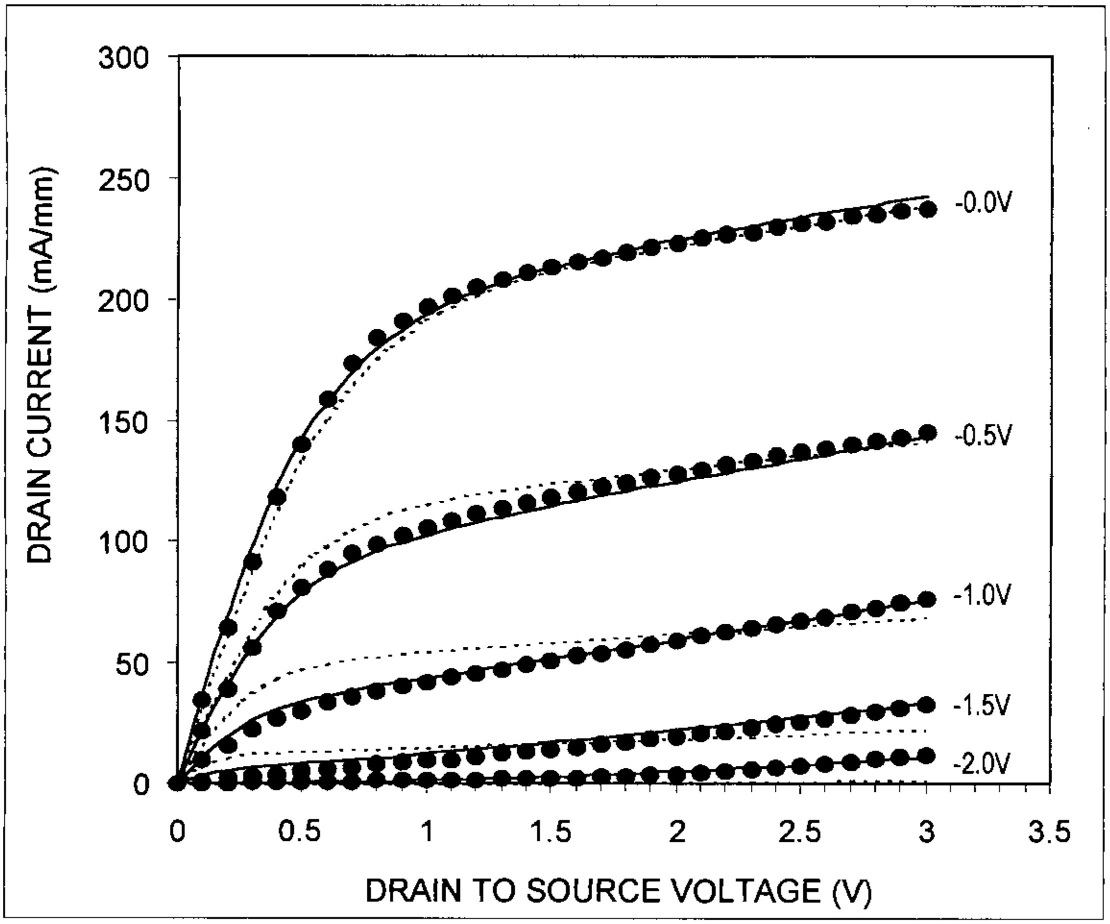


Figure 4.7: Observed and simulated output characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristics [19] and dash lines show the simulated characteristics using *FSIM* model and solid lines show the simulated characteristics using *SSIM* model.

I_{DSS} = 350 mA, V_T = - 2.8 V,
(SSIM Model) α = 3.3213, β = - 0.1219, γ = - 0.1901, λ = 0.0239,
(FSIM Model) α = 3.5168, γ = - 0.3117, λ = 0.0302.

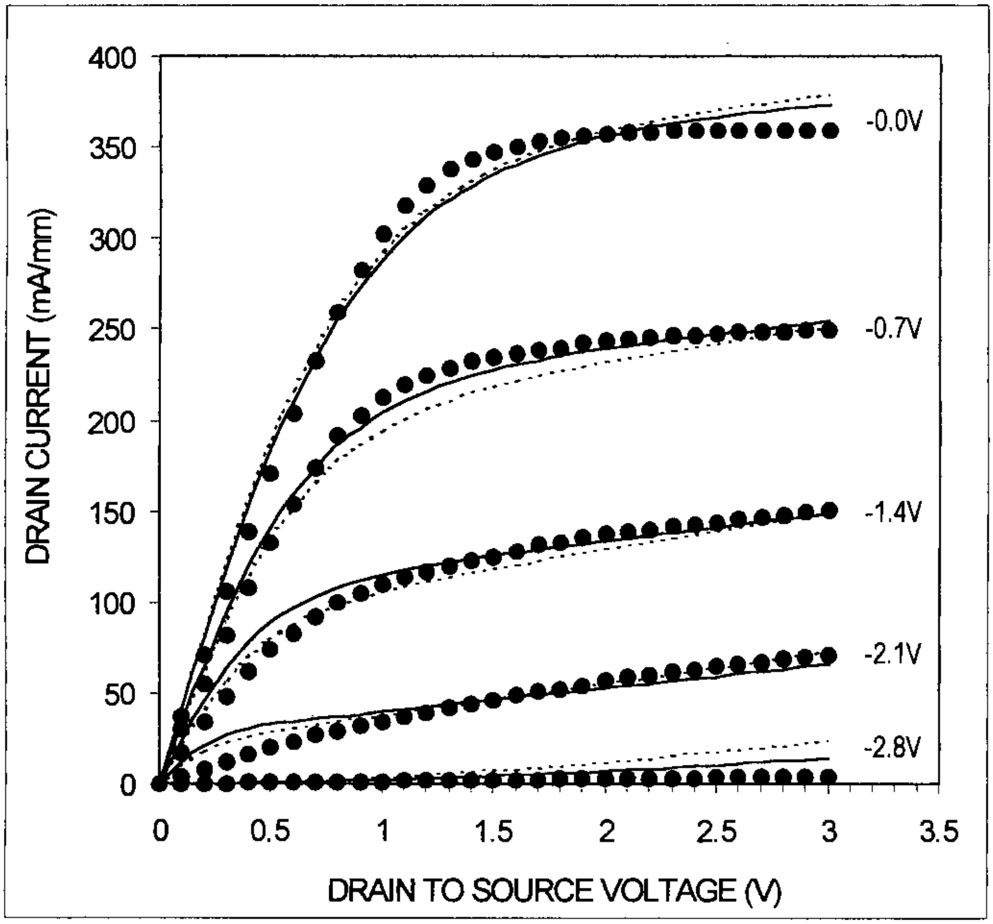


Figure 4.8: Observed and simulated output characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristics [18] and dash lines show the simulated characteristics using *FSIM* model and solid lines show the simulated characteristics using *SSIM* model.

$I_{DSS} = 192\ mA,\ V_T = -\ 2\ V,\ V_{DS} = 2\ V,$

$(SSIM\ Model)\ \alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -\ 0.3558,\ \lambda = 0.0878,$

$(FSIM\ Model)\ \alpha = 3.2858,\ \gamma = -\ 0.0553,\ \lambda = 0.0807.$

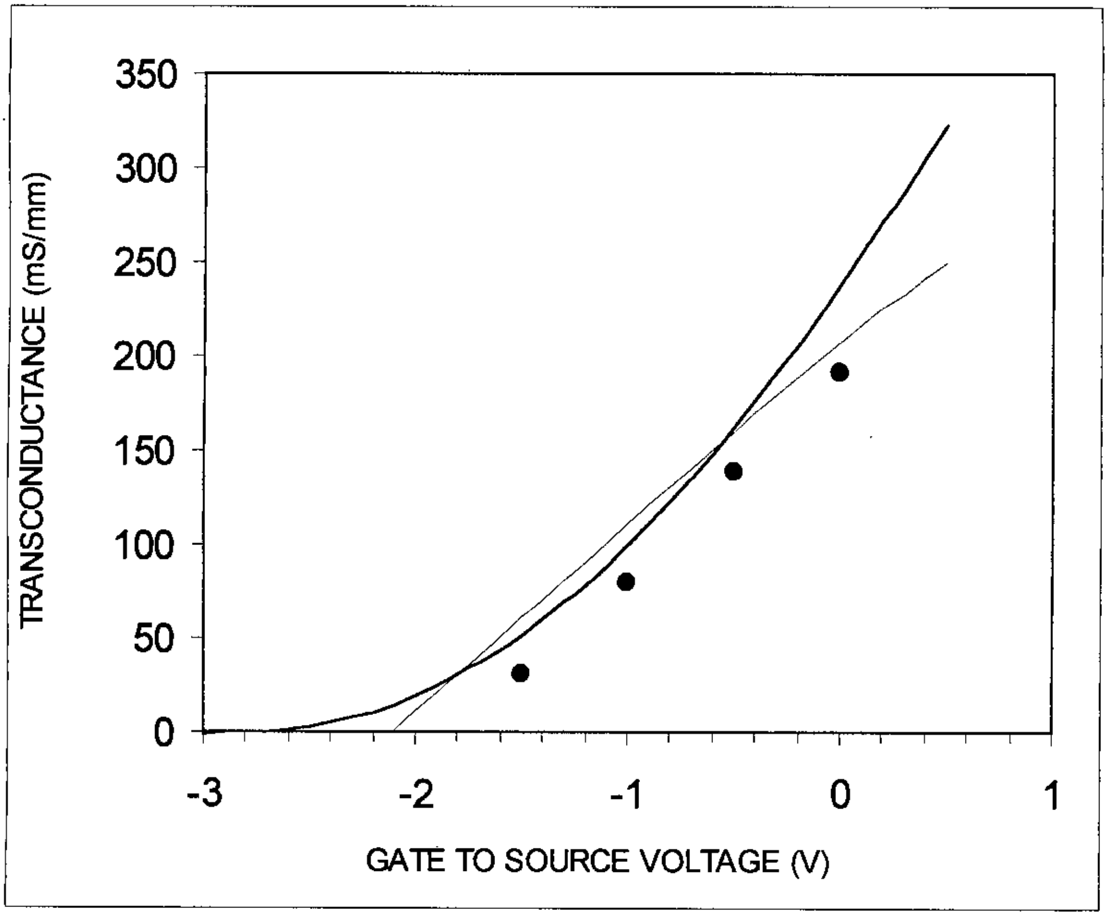


Figure 4.9: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristic [19] and dash line shows the simulated characteristic using *FSIM* model and solid line shows the simulated characteristic using *SSIM* model.

I_{DSS} = 350 mA, V_T = - 2.8 V, V_{DS} = 2 V,
(SSIM Model) α = 3.3213, β = - 0.1219, γ = - 0.1901, λ = 0.0239,
(FSIM Model) α = 3.5168, γ = - 0.3117, λ = 0.0302.

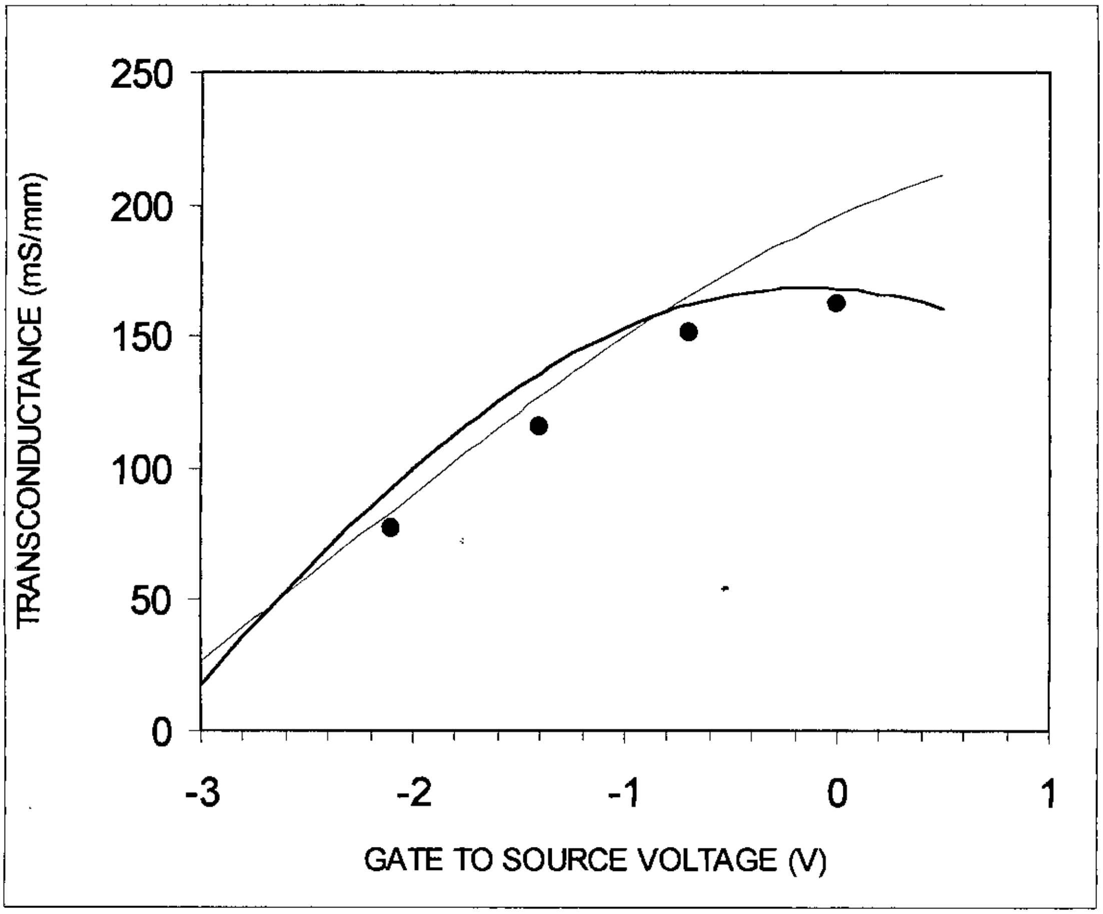


Figure 4.10: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using *FSIM* model and solid line shows the simulated characteristic using *SSIM* model.

$I_{DSS} = 192\ mA,\ V_T = -2\ V,\ V_{GS} = 0\ V,$

(SSIM Model) $\alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -0.3558,\ \lambda = 0.0878,$

(FSIM Model) $\alpha = 3.2858,\ \gamma = -0.0553,\ \lambda = 0.0807.$

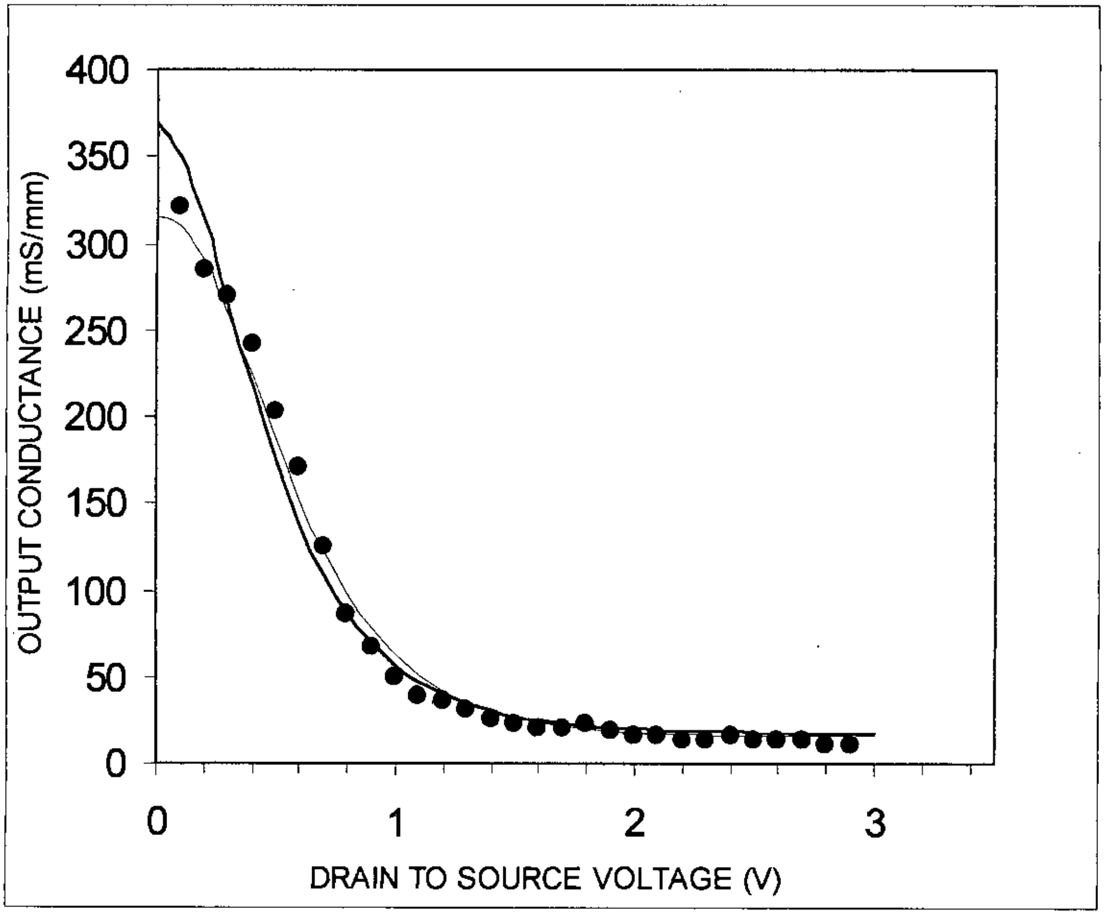


Figure 4.11: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristic [19] and dash line shows the simulated characteristic using *FSIM* model and solid line shows the simulated characteristic using *SSIM* model.

$I_{DSS} = 350\ mA,\ V_T = -2.8\ V,\ V_{GS} = 0\ V,$

(SSIM Model) $\alpha = 3.3213,\ \beta = -0.1219,\ \gamma = -0.1901,\ \lambda = 0.0239,$

(FSIM Model) $\alpha = 3.5168,\ \gamma = -0.3117,\ \lambda = 0.0302.$

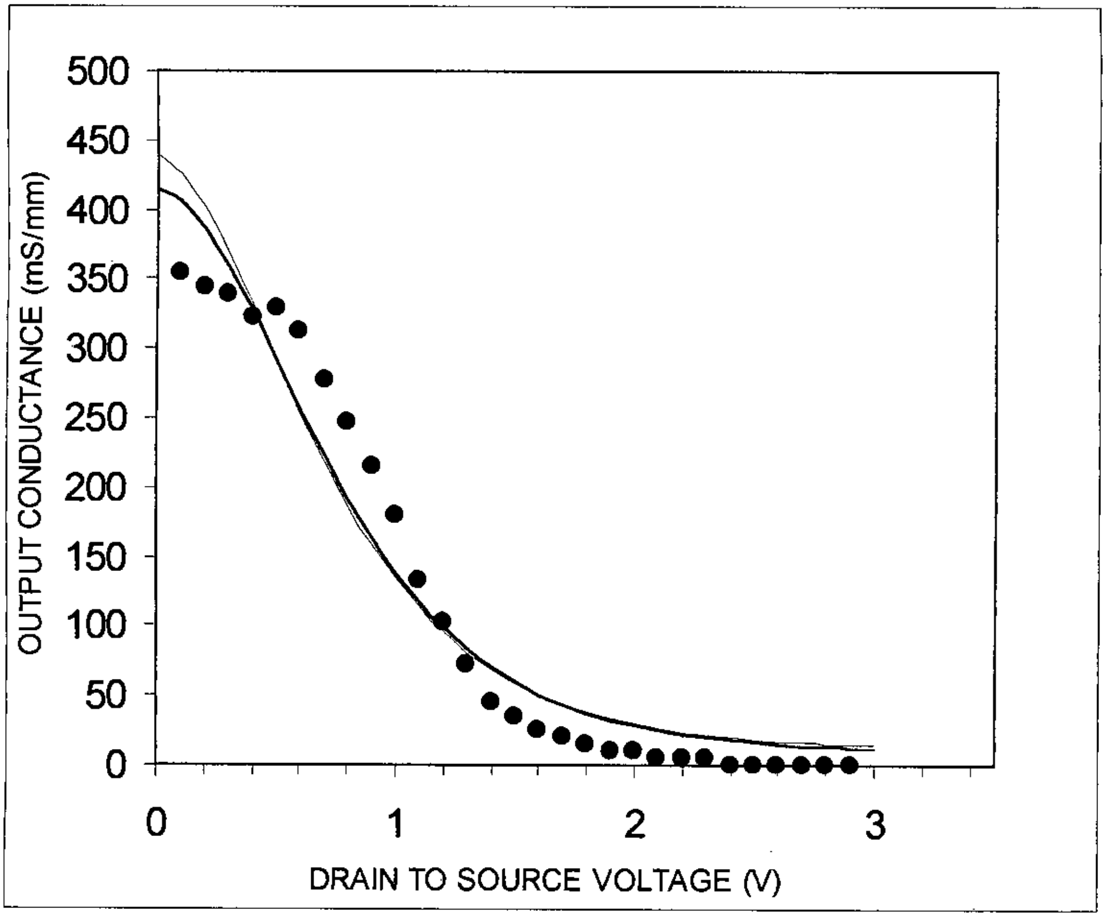


Figure 4.12: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using *FSIM* model and solid line shows the simulated characteristic using *SSIM* model.

4.4 Third Step Improvement of Materka Model

So far the proposed model i.e. *SSIM model* (equation 4.6) is a *six-parameter* model. The parameters are as follows:

I_{DSS} = Drain saturation current at $V_{GS} = 0\ V$,

V_T = Threshold voltage, V;

λ = Simulates V_{DS} effectively for the finite conductance,

β = Simulates V_{GS} effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

γ = Bias dependency of the pinch-off voltage on V_{DS}.

Four of them (α, β, γ and λ) are empirical constants, which are estimated by the best-fit method from the observed and simulated characteristics. They have no physical significances. The values of I_{DSS} and V_T are attained from the terminal measurements of the device. To reduce the number of parameters in this step of improvement, V_T can be determined from the physical parameters given by the equation (2.2):

$$V_T = \frac{q \cdot N \cdot W^2}{2\,\varepsilon_S} - V_{BO}$$

Where,

q = Electronic charge ($1.6 \times 10^{-19}\ C$),

N = Channel doping density,

W = Channel thickness,

ε_s = Permitivity of GaAs ($13.2 \times 8.85 \times 10^{-12}\ F/m$),

V_{BO} = Schottky barrier height.

But there is a significant difference between the observed V_T and the calculated V_T. Moreover the difference is greater for short gate length devices. Though according to the one-dimensional equation (2.2) V_T should be independent of channel length L_G, Enoki et al. [16] have shown that, in fact, V_T is a function of the device channel length in short channel devices.

4.4.1 Enoki et al. Model for Threshold Voltage Shift

A simple theory for threshold-voltage shift was proposed based on the charge share model [17]. Figure 4.13 shows a schematic view of a MESFET. In the one-dimensional model, considering only region *II*, the potential V_T across the depletion layer can be obtained by the application of Gauss's law to the surface charges on the Schottky interface that corresponds to the space charges in the depletion region and is given by

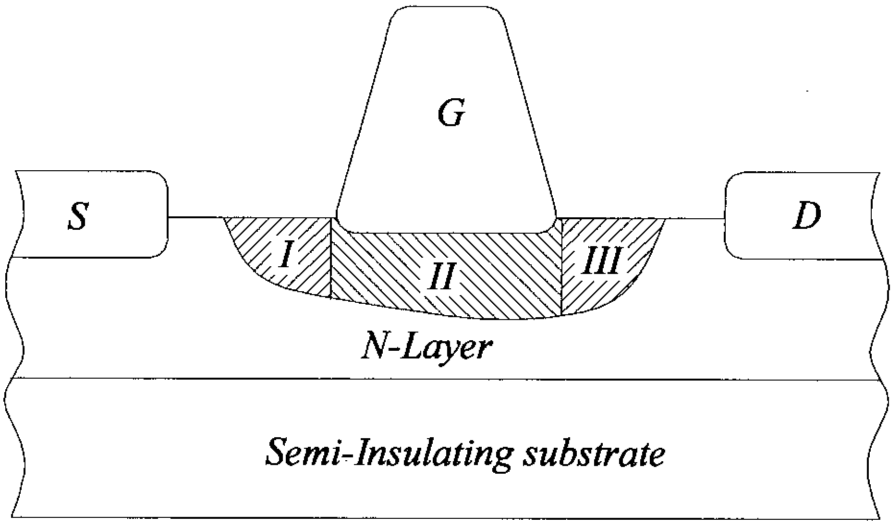


Figure 4.13: Schematic view of a MESFET illustrating the regions in the depletion layer.

$$V_T = \delta \cdot \frac{Q_{II}}{\varepsilon_S \cdot S} = \frac{q \cdot N \cdot W^2}{2\,\varepsilon_S} \tag{4.9}$$

where,

Q_{II} = $q\,N\,W\,L_G\,Z_G$ is the space charge in region *II*,

q = Electronic charge (1.6×10^{-19} C),

N = Channel doping density,

L_G = Gate length,

Z_G = Gate width,

δ = Average depth of the space charge,

= *W/2* for the uniform carrier density,

W = Gate channel thickness,

S = Gate surface area, $L_G\,Z_G$.

Next considering the two-dimensional effect including Regions *I* and *III*, these regions are attributed to the two-dimensional spread of the electric field. The excess surface charges in regions *I* and *III* are

$$Q_I + Q_{III} = \frac{q \cdot N \cdot W^2 \cdot \pi \cdot Z_G}{2} \qquad (4.10)$$

Taking quarter-circle shape into account, the average depth, δ of the space charges in Regions *I* and *III* is $4W/3\pi$. If it is assumed roughly that these excess charges are uniformly distributed on the Schottky interface, the threshold voltage shift (ΔV_T) due to these excess charges can be derived in the same way as in equation (4.9). Thus,

$$\Delta V_T = \delta \cdot \frac{(Q_I + Q_{III})}{\varepsilon_S \cdot S} = V_T \frac{4\,W}{3\,L_G} \qquad (4.11)$$

Therefore the total threshold voltage is now given by ($V_T + \Delta V_T$).

So in order to simulate drain-to-source current I_{DS} as a function of gate-to-source voltage V_{GS} and drain-to-source voltage V_{DS} for sub-micron GaAs MESFETs, following relationship can be proposed:

$$I_{DS} = I_{DSS}\left(1 - \frac{V_{GS}}{V_T + \Delta V_T + \gamma \cdot V_{DS}}\right)^2 \tanh\left(\frac{\alpha \cdot V_{DS}}{V_{GS} - V_T - \Delta V_T - \gamma \cdot V_{DS}}\right)\left(1 + \lambda \cdot V_{DS} + \beta \cdot V_{GS}\right) \qquad (4.12)$$

Where,

λ = Simulates V_{DS} effectively for the finite conductance,

β = Simulates V_{GS} effectively for the finite conductance,

α = Determines the V_{DS} where the drain current saturates,

γ = Bias dependency of the pinch-off voltage on V_{DS}.

ΔV_T = Geometrical threshold voltage shift.

To find out the validity of the ***Third Step Improvement of Materka (TSIM) model*** two different GaAs MESFETs having different device dimensions are tested with both *SSIM* model and the new *TSIM* model to achieve the characteristics as shown in Figs. 4.14 and 4.15. In these figures , the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using *SSIM* model and solid lines show the simulated characteristics using *TSIM* model.

The magnitudes of transconductance, $g_m = \left(\dfrac{\partial I_{DS}}{\partial V_{GS}}\right)_{V_{DS}=const.}$ and output conductance, $g_d = \left(\dfrac{\partial I_{DS}}{\partial V_{DS}}\right)_{V_{GS}=const.}$ for *TSIM* model can be evaluated from *TSIM* equation (4.12) and are expressed as follows:

Transconductance,

$$
\begin{aligned}
g_m = I_{DSS}\Bigg[& 2\cdot\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)\cdot\left(\frac{-1}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right) \\
& (1+\lambda\cdot V_{DS}+\beta\cdot V_{GS})+\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)^2\Bigg\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right)\Bigg\} \\
& \left\{\frac{\alpha\cdot(-V_{DS})}{(V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS})^2}\right\}\cdot(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS})+\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)^2 \\
& \cdot\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right)\cdot\beta\Bigg]
\end{aligned}
\tag{4.13}
$$

And, output conductance,

$$g_d = I_{DSS}\left[2\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)\frac{\gamma\cdot V_{GS}}{\left(V_T+\Delta V_T+\gamma\cdot V_{DS}\right)^2}\cdot\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right)\right.$$

$$\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right)+\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)^2\left\{1-\tanh^2\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right)\right\}$$

$$\left\{\frac{\alpha\cdot\left(V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}\right)+\alpha\cdot\gamma\cdot V_{DS}}{\left(V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}\right)^2}\right\}\cdot\left(1+\lambda\cdot V_{DS}+\beta\cdot V_{GS}\right)+\left(1-\frac{V_{GS}}{V_T+\Delta V_T+\gamma\cdot V_{DS}}\right)^2$$

$$\left.\tanh\left(\frac{\alpha\cdot V_{DS}}{V_{GS}-V_T-\Delta V_T-\gamma\cdot V_{DS}}\right)\cdot\lambda\right] \tag{4.14}$$

To find out the validity of *TSIM* model, the simulated and the observed transconductances, g_m are plotted against gate-to-source voltage (V_{GS}) both for *SSIM* model and the new *TSIM* model as shown in Figs. 4.16 and 4.17. In these figures, the dots represent the experimental or measured characteristics and dash lines show the simulated characteristics using *SSIM* model and solid lines show the simulated characteristics using *TSIM* model.

Similarly, the simulated and the observed output conductances, g_d are plotted against drain-to-source voltage (V_{DS}) both for *SSIM* model and the new *TSIM* model as shown in Figs. 4.18 and 4.19.

These plots are shown on the following pages and it is observed that the simulated and the observed characteristics show a good degree of agreement (exception Fig. 4.17). Table 4.3 shows the comparison between these two models.

Table 4.3: Comparative data of the *SSIM* model and *TSIM* model.

Device	V_{GS}	**SSIM Model**				**TSIM Model** **For A-74-1 is -1.69 V** **For A-64-2 is -2.45 V**			
		RMS Error	MSE	Avg. MSE	Avg. RMS Error	RMS Error	MSE	Avg. MSE	Avg. RMS Error
A-74-1	0.0 V	0.7894	3.0293	2.3432	0.1603	0.1164	2.814	1.8905	0.1870
	-0.5 V	2.7422	2.9244			1.1388	1.7784		
	-1.0 V	0.6025	1.684			0.9574	1.6006		
	-1.5 V	1.8716	2.5513			0.5181	0.6806		
	-2.0 V	0.0093	0.617			1.4249	1.9401		
A-64-2	0.0 V	0.5549	10.3003	7.2207	0.5312	2.4292	11.2096	7.3046	0.7551
	-0.7 V	1.9807	6.232			0.8736	7.8963		
	-1.4 V	1.052	6.9696			0.6241	5.2085		
	-2.1 V	0.5261	6.9007			4.3479	5.5654		
	-2.8 V	4.3375	4.4232			3.7221	4.5477		

$I_{DSS} = 192\ mA,$

(TSIM Model) $V_T + \Delta V_T = -\ 1.69\ V,\ \alpha = 3.241,\ \beta = 0.1929,\ \gamma = -\ 0.3042,\ \lambda = 0.0853,$

(SSIM Model) $V_T = -\ 2\ V,\ \alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -\ 0.3558,\ \lambda = 0.0878.$

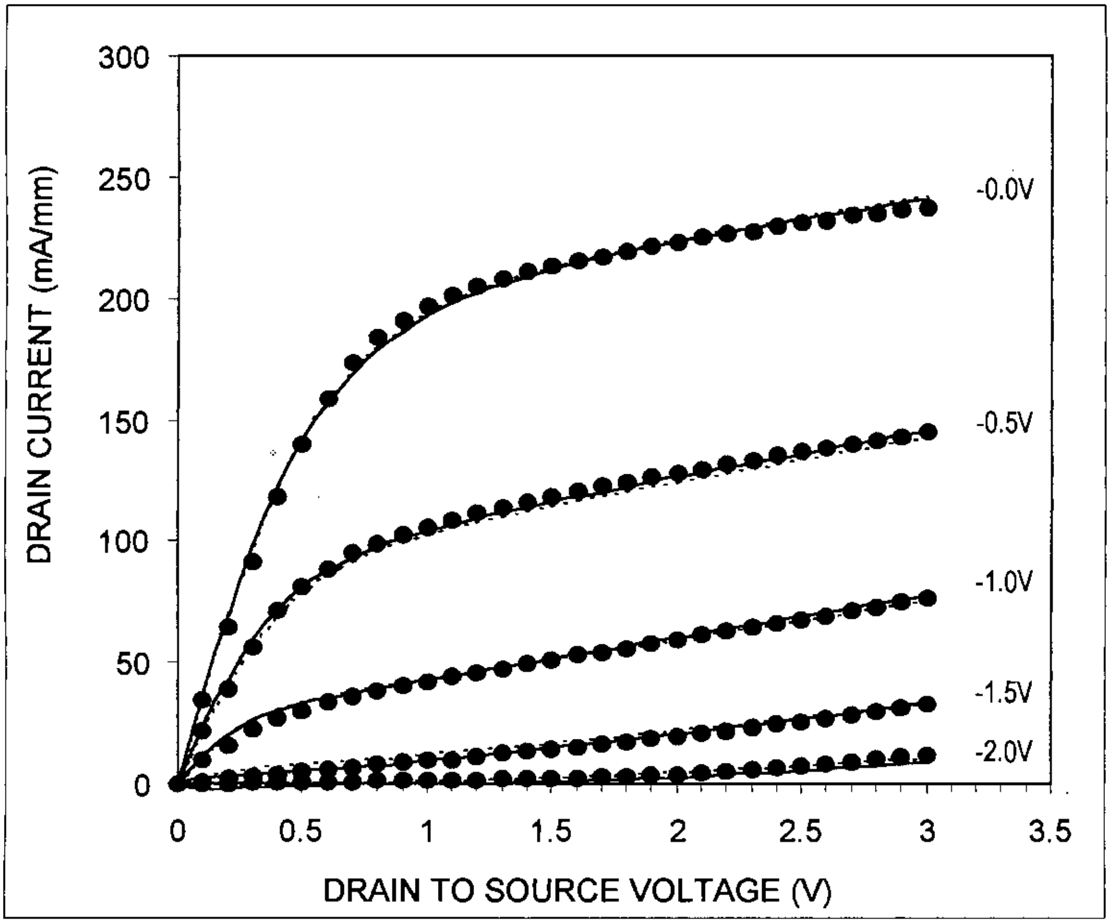


Figure 4.14: Observed and simulated output characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristics [19] and dash lines show the simulated characteristics using *SSIM* model and solid lines show the simulated characteristics using *TSIM* model.

$I_{DSS} = 350\ mA,$

(TSIM Model) $V_T + \Delta V_T = -2.45\ V,\ \alpha = 3.075,\ \beta = -0.194,\ \gamma = -0.2984,\ \lambda = 0.0211,$

(SSIM Model) $V_T = -2.8\ V,\ \alpha = 3.3213,\ \beta = -0.1219,\ \gamma = -0.1901,\ \lambda = 0.0239.$

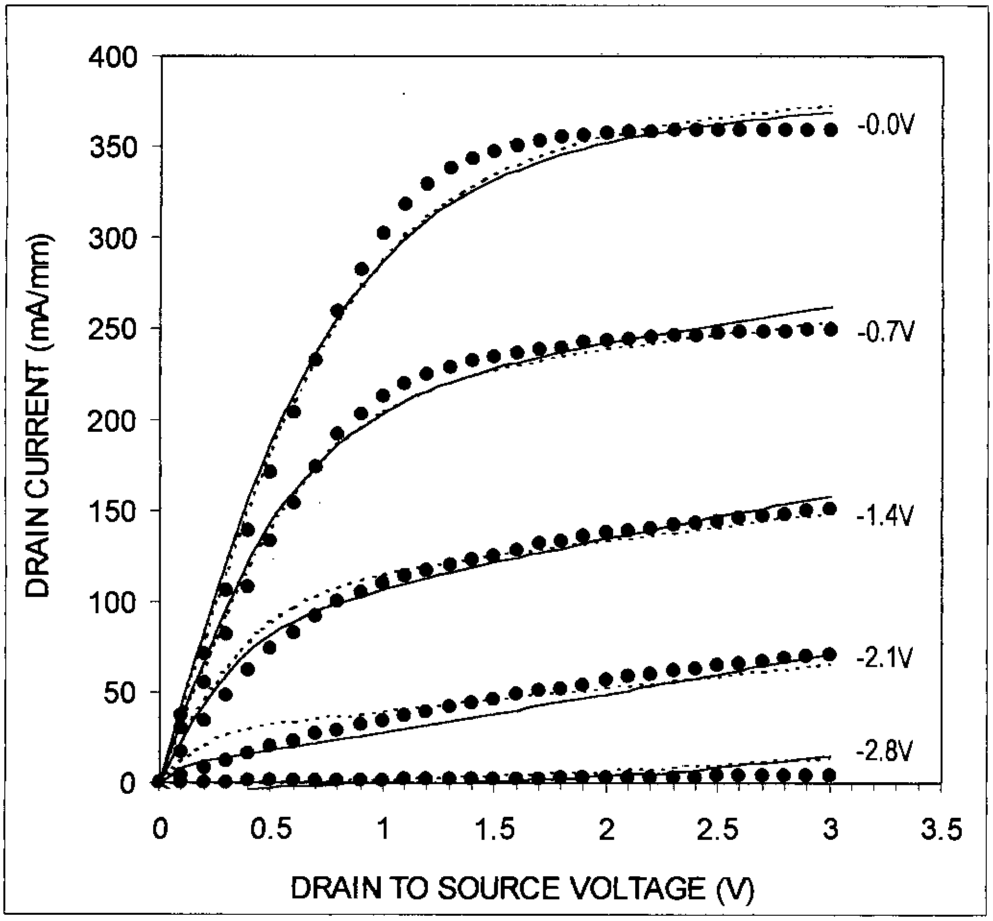


Figure 4.15: Observed and simulated output characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristics [18] and dash lines show the simulated characteristics using *SSIM* model and solid lines show the simulated characteristics using *TSIM* model.

$I_{DSS} = 192\ mA,\ V_{DS} = 2\ V,$

$(TSIM\ Model)\ V_T + \Delta V_T = -1.69\ V,\ \alpha = 3.241,\ \beta = 0.1929,\ \gamma = -0.3042,\ \lambda = 0.0853,$

$(SSIM\ Model)\ V_T = -2\ V,\ \alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -0.3558,\ \lambda = 0.0878.$

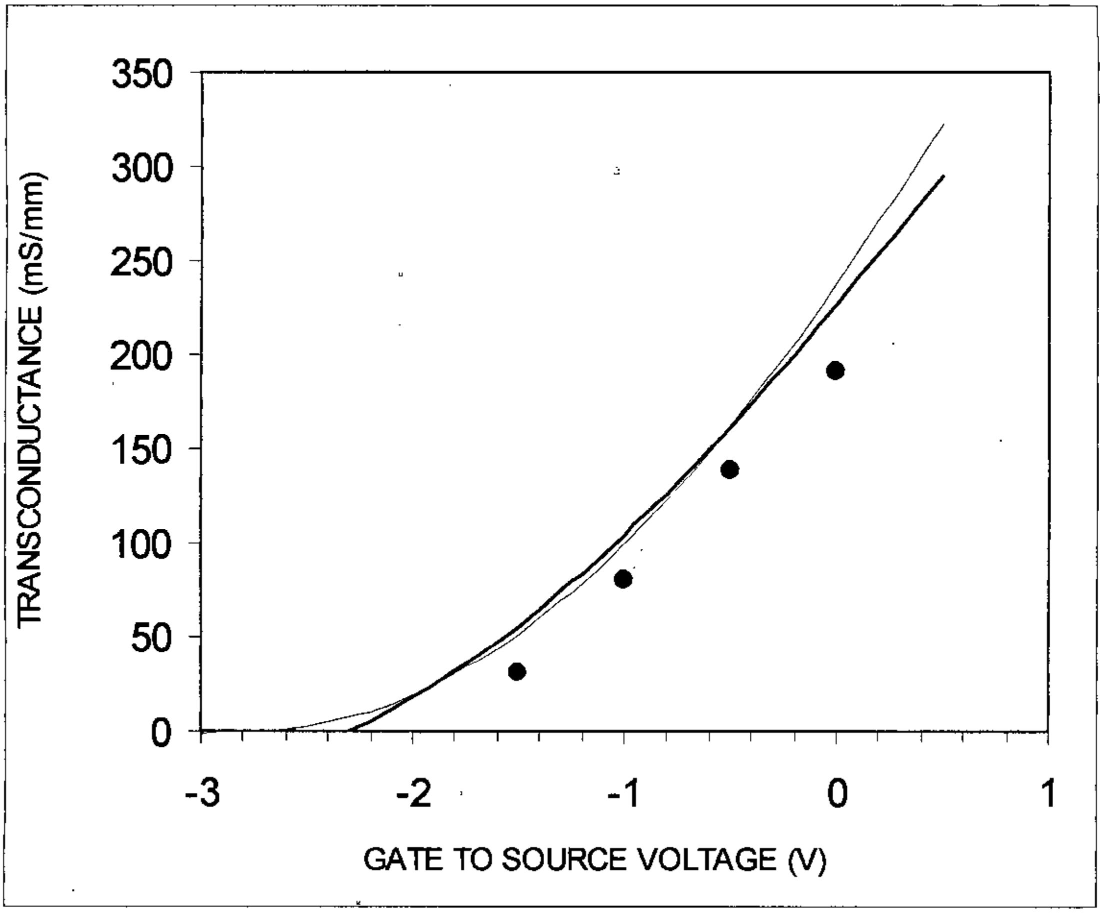


Figure 4.16: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristic [19] and dasb line shows the simulated characteristic using *SSIM* model and solid line shows the simulated characteristic using *TSIM* model.

$I_{DSS} = 350\ mA$, $V_{DS} = 2\ V$,

(TSIM Model) $V_T + \Delta V_T = -2.45\ V$, $\alpha = 3.075$, $\beta = -0.194$, $\gamma = -0.2984$, $\lambda = 0.0211$,

(SSIM Model) $V_T = -2.8\ V$, $\alpha = 3.3213$, $\beta = -0.1219$, $\gamma = -0.1901$, $\lambda = 0.0239$.

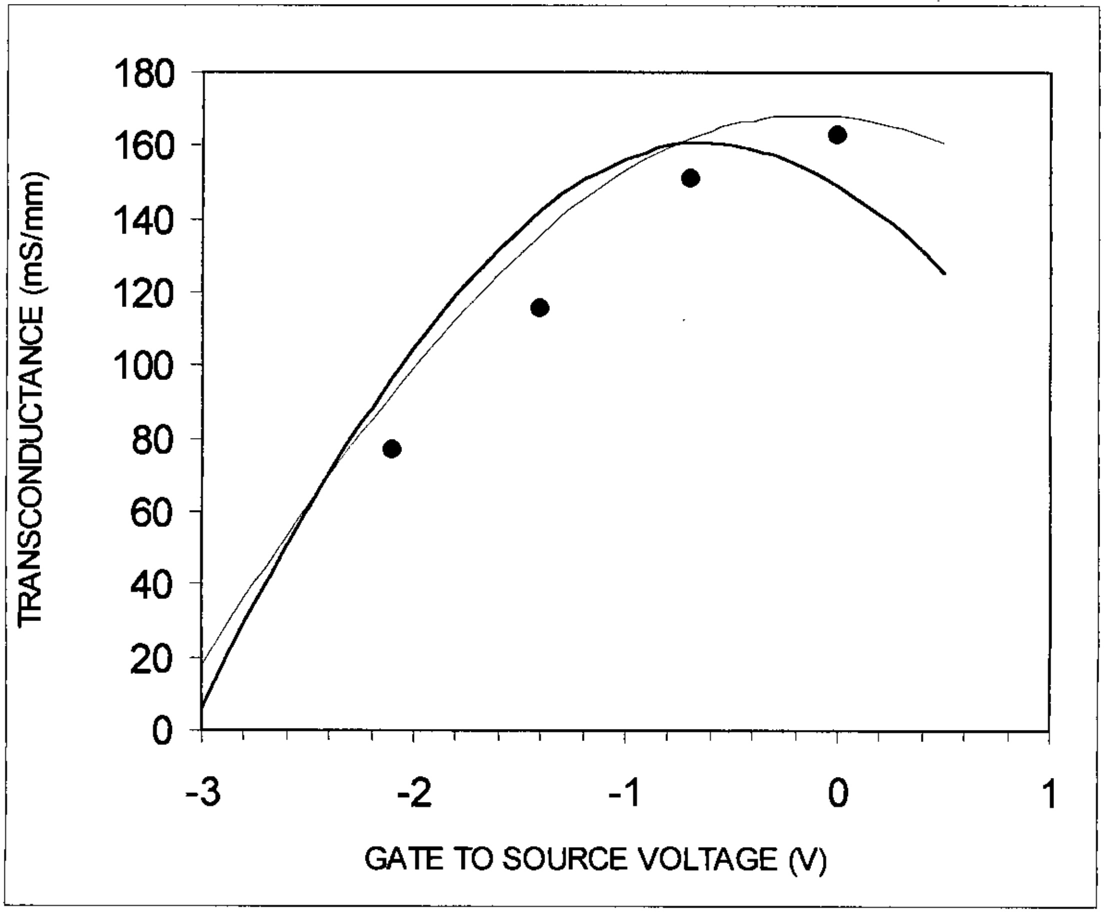


Figure 4.17: Observed and simulated transconductance versus gate-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using *SSIM* model and solid line shows the simulated characteristic using *TSIM* model.

$I_{DSS} = 192\ mA,\ V_{GS} = 0\ V,$

(TSIM Model) $V_T + \Delta V_T = -1.69\ V,\ \alpha = 3.241,\ \beta = 0.1929,\ \gamma = -0.3042,\ \lambda = 0.0853,$

(SSIM Model) $V_T = -2\ V,\ \alpha = 3.8458,\ \beta = 0.3988,\ \gamma = -0.3558,\ \lambda = 0.0878.$

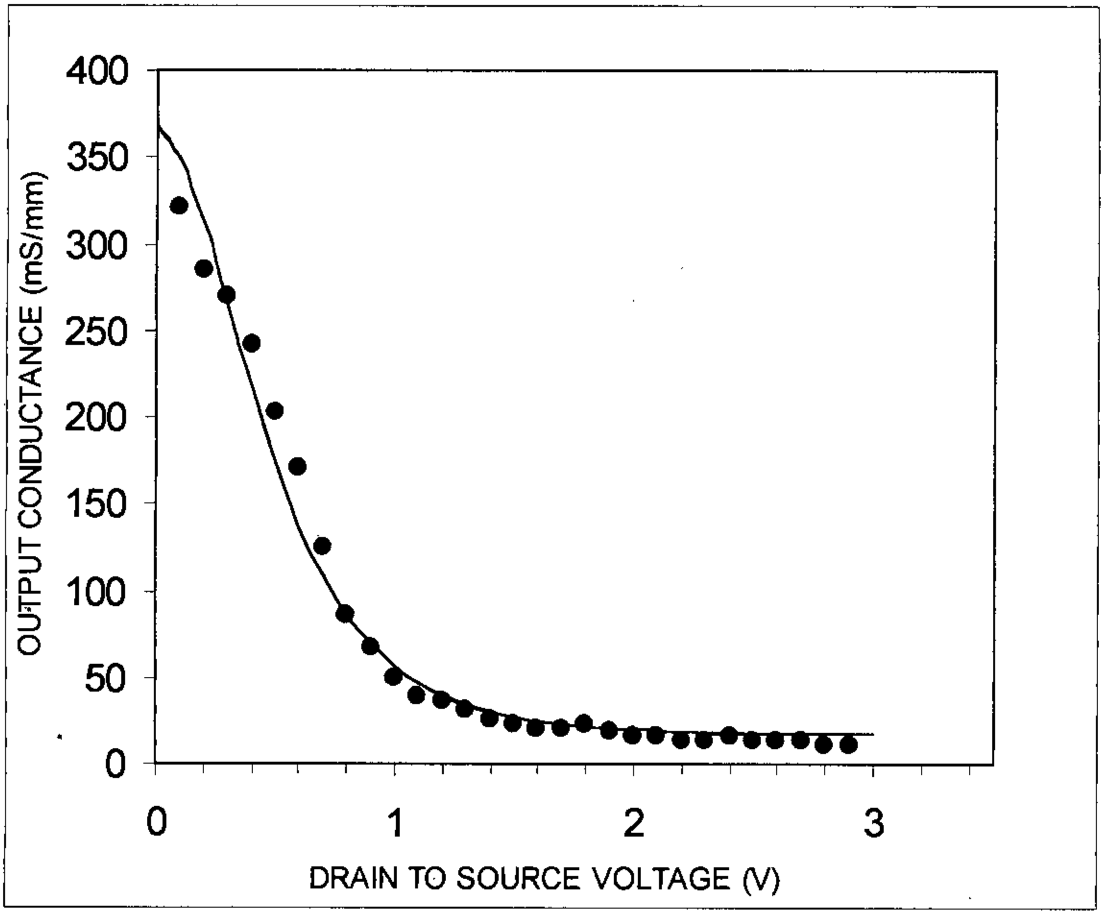


Figure 4.18: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-74-1. In this Figure, the dots represent the experimental or measured characteristic [19] and dash line shows the simulated characteristic using *SSIM* model and solid line show the simulated characteristic using *TSIM* model. In this figure the dash line and solid line merge together.

I_{DSS} = 350 mA, V_{GS} = 0 V,

(TSIM Model) $V_T + \Delta V_T = -2.45$ V, $\alpha = 3.075$, $\beta = -0.194$, $\gamma = -0.2984$, $\lambda = 0.0211$,

(SSIM Model) $V_T = -2.8$ V, $\alpha = 3.3213$, $\beta = -0.1219$, $\gamma = -0.1901$, $\lambda = 0.0239$.

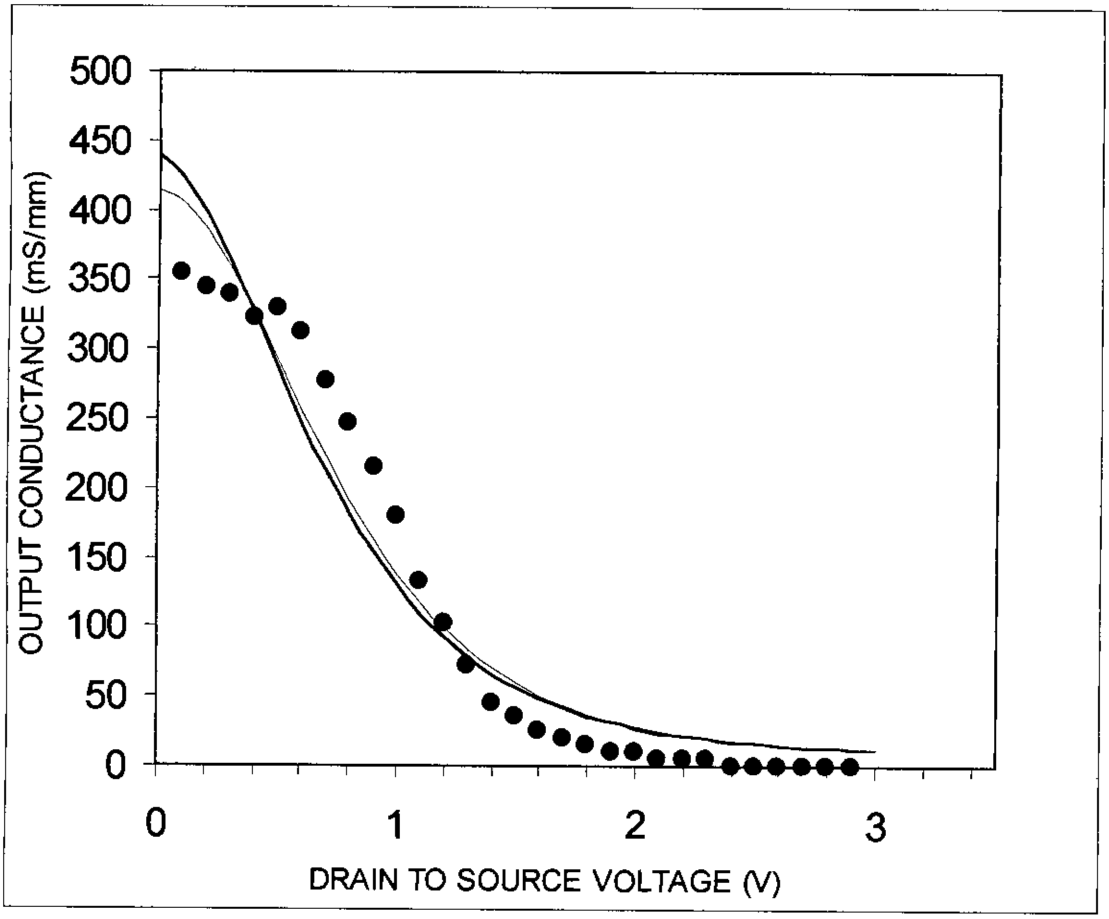


Figure 4.19: Observed and simulated output conductance versus drain-to-source voltage characteristics of GaAs MESFET for device A-64-2. In this Figure, the dots represent the experimental or measured characteristic [18] and dash line shows the simulated characteristic using *SSIM* model and solid line shows the simulated characteristic using *TSIM* model.

CHAPTER V

Chapter Five

ALGORITHM DEVELOPMENT

5.1 MSE and RMS

The difference between the root mean square (RMS) values of two curves may be calculated to find out the degree of accuracy between two curves. It may be assumed that if this difference between RMS values i.e. RMS error is zero, then these two curves are same. Higher the RMS error, more they differ from each other. But sometimes RMS values of two completely different curves may be same. So their difference will be zero and hence decision might have been taken that they are same. Therefore, measuring RMS error is not an efficient method to find out degree of accuracy of the simulated curve with the measured one.

On the other hand, Mean Square Error (MSE) deals with the difference in every point (at a certain interval along the x-axis). Average of the square of each distance is called the mean square error (MSE). So if a curve differs from another curve even by slight margin, then the MSE becomes non-zero and indicates that these curves do not coincide. Therefore, MSE have been used in the simulation process to find out the degree of accuracy of the simulated curves.

5.2 Algorithm for Materka Model

The values of threshold voltage, V_T, and drain saturation current, I_{DSS} are attained from the terminal measurements of the device and empirical constants are estimated by computing mean square error (MSE) values from the observed and simulated characteristics. In this procedure, an algorithm has been designed which initially chooses the best possible value of empirical constant α, by iterating all the values within the prescribed limits. After evaluating the best value of α for which MSE becomes smallest, the algorithm generates all possible combination of α and γ to calculate the optimal output characteristics. Finally, the optimal values of α and γ

are used to plot the *I-V* characteristics of the GaAs MESFETs. Now it is quite clear from the algorithm that the accuracy of selecting the optimum values of empirical constants depends totally on the "*step increase*" of these constants provided by the user. If the step sizes are large, optimum values obtained by the algorithm may be far away from the correct optimum values. This happens due to non-linearity of the *I-V* characteristics of the GaAs MESFET. Therefore, the degree of accuracy depends on how small the step sizes of the two empirical constants are. Smaller the value of the step sizes, higher the degree of accuracy. But making the step size smaller in a certain range of empirical constants increases the number of iteration and hence the computer program takes too much time to execute. So a new algorithm is introduced to increase the degree of accuracy without increasing the number of iteration that much. The algorithm process is shown in the flow-chart in Fig. 5.1, which has been used to simulate the Materka model in order to generate data on a PC.

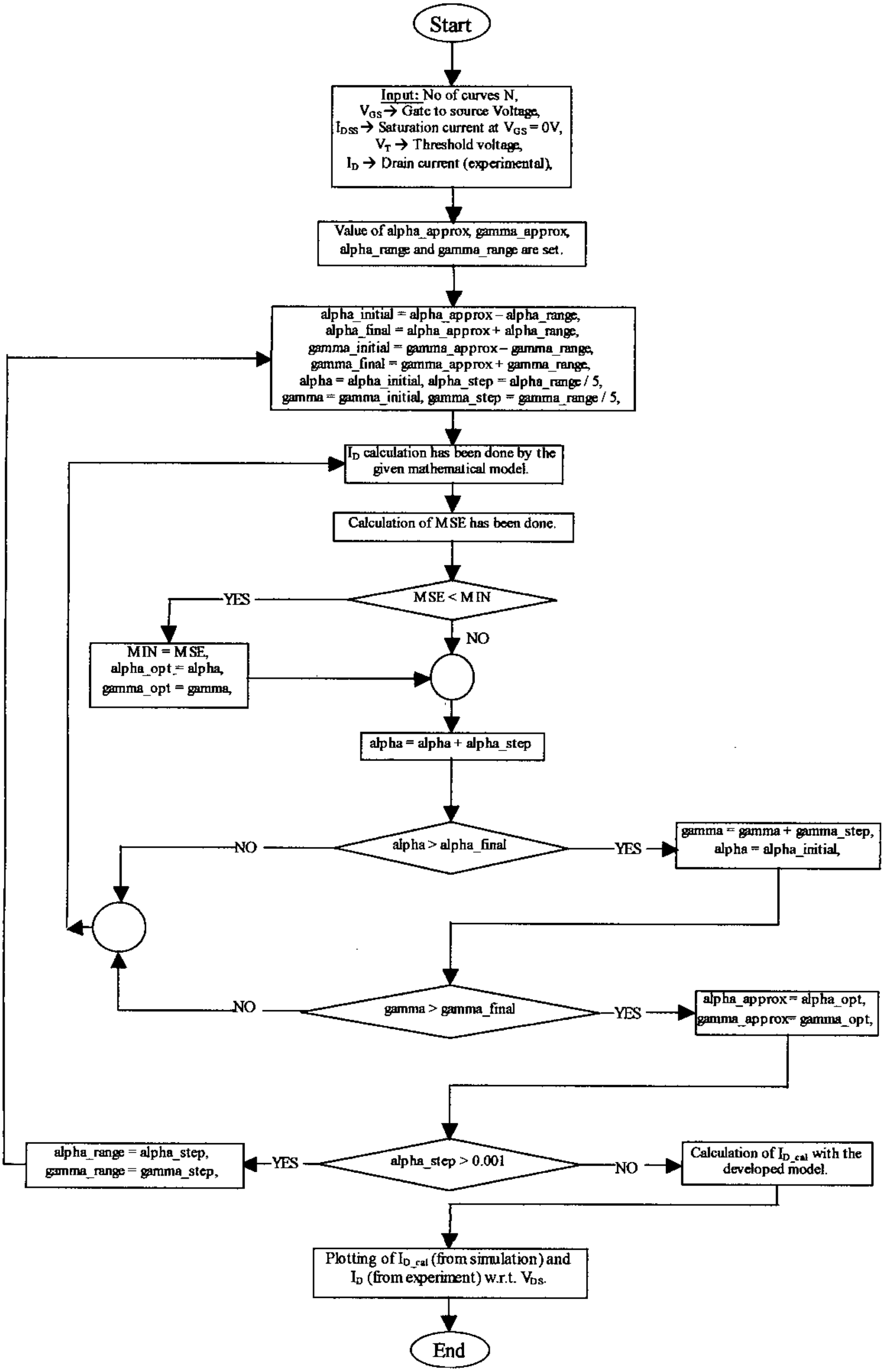

Start
Input: No of curves N,
V_{GS} → Gate to source Voltage,
I_{DSS} → Saturation current at V_{GS} = 0V,
V_T → Threshold voltage,
I_D → Drain current (experimental),
Value of alpha_approx, gamma_approx, alpha_range and gamma_range are set.
alpha_initial = alpha_approx – alpha_range,
alpha_final = alpha_approx + alpha_range,
gamma_initial = gamma_approx – gamma_range,
gamma_final = gamma_approx + gamma_range,
alpha = alpha_initial, alpha_step = alpha_range / 5,
gamma = gamma_initial, gamma_step = gamma_range / 5,
I_D calculation has been done by the given mathematical model.
Calculation of MSE has been done.
MSE < MIN
YES
NO
MIN = MSE,
alpha_opt = alpha,
gamma_opt = gamma,
alpha = alpha + alpha_step
alpha > alpha_final
NO
YES
gamma = gamma + gamma_step,
alpha = alpha_initial,
gamma > gamma_final
NO
YES
alpha_approx = alpha_opt,
gamma_approx = gamma_opt,
alpha_step > 0.001
YES
NO
alpha_range = alpha_step,
gamma_range = gamma_step,
Calculation of I_{D_cal} with the developed model.
Plotting of I_{D_cal} (from simulation) and I_D (from experiment) w.r.t. V_{DS}.
End

Figure 5.1: Flow chart for Materka model.

5.3 Algorithm Description

At first the approximate values of all empirical constants and the range of each constants are provided by the user. The range may be very high even in the order of hundreds. Then according to the approximate values and the ranges, the initial and the final values of constants are set. Initial value is obtained by subtracting the range from the approximate value and final value is obtained by summing the approximate value with the range. The whole is then divided into ten equal parts. Then with every combination of the empirical constants, I_{DS} is calculated from the mathematical model and MSE (mean square error) is calculated. The lowest value of the MSE is then considered and the new approximate values of the empirical constants are set by the corresponding values of the lowest MSE. Then the range is set as the value of step increase of the previous iteration. This procedure iterated as long as the constants get almost at a fixed value considering a certain decimal places as wanted. After that the experimental and the simulated curves are plotted.

5.4 Algorithm for FSIM Model

When the above-mentioned algorithm is again used with *FSIM* model for GaAs MESFET, it had to be modified so that it can operate for 3 empirical constants in place of 2. The algorithm process is shown in the flow-chart in Fig. 5.2, which is used to simulate the *FSIM* model in order to generate data on a PC.

5.5 Algorithm for SSIM Model

The algorithm process, which had been developed before for *FSIM* model, is designed to optimize 3 empirical constants. Now the *SSIM* model has 4 empirical constants. To simulate the new model, the algorithm has been modified as shown in the flow chart in Fig. 5.3.

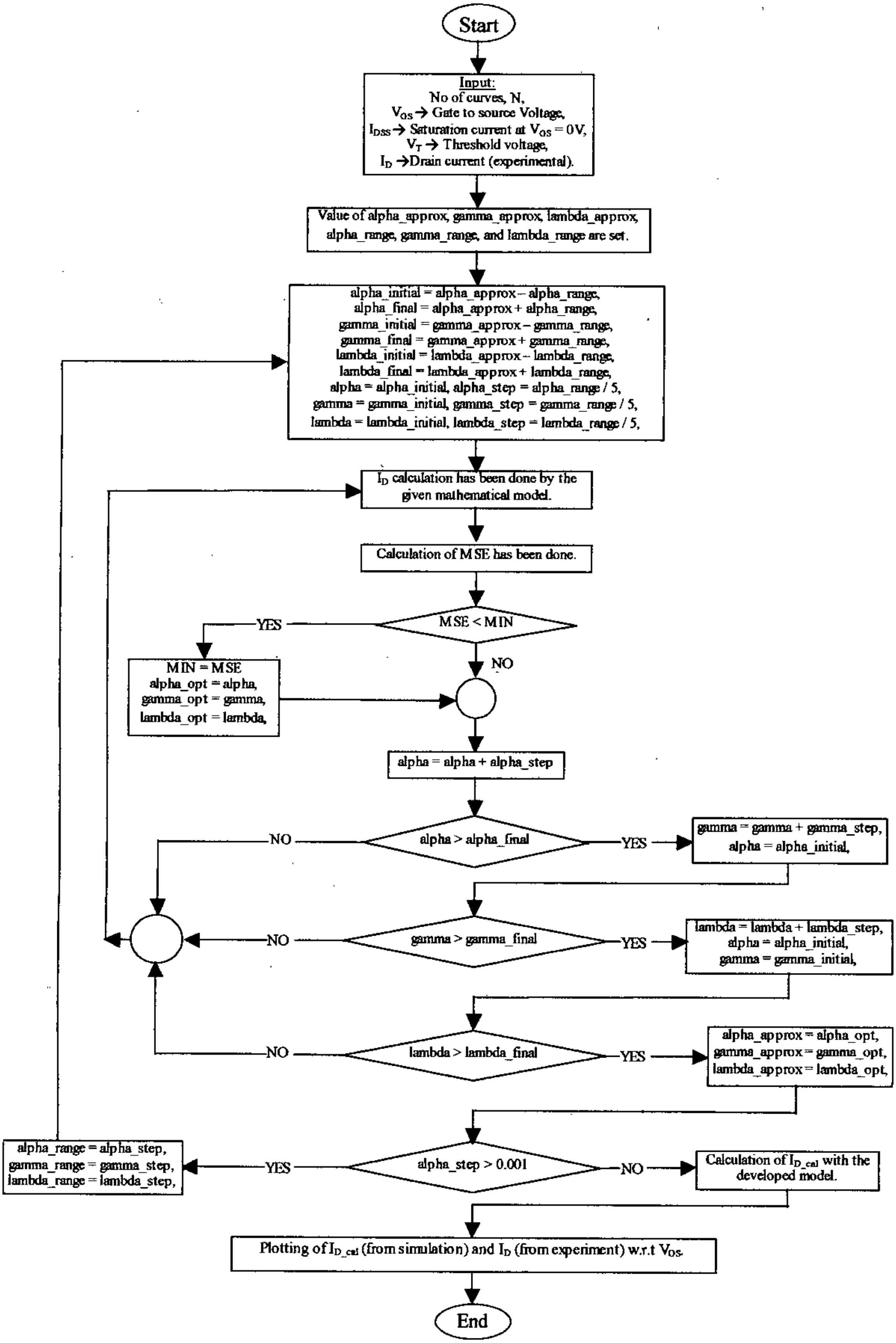

Start
Input:
No of curves, N,
V_OS → Gate to source Voltage,
I_DSS → Saturation current at V_OS = 0V,
V_T → Threshold voltage,
I_D → Drain current (experimental).
Value of alpha_approx, gamma_approx, lambda_approx, alpha_range, gamma_range, and lambda_range are set.
alpha_initial = alpha_approx – alpha_range,
alpha_final = alpha_approx + alpha_range,
gamma_initial = gamma_approx – gamma_range,
gamma_final = gamma_approx + gamma_range,
lambda_initial = lambda_approx – lambda_range,
lambda_final = lambda_approx + lambda_range,
alpha = alpha_initial, alpha_step = alpha_range / 5,
gamma = gamma_initial, gamma_step = gamma_range / 5,
lambda = lambda_initial, lambda_step = lambda_range / 5,
I_D calculation has been done by the given mathematical model.
Calculation of MSE has been done.
MSE < MIN
YES
NO
MIN = MSE
alpha_opt = alpha,
gamma_opt = gamma,
lambda_opt = lambda,
alpha = alpha + alpha_step
alpha > alpha_final
NO
YES
gamma = gamma + gamma_step,
alpha = alpha_initial,
gamma > gamma_final
NO
YES
lambda = lambda + lambda_step,
alpha = alpha_initial,
gamma = gamma_initial,
lambda > lambda_final
NO
YES
alpha_approx = alpha_opt,
gamma_approx = gamma_opt,
lambda_approx = lambda_opt,
alpha_step > 0.001
YES
NO
alpha_range = alpha_step,
gamma_range = gamma_step,
lambda_range = lambda_step,
Calculation of I_D_cal with the developed model.
Plotting of I_D_cal (from simulation) and I_D (from experiment) w.r.t V_OS.
End

Figure 5.2: Flow chart for *FSIM* model.

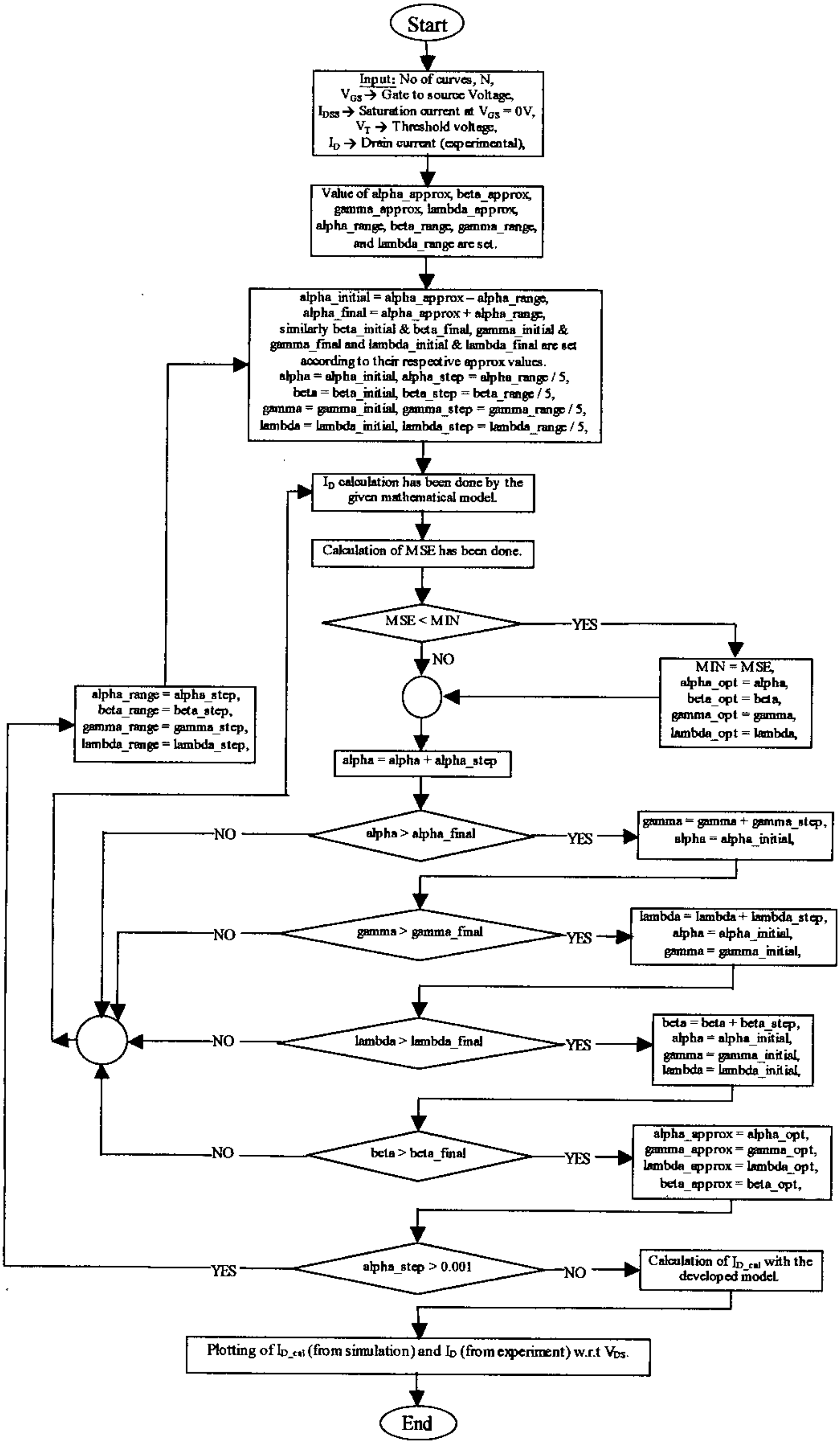

Start
Input: No of curves, N,
V_{GS} → Gate to source Voltage,
I_{DSS} → Saturation current at V_{GS} = 0V,
V_T → Threshold voltage,
I_D → Drain current (experimental),
Value of alpha_approx, beta_approx, gamma_approx, lambda_approx, alpha_range, beta_range, gamma_range, and lambda_range are set.
alpha_initial = alpha_approx – alpha_range,
alpha_final = alpha_approx + alpha_range,
similarly beta_initial & beta_final, gamma_initial & gamma_final and lambda_initial & lambda_final are set according to their respective approx values.
alpha = alpha_initial, alpha_step = alpha_range / 5,
beta = beta_initial, beta_step = beta_range / 5,
gamma = gamma_initial, gamma_step = gamma_range / 5,
lambda = lambda_initial, lambda_step = lambda_range / 5,
I_D calculation has been done by the given mathematical model.
Calculation of MSE has been done.
MSE < MIN
YES
NO
MIN = MSE,
alpha_opt = alpha,
beta_opt = beta,
gamma_opt = gamma,
lambda_opt = lambda,
alpha_range = alpha_step,
beta_range = beta_step,
gamma_range = gamma_step,
lambda_range = lambda_step,
alpha = alpha + alpha_step
NO
alpha > alpha_final
YES
gamma = gamma + gamma_step,
alpha = alpha_initial,
NO
gamma > gamma_final
YES
lambda = lambda + lambda_step,
alpha = alpha_initial,
gamma = gamma_initial,
NO
lambda > lambda_final
YES
beta = beta + beta_step,
alpha = alpha_initial,
gamma = gamma_initial,
lambda = lambda_initial,
NO
beta > beta_final
YES
alpha_approx = alpha_opt,
gamma_approx = gamma_opt,
lambda_approx = lambda_opt,
beta_approx = beta_opt,
YES
alpha_step > 0.001
NO
Calculation of I_{D_cal} with the developed model.
Plotting of I_{D_cal} (from simulation) and I_D (from experiment) w.r.t V_{DS}.
End

Figure 5.3: Flow chart for *SSIM* model.

CHAPTER VI

Chapter Six

DISCUSSIONS, CONCLUSIONS AND SUGGESTIONS

6.1 Discussions

In this section, the effects of the ranges of empirical constants and also the effects for the long or short range of drain-source voltage are discussed.

6.1.1 Range of Empirical Constants

In the new developed algorithm, the values of empirical constants are chosen between the maximum and the minimum values. These maximum and the minimum values are calculated from the approximate value and range of each constant. The user supplies both these approximate values and the ranges. So if the accurate values of the empirical constants of a GaAs MESFET do not stay within the specified range of the approximate value, then the values of empirical constants obtained become erroneous.

6.1.2 Range of Drain-Source Voltage

The result of the simulation of numerical model to find out empirical constants depends on the range of V_{DS}. If V_{DS} range is large, the *I-V* characteristics have a long part after saturation. In that case, the effect of this post saturation part on simulation is strong. As a result the curve at pre-saturation part do not agree with the measured curve much. Again if the post-saturation curve is small (V_{DS} range is small), the curve at pre-saturation region becomes almost same as the measured curve but the nature of post-saturation region does not obey the measured data much. So, it is a good practice to take a moderate value of V_{DS} range across which simulation is done.

6.2 Conclusions

Introduction of the two new empirical constants to the Kacprzak-Materka model have increased the time of simulation, but still the performance has been improved. In the modern world of digital computers, the speed of microcomputer is increasing rapidly. Now, where anyone can have a personal computer of *1 GHz* speed, time taken to simulate this program may be overlooked comparing with the improvement of error by designing the new dc numerical model of GaAs MESFETs.

If execution time of the algorithm is the main factor for any case then the accuracy of the algorithm may be easily changed. We have also shown these relationships between the accuracy and the execution time for various models in Table 6.1.

We have further designed a new algorithm to reduce the execution time drastically. In this research this algorithm proves its suitability but it may be erroneous for other cases due to its poor combination procedure of the empirical constants.

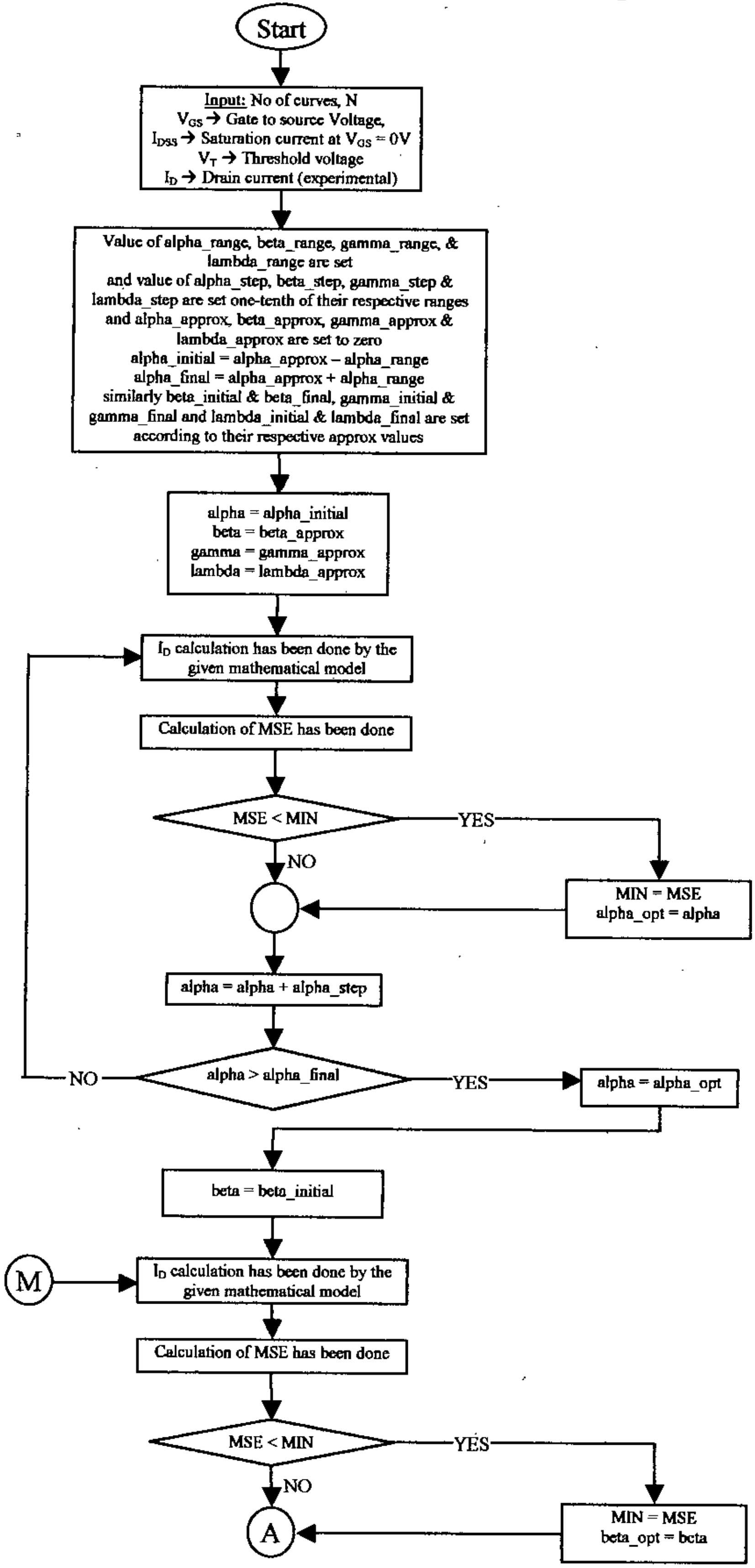
Table 6.1: Comparison of required execution time for all the algorithms proposed in this research.

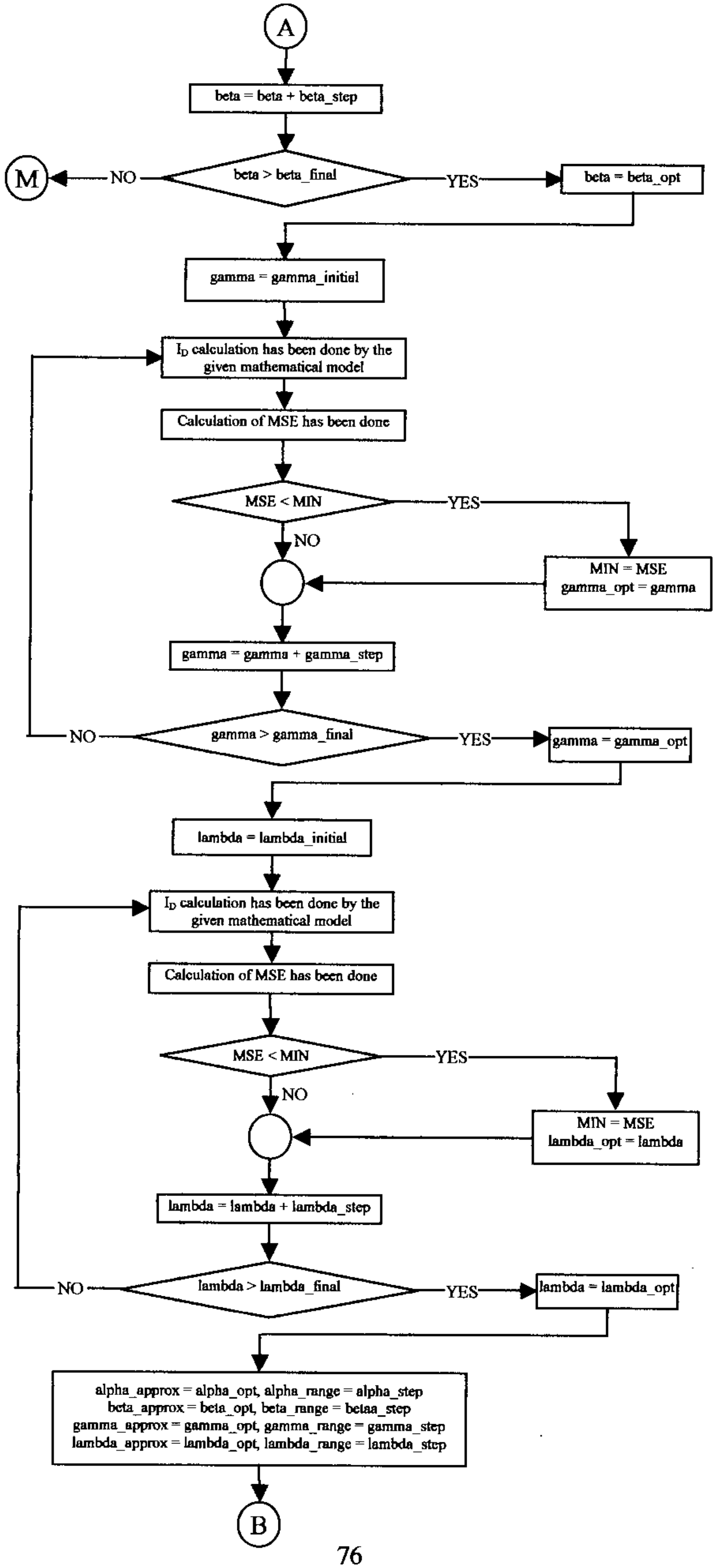
Accuracy	Execution Time (min:sec)				Algorithm
	Materka Model	FSIM Model	SSIM Model	TSIM Model	
0.001	00:02.09	00:21.13	04:07.49	04:07.49	Used in this research
0.01	00:01.49	00:14.72	02:45.95	02:45.95	
0.001	Instant	00:14.25	02:39.33	02:39.33	New Reduced-Time Algorithm
0.01	Instant	00:10.40	02:02.40	02:02.40	

The execution time in the table means that the time required to extract the values of the empirical constants. The time mentioned above is obtained by using a personal computer having a μ-processor of *350 MHz*. The results so far mentioned in this research are obtained using the previous algorithms. The *New Reduced-Time Algorithm* is mentioned here only for future research and for further improvement.

Analytical model for non-linear *I-V* characteristics of submicron GaAs MESFETs has been developed in this research. The developed model exhibits excellent accuracy with the published experimental results. A complete algorithm has also been developed to determine the empirical constants of the model. An improved technique has been employed to determine the accuracy of the model. The model can easily be implemented in programs of computer aided analysis and design of circuits with sub-micron GaAs MESFETs.

6.2.1 Flow Chart for New Reduced-Time Algorithm


Start
Input: No of curves, N
V_{GS} → Gate to source Voltage,
I_{DSS} → Saturation current at V_{GS} = 0V
V_T → Threshold voltage
I_D → Drain current (experimental)
Value of alpha_range, beta_range, gamma_range, & lambda_range are set
and value of alpha_step, beta_step, gamma_step & lambda_step are set one-tenth of their respective ranges
and alpha_approx, beta_approx, gamma_approx & lambda_approx are set to zero
alpha_initial = alpha_approx – alpha_range
alpha_final = alpha_approx + alpha_range
similarly beta_initial & beta_final, gamma_initial & gamma_final and lambda_initial & lambda_final are set according to their respective approx values
alpha = alpha_initial
beta = beta_approx
gamma = gamma_approx
lambda = lambda_approx
I_D calculation has been done by the given mathematical model
Calculation of MSE has been done
MSE < MIN
YES
NO
MIN = MSE
alpha_opt = alpha
alpha = alpha + alpha_step
alpha > alpha_final
NO
YES
alpha = alpha_opt
beta = beta_initial
M
I_D calculation has been done by the given mathematical model
Calculation of MSE has been done
MSE < MIN
YES
NO
MIN = MSE
beta_opt = beta
A


A
beta = beta + beta_step
beta > beta_final
NO
M
YES
beta = beta_opt
gamma = gamma_initial
I_D calculation has been done by the given mathematical model
Calculation of MSE has been done
MSE < MIN
YES
NO
MIN = MSE
gamma_opt = gamma
gamma = gamma + gamma_step
gamma > gamma_final
NO
YES
gamma = gamma_opt
lambda = lambda_initial
I_D calculation has been done by the given mathematical model
Calculation of MSE has been done
MSE < MIN
YES
NO
MIN = MSE
lambda_opt = lambda
lambda = lambda + lambda_step
lambda > lambda_final
NO
YES
lambda = lambda_opt
alpha_approx = alpha_opt, alpha_range = alpha_step
beta_approx = beta_opt, beta_range = betaa_step
gamma_approx = gamma_opt, gamma_range = gamma_step
lambda_approx = lambda_opt, lambda_range = lambda_step
B

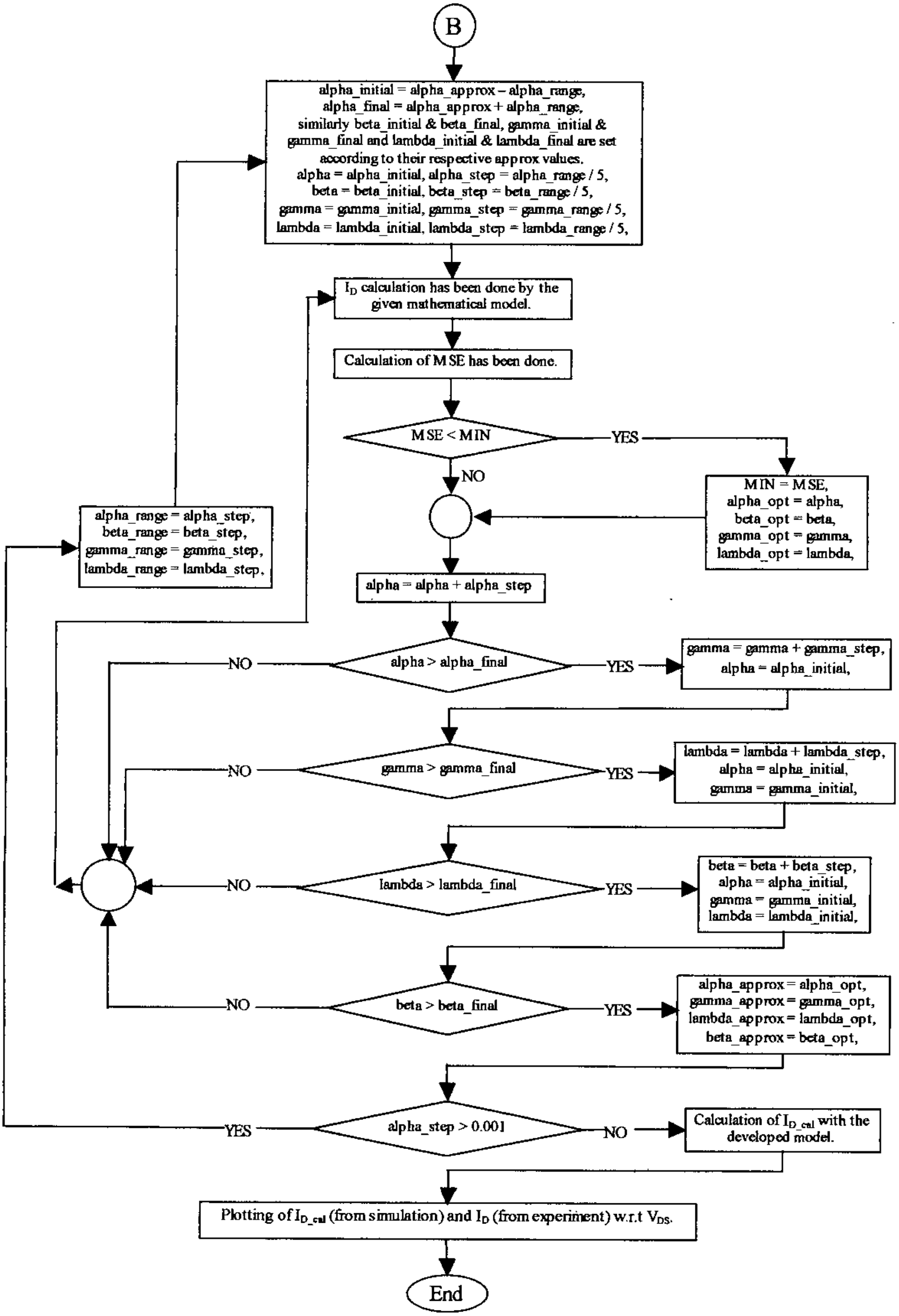

B
alpha_initial = alpha_approx – alpha_range,
alpha_final = alpha_approx + alpha_range,
similarly beta_initial & beta_final, gamma_initial &
gamma_final and lambda_initial & lambda_final are set
according to their respective approx values.
alpha = alpha_initial, alpha_step = alpha_range / 5,
beta = beta_initial, beta_step = beta_range / 5,
gamma = gamma_initial, gamma_step = gamma_range / 5,
lambda = lambda_initial, lambda_step = lambda_range / 5,
I$_D$ calculation has been done by the given mathematical model.
Calculation of MSE has been done.
MSE < MIN
YES
NO
MIN = MSE,
alpha_opt = alpha,
beta_opt = beta,
gamma_opt = gamma,
lambda_opt = lambda,
alpha_range = alpha_step,
beta_range = beta_step,
gamma_range = gamma_step,
lambda_range = lambda_step,
alpha = alpha + alpha_step
NO
alpha > alpha_final
YES
gamma = gamma + gamma_step,
alpha = alpha_initial,
NO
gamma > gamma_final
YES
lambda = lambda + lambda_step,
alpha = alpha_initial,
gamma = gamma_initial,
NO
lambda > lambda_final
YES
beta = beta + beta_step,
alpha = alpha_initial,
gamma = gamma_initial,
lambda = lambda_initial,
NO
beta > beta_final
YES
alpha_approx = alpha_opt,
gamma_approx = gamma_opt,
lambda_approx = lambda_opt,
beta_approx = beta_opt,
YES
alpha_step > 0.001
NO
Calculation of I$_{D\_cal}$ with the developed model.
Plotting of I$_{D\_cal}$ (from simulation) and I$_D$ (from experiment) w.r.t V$_{DS}$.
End

Figure 6.1: Flow chart for New Reduced-Time algorithm.

6.3 Suggestions for Future Research

Although the performance has increased after introducing the new model, there are still some errors, which can be further minimized. So, the following suggestions are made for those who want to involve themselves with the development of numerical model of GaAs MESFETs.

6.3.1 Adding More Effects for Threshold Voltage Shift

To reduce the number of parameters in the final model (i.e. *TSIM* Model), we have calculated threshold voltage, V_T considering only the short gate length effect. But we have discussed before that with the addition of this effect the total threshold voltage, V_T is still have a greater discrepancy from the observed threshold voltage. So the calculated threshold voltage must have ignored some device parameters. Among them, interface states are very important because the surface state density is an uncontrollable parameter as reported in [19]. If extra consideration, in this respect could be added with the proposed model then more accurate model can be developed.

6.3.2 Introduction of Nonlinear Effect on Bias Voltages

The output conductance does not vary with the variation of drain-to-source voltage, V_{DS} and gate-to-source voltage, V_{GS} linearly. So, the output conductance term of the numerical model $(1+ \lambda V_{DS} + \beta V_{GS})$ may be modified to form a better model. For this our suggestion is to introduce higher powers to V_{DS} and V_{GS}.

6.3.3 Taking into Account Pre and Post Saturation

The dependency of output conductance on gate-to-source voltage, V_{GS} before and after saturation is different. So, if this phenomenon can be introduced mathematically then there is a possibility that a better numerical model will be arisen.

References

[1] Tomasz Kacprzak and Andrzej Materka, "Compact DC Model of GaAs MESFETs for Large Signal Computer Calculations," *IEEE Journal of Solid-State Circuits,* Vol. SC. - 18, No. 2, pp. 211-213, April 1983.

[2] P. H. Ladbrooke, *MMIC Design: GaAs FETs & HEMTs,* Boston and London, Artech House, 1989.

[3] J. M. Golio, *Microwave MESFETs & HEMTs,* Boston M. A.: Artech House, 1991.

[4] Walter R. Curtice, "A MESFET Model for Use in the Design of GaAs Integrated Circuits," *IEEE Transactions on Microwave Theory and Technique*, Vol. MTT - 28, No. 5, pp. 448 - 456, May 1980.

[5] W. R. Curtice and M. Etterberg, "A Nonlinear GaAs FET for Use in the Design of Output Circuits for Power Amplifiers," *IEEE Transactions on Microwave Theory and Technique,* Vol. MTT-33, pp. 1383-1388, 1985.

[6] H. Statz, P. Newman, I. W. Smith, R. A. Pucel and H. A. Haus, "GaAs FET Device and Circuit Simulation in SPICE," *IEEE Transactions on Electron Devices,* Vol. 34, pp. 160-169, 1987.

[7] T. J. Rodriguez and P. J. England, "A Five Parameter DC GaAs MESFET Model for Nonlinear Circuit Design," *Proc. Inst. Elec. Eng.*, Vol. 139, pp. 325-332, 1992.

[8] M. S. Shur, "Analytical Model of GaAs MESFETs," *IEEE Transactions on Electron Devices,* Vol. ED - 25, pp. 612-618, June 1978.

[9] H. Shichman and D. A. Hodges, "Modeling and Simulation of Insulated Gate Field Effect Transistor Switching Circuits," *IEEE Journal of Solid-State Circuits,* Vol. SC-3, pp. 285-289, Sept. 1968.

[10] B. Himsworth, "A Two-Dimensional Analysis of Gallium Arsenide Junction Field-Effect Transistors with Long and Short Channels," *Solid-State Electron,* Vol. 15, pp. 1353-1361, 1972.

[11] R. W. H. Engelmann and C. A. Liechti, "Gunn Domain Formation in the Saturated Current Region of GaAs MESFETs," *IEDM Tech. Dig,* pp. 351-354, Dec. 1976.

[12] K. Yamaguchi, S. Asai and H. Kodera, "Two-Dimensional Numerical Analysis of Stability Criteria of GaAs FETs," *IEEE Transactions on Electron Devices,* Vol. ED-23, pp. 1283-1290, 1976.

[13] T. Taki, "Approximation of Junction Field-Effect Transistor Characteristics by a Hyperbolic Function," *IEEE Journal of Solid-State Circuits,* Vol. SC-13, pp. 724-726, Oct. 1978.

[14] T. J. Rodriguez, M. Al-Dass and K. A. Mezher, "Comparison of Nonlinear MESFET Model for Wide Band Circuit Design," *IEEE Transactions on Electron Devices,* Vol. 41, pp. 288-296, 1994.

[15] J. A. Nedler, "A Simplex Model for Function Minimization," *The Computer Journal,* Vol. 7, pp. 308, Mar. 1964.

[16] Takatomo Enoki, Suehiro Sugitani and Yasuro Yamane, "Characteristics Including Electron Velocity Overshoot for 0.1-µm Gate Length GaAs SAINT MESFETs," *IEEE Transactions on Electron Devices,* Vol. 37. No. 4, pp. 935-941, April 1990.

[17] G. W. Taylor, "The Effects of Two Dimensional Charge Sharing on the Above-Threshold Characteristics of Short-Channel IGFETs," *Solid - State Electronics,* Vol. 22, No. 8-B, pp. 701-717, 1979.

[18] Mansoor M. Ahmed, Haroon Ahmed, and Peter H. Ladbrooke, "Effects of Interface States on Sub-micron GaAs Metal-Semiconductors Field-Effect Transistors Assessed by Gate Leakage Current," *Journal of Vacuum Science Technology B,* Vol. 13, pp. 1519-1525, 1995.

[19] Mansoor M. Ahmed, and Haroon Ahmed, "Novel Electron Beam Lithography Technique for Sub-micron T-Gates Fabrications," *Journal of Vacuum Science Technology B,* Vol. 15, pp. 306-310, 1997.

[20] L. F. Eastman and M. S. Shur, "Substrate Current in GaAs MESFETs," *IEEE Transactions on Electron Devices,* Vol. ED-26, No. 9, pp. 1359-1361, 1979.

[21] Mansoor M. Ahmed, "Effects of active Channel thickness on sub micron GaAs MESFETs characteristics," *Journal of Vacuum Science Technology B,* Vol. 16, pp. 968-971, 1998.

APPENDIX

PROGRAMS

Program of Materka Model for Device A-74-1

```
#include <stdio.h>
#include <conio.h>
#include <math.h>


#define Idss 192.0
#define VT -1.99999
#define N 5 // Number of curve to be simulated.


void main(void)
{
   int i, j, n;
   double Vgs[N+1],Id[N+1][31], gm[36], gd[36], Vds[31], Vgss[36];
   double Id_cal[N+1][31], gm_cal[36], gd_cal[36];
   double Vds_step;
   double MSE[N+1], MIN = 1e10;
   double obsRMS[N+1], calRMS[N+1];
   double optalpha, optgamma;
   double alpha, gamma;
   double alpha_approx,  alpha_range,  alpha_initial,  alpha_final,  alpha_step;
   double gamma_approx, gamma_range, gamma_initial, gamma_final, gamma_step;


   Vds_step = 0.1;

   Vgs[1] =  0.0;
   Vgs[2] = -0.5;
   Vgs[3] = -1.0;
   Vgs[4] = -1.5;
   Vgs[5] = -2.0;


   for ( j = 0; j < 31 ;j++)
      Vds[j] = j* Vds_step;
   Id[1][0]  = 0.00,        Id[2][0]  = 0.00,        Id[3][0]  = 0.00;
   Id[1][1]  = 34.52,    Id[2][1]  = 21.71,    Id[3][1]  = 9.47;
   Id[1][2]  = 64.03,    Id[2][2]  = 38.98,    Id[3][2]  = 15.59;
   Id[1][3]  = 91.31,    Id[2][3]  = 56.24,    Id[3][3]  = 22.27;
   Id[1][4]  = 118.04,      Id[2][4]  = 71.27,      Id[3][4]  = 26.73;
   Id[1][5]  = 139.76,      Id[2][5]  = 80.73,      Id[3][5]  = 30.07;
```

```
Id[1][6]  = 158.69,
Id[1][7]  = 173.72,
Id[1][8]  = 183.74,
Id[1][9]  = 190.98,
Id[1][10] = 197.10,
Id[1][11] = 201.00,
Id[1][12] = 204.90,
Id[1][13] = 208.24,
Id[1][14] = 211.02,
Id[1][15] = 213.25,
Id[1][16] = 215.48,
Id[1][17] = 217.15,
Id[1][18] = 219.38,
Id[1][19] = 221.60,
Id[1][20] = 223.27,
Id[1][21] = 224.94,
Id[1][22] = 226.61,
Id[1][23] = 227.73,
Id[1][24] = 229.40,
Id[1][25] = 231.07,
Id[1][26] = 232.18,
Id[1][27] = 233.85,
Id[1][28] = 234.97,
Id[1][29] = 236.08,
Id[1][30] = 237.19,

Id[2][6]  = 88.53,
Id[2][7]  = 94.65,
Id[2][8]  = 99.11,
Id[2][9]  = 102.45,
Id[2][10] = 105.79,
Id[2][11] = 108.57,
Id[2][12] = 111.36,
Id[2][13] = 113.59,
Id[2][14] = 115.81,
Id[2][15] = 118.04,
Id[2][16] = 120.27,
Id[2][17] = 122.49,
Id[2][18] = 124.16,
Id[2][19] = 126.39,
Id[2][20] = 128.06,
Id[2][21] = 129.73,
Id[2][22] = 131.40,
Id[2][23] = 133.07,
Id[2][24] = 135.30,
Id[2][25] = 136.97,
Id[2][26] = 138.64,
Id[2][27] = 139.76,
Id[2][28] = 141.43,
Id[2][29] = 143.10,
Id[2][30] = 144.77,

Id[3][6]  = 33.41;
Id[3][7]  = 35.63;
Id[3][8]  = 37.86;
   Id[3][9]  = 40.09;
   Id[3][10] = 41.76;
   Id[3][11] = 43.99;
   Id[3][12] = 45.66;
   Id[3][13] = 47.33;
   Id[3][14] = 49.55;
   Id[3][15] = 50.67;
   Id[3][16] = 52.90;
   Id[3][17] = 54.01;
   Id[3][18] = 55.68;
   Id[3][19] = 57.35;
   Id[3][20] = 59.02;
   Id[3][21] = 61.25;
   Id[3][22] = 62.92;
   Id[3][23] = 64.03;
   Id[3][24] = 65.70;
   Id[3][25] = 67.37;
   Id[3][26] = 69.04;
   Id[3][27] = 70.71;
   Id[3][28] = 72.38;
   Id[3][29] = 74.61;
   Id[3][30] = 76.28;


Id[4][0]  = 0.00,
Id[4][1]  = 1.00,
Id[4][2]  = 2.00,
Id[4][3]  = 3.00,
Id[4][4]  = 4.00,
Id[4][5]  = 5.01,
Id[4][6]  = 6.01,
Id[4][7]  = 7.01,
Id[4][8]  = 8.01,
Id[4][9]  = 9.01,
Id[4][10] = 10.02,
Id[4][11] = 10.08,
Id[4][12] = 11.58,
Id[4][13] = 12.36,
Id[4][14] = 13.14,
Id[4][15] = 13.92,
Id[4][16] = 15.03,
Id[4][17] = 16.15,
Id[4][18] = 17.26,
Id[4][19] = 18.38,

Id[5][0]  = 0.00;
Id[5][1]  = 0.15;
Id[5][2]  = 0.30;
Id[5][3]  = 0.45;
Id[5][4]  = 0.59;
Id[5][5]  = 0.74;
Id[5][6]  = 0.89;
Id[5][7]  = 1.04;
Id[5][8]  = 1.19;
Id[5][9]  = 1.34;
Id[5][10] = 1.49;
Id[5][11] = 1.64;
Id[5][12] = 1.78;
Id[5][13] = 1.93;
Id[5][14] = 2.08;
Id[5][15] = 2.23;
Id[5][16] = 2.56;
Id[5][17] = 2.90;
Id[5][18] = 3.23;
Id[5][19] = 3.57;
```

```
    Id[4][20] = 19.49,          Id[5][20] = 3.90;
    Id[4][21] = 20.71,          Id[5][21] = 4.57;
    Id[4][22] = 21.94,          Id[5][22] = 5.24;
    Id[4][23] = 23.16,          Id[5][23] = 5.90;
    Id[4][24] = 24.39,          Id[5][24] = 6.57;
    Id[4][25] = 25.61,          Id[5][25] = 7.24;
    Id[4][26] = 27.06,          Id[5][26] = 8.24;
    Id[4][27] = 28.51,          Id[5][27] = 9.24;
    Id[4][28] = 29.95,          Id[5][28] = 10.25;
    Id[4][29] = 31.40,          Id[5][29] = 11.25;
    Id[4][30] = 32.85,          Id[5][30] = 12.25;


    alpha_approx  = 0.0;
    alpha_range   = 5.0;

    gamma_approx  = 0.0;
    gamma_range   = 5.0;

looping:
    alpha_initial  = alpha_approx - alpha_range;
    alpha_final    = alpha_approx + alpha_range;
    alpha_step     = alpha_range  / 5.0;

    gamma_initial  = gamma_approx - gamma_range;
    gamma_final    = gamma_approx + gamma_range;
    gamma_step     = gamma_range  / 5.0;

    for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
       for  (  gamma  =  gamma_initial;  gamma  <=  gamma_final;  gamma  +=
gamma_step)
       {
          MSE[0] = 0.0;
          for (i = 1; i <= N; i++)
             for (j = 0; j < 31; j++)
             {
                Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) *
tanh (alpha * Vds[j] / ( Vgs[i] - VT - gamma * Vds[j]));

                MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
             }
             if (MSE[0] < MIN && MSE[0] > 0.0)
             {
                MIN = MSE[0];
                optalpha  = alpha;
                optgamma  = gamma;
             }
```

```
            printf ("alpha : %0.4f\n", alpha);
            printf ("gamma : %0.4f\n", gamma);
            printf ("MSE   : %0.4f\n", MSE[0]);
            printf ("MIN   : %0.4f\n", MIN);
        }
    alpha_approx = optalpha;
    alpha_range  = alpha_step;

    gamma_approx = optgamma;
    gamma_range  = gamma_step;

    if (alpha_step > 0.001) goto looping;


//plotting:

    printf ("\nOptimized value of constants\n");
    printf ("--------------------------------\n");
    printf ("alpha : %0.4f\n", optalpha);
    printf ("gamma : %0.4f\n", optgamma);

    getch();

    for (i = 0; i <= N ; i++)
    {
        obsRMS[i] = 0.0;
        calRMS[i] = 0.0;
        MSE[i]    = 0.0;
    }

    for (i = 1 ; i <= N ; i++)
    {
        for (j = 0; j < 31; j++)
        {
            Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / (Vgs[i] - VT - optgamma * Vds[j]));

            MSE[0]    += pow ((Id[i][j] - Id_cal[i][j]), 2);

            MSE[i]    += pow ((Id[i][j] - Id_cal[i][j]), 2);
            obsRMS[i] += pow (Id[i][j], 2);
            calRMS[i] += pow (Id_cal[i][j], 2);
            obsRMS[0] += pow (Id[i][j], 2);
            calRMS[0] += pow (Id_cal[i][j], 2);
        }

        MSE[i]    = sqrt ( MSE[i]   / 31.0);
        obsRMS[i]  = sqrt (obsRMS[i] / 31.0);
        calRMS[i]  = sqrt (calRMS[i] / 31.0);
    }

    MSE[0]    = sqrt ( MSE[0]    / ( N * 31.0));

    obsRMS[0] = sqrt ( obsRMS[0] / ( N * 31.0 ) );
    calRMS[0] = sqrt ( calRMS[0] / ( N * 31.0 ) );

    printf ("MSE calculated   : %0.4f\n", MSE[0]);
    printf ("Average RMS error: %0.4f\n", fabs ( calRMS[0] - obsRMS[0] ) );

    Vgss[0] = -3.0;

    for( i = 1 ; i < 36; i++)
        Vgss[i] = Vgss[i - 1] + 0.1;
    for( i = 1; i <= N ; i++ )
    {
        for( j = 0; j < 36; j++)
        {
            if((Vgss[j] - Vgs[i]) < Vds_step)
                n = j;
        }

        gm[n] = (Id[i][20] - Id[i+1][20])/(Vgs[i] - Vgs[i+1]);//20 is used for Vds = 2 v.
    }
    j = 20;
    for( i = 0; i < 36 ; i++ )
        gm_cal[i] = Idss *( 2 * ( 1 - Vgss[i] / (VT + optgamma * Vds[j])) * (- 1  / (VT +
optgamma * Vds[j])) * tanh (optalpha * Vds[j] / (Vgss[i] - VT - optgamma * Vds[j]))
+ pow((1 - Vgss[i] / (VT + optgamma * Vds[j])), 2) *  (1 - pow(tanh ( optalpha *
Vds[j] / (Vgss[i] - VT - optgamma * Vds[j])), 2)) * ( optalpha * (- Vds[j]) /
pow((Vgss[i] - VT - optgamma * Vds[j]), 2)));


    i = 1;
    for ( j = 0; j < 31; j++)
    {
        gd[j+1]   = (Id[i][j+2]-Id[i][j])/(Vds_step*2);

        gd_cal[j] = Idss * (2 *( 1 - Vgs[i] / (VT + optgamma * Vds[j])) * (optgamma *
Vgs[i] / pow((VT + optgamma * Vds[j]), 2)) *  tanh (optalpha * Vds[j] / (Vgs[i] - VT
- optgamma  * Vds[j]))  + pow((1 - Vgs[i] / (VT + optgamma * Vds[j])), 2) *  (1 -
pow(tanh ( optalpha * Vds[j] / (Vgs[i] - VT - optgamma * Vds[j])), 2)) * ( optalpha *
(Vgs[i] - VT - optgamma * Vds[j]) + (optalpha * optgamma * Vds[j]))/ pow((Vgs[i] -
VT - gamma * Vds[j]), 2));
```

```
    }

    FILE *fp;
    fp = fopen("optimal.xls", "w");

    fprintf (fp,"\t\t\tIdss =%6.3f mA / mm\n\t\t\tVT = %6.3fV\n\n",Idss, VT);
    fprintf (fp,"\t\tMSE Calculated = %0.4f\t\t\tAv. RMS Error = %0.4f\n\n", MSE[0],
fabs ( calRMS[0] - obsRMS[0] ) );

    fprintf (fp,"\t\tMSE[1]\tMSE[2]\tMSE[3]\tMSE[4]\tMSE[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",MSE[1], MSE[2], MSE[3],
MSE[4], MSE[5]);

    fprintf (fp,"\t\tobsRMS[1]\tobsRMS[2]\tobsRMS[3]\tobsRMS[4]\tobsRMS[5]\n");
    fprintf   (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",obsRMS[1],   obsRMS[2],
obsRMS[3], obsRMS[4], obsRMS[5]);

    fprintf (fp,"\t\tcalRMS[1]\tcalRMS[2]\tcalRMS[3]\tcalRMS[4]\tcalRMS[5]\n");
    fprintf   (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",calRMS[1],   calRMS[2],
calRMS[3], calRMS[4], calRMS[5]);

    fprintf (fp,"\t\tError1\tError2\tError3\tError4\tError5\n");
    fprintf     (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",     fabs(calRMS[1]    -
obsRMS[1]),  fabs(calRMS[2]  -  obsRMS[2]),  fabs(calRMS[3]  -  obsRMS[3]),
fabs(calRMS[4] - obsRMS[4]), fabs(calRMS[5] - obsRMS[5]));

    fprintf (fp,"\t\tAlpha = %0.4f\t\tGamma  = %0.4f\n\n",optalpha, optgamma);

    fprintf
(fp,"Vds\tIds1(meas.)\tIds1(cal.)\tIds2(meas.)\tds2(cal.)\tIds3(meas.)\tIds3(cal.)\tIds4(
meas.)\tIds4(cal.)\tIds5(meas.)\tIds5(cal.)\n");
    for ( j = 0; j < 31; j++)
       fprintf(fp,"
%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t\n",
Vds[j],Id[1][j],Id_cal[1][j],Id[2][j],Id_cal[2][j],Id[3][j],Id_cal[3][j],Id[4][j],Id_cal[4][j
],Id[5][j],Id_cal[5][j]);
    fprintf (fp,"\nVgss\tgm(meas.)\tgm(cal.)\n");
    for ( j = 0; j < 36; j++)
       fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vgss[j], gm[j], gm_cal[j]);
    fprintf (fp,"\nVds\tgd(meas.)\tgd(cal.)\n");
    for ( j = 0; j < 31; j++)
       fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vds[j], gd[j], gd_cal[j]);

    fclose(fp);

    getch();
}
```

Program of FSIM Model for Device A-74-1

```
#include <stdio.h>
#include <conio.h>
#include <math.h>

#define Idss 192.0
#define VT -1.99999
#define N 5 // Number of curve to be simulated.


void main(void)
{
   int i, j, n;
   double Vgs[N+1],Id[N+1][31], gm[36], gd[36], Vds[31], Vgss[36];
   double Id_cal[N+1][31], gm_cal[36], gd_cal[36];
   double Vds_step;
   double MSE[N+1], MIN = 1e10;
   double obsRMS[N+1], calRMS[N+1];
   double optalpha, optgamma, optlambda;
   double alpha, gamma, lambda;
   double alpha_approx, alpha_range, alpha_initial, alpha_final, alpha_step;
   double   gamma_approx,    gamma_range,    gamma_initial,    gamma_final,
gamma_step;
   double lambda_approx, lambda_range, lambda_initial, lambda_final, lambda_step;

   Vds_step = 0.1;

   Vgs[1] =  0.0;
   Vgs[2] = -0.5;
   Vgs[3] = -1.0;
   Vgs[4] = -1.5;
   Vgs[5] = -2.0;

   for ( j = 0; j < 31 ;j++)
      Vds[j] = j* Vds_step;
   Id[1][0]  = 0.00,        Id[2][0]  = 0.00,          Id[3][0]  = 0.00;
   Id[1][1]  = 34.52,     Id[2][1]  = 21.71,      Id[3][1]  = 9.47;
   Id[1][2]  = 64.03,     Id[2][2]  = 38.98,      Id[3][2]  = 15.59;
   Id[1][3]  = 91.31,     Id[2][3]  = 56.24,      Id[3][3]  = 22.27;
   Id[1][4]  = 118.04,      Id[2][4]  = 71.27,      Id[3][4]  = 26.73;
   Id[1][5]  = 139.76,      Id[2][5]  = 80.73,      Id[3][5]  = 30.07;
   Id[1][6]  = 158.69,      Id[2][6]  = 88.53,      Id[3][6]  = 33.41;
   Id[1][7]  = 173.72,      Id[2][7]  = 94.65,      Id[3][7]  = 35.63;
   Id[1][8]  = 183.74,      Id[2][8]  = 99.11,      Id[3][8]  = 37.86;
   Id[1][9]  = 190.98,      Id[2][9]  = 102.45,        Id[3][9]  = 40.09;
   Id[1][10] = 197.10,      Id[2][10] = 105.79,        Id[3][10] = 41.76;
```

```
Id[1][11] = 201.00,     Id[2][11] = 108.57,     Id[3][11] = 43.99;
Id[1][12] = 204.90,     Id[2][12] = 111.36,     Id[3][12] = 45.66;
Id[1][13] = 208.24,     Id[2][13] = 113.59,     Id[3][13] = 47.33;
Id[1][14] = 211.02,     Id[2][14] = 115.81,     Id[3][14] = 49.55;
Id[1][15] = 213.25,     Id[2][15] = 118.04,     Id[3][15] = 50.67;
Id[1][16] = 215.48,     Id[2][16] = 120.27,     Id[3][16] = 52.90;
Id[1][17] = 217.15,     Id[2][17] = 122.49,     Id[3][17] = 54.01;
Id[1][18] = 219.38,     Id[2][18] = 124.16,     Id[3][18] = 55.68;
Id[1][19] = 221.60,     Id[2][19] = 126.39,     Id[3][19] = 57.35;
Id[1][20] = 223.27,     Id[2][20] = 128.06,     Id[3][20] = 59.02;
Id[1][21] = 224.94,     Id[2][21] = 129.73,     Id[3][21] = 61.25;
Id[1][22] = 226.61,     Id[2][22] = 131.40,     Id[3][22] = 62.92;
Id[1][23] = 227.73,     Id[2][23] = 133.07,     Id[3][23] = 64.03;
Id[1][24] = 229.40,     Id[2][24] = 135.30,     Id[3][24] = 65.70;
Id[1][25] = 231.07,     Id[2][25] = 136.97,     Id[3][25] = 67.37;
Id[1][26] = 232.18,     Id[2][26] = 138.64,     Id[3][26] = 69.04;
Id[1][27] = 233.85,     Id[2][27] = 139.76,     Id[3][27] = 70.71;
Id[1][28] = 234.97,     Id[2][28] = 141.43,     Id[3][28] = 72.38;
Id[1][29] = 236.08,     Id[2][29] = 143.10,     Id[3][29] = 74.61;
Id[1][30] = 237.19,     Id[2][30] = 144.77,     Id[3][30] = 76.28;


Id[4][0]  = 0.00,       Id[5][0]  = 0.00;
Id[4][1]  = 1.00,       Id[5][1]  = 0.15;
Id[4][2]  = 2.00,       Id[5][2]  = 0.30;
Id[4][3]  = 3.00,       Id[5][3]  = 0.45;
Id[4][4]  = 4.00,       Id[5][4]  = 0.59;
Id[4][5]  = 5.01,       Id[5][5]  = 0.74;
Id[4][6]  = 6.01,       Id[5][6]  = 0.89;
Id[4][7]  = 7.01,       Id[5][7]  = 1.04;
Id[4][8]  = 8.01,       Id[5][8]  = 1.19;
Id[4][9]  = 9.01,       Id[5][9]  = 1.34;
Id[4][10] = 10.02,      Id[5][10] = 1.49;
Id[4][11] = 10.08,      Id[5][11] = 1.64;
Id[4][12] = 11.58,      Id[5][12] = 1.78;
Id[4][13] = 12.36,      Id[5][13] = 1.93;
Id[4][14] = 13.14,      Id[5][14] = 2.08;
Id[4][15] = 13.92,      Id[5][15] = 2.23;
Id[4][16] = 15.03,      Id[5][16] = 2.56;
Id[4][17] = 16.15,      Id[5][17] = 2.90;
Id[4][18] = 17.26,      Id[5][18] = 3.23;
Id[4][19] = 18.38,      Id[5][19] = 3.57;
Id[4][20] = 19.49,      Id[5][20] = 3.90;
Id[4][21] = 20.71,      Id[5][21] = 4.57;
Id[4][22] = 21.94,      Id[5][22] = 5.24;
Id[4][23] = 23.16,      Id[5][23] = 5.90;
Id[4][24] = 24.39,      Id[5][24] = 6.57;
```

```
    Id[4][25] = 25.61,          Id[5][25] = 7.24;
    Id[4][26] = 27.06,          Id[5][26] = 8.24;
    Id[4][27] = 28.51,          Id[5][27] = 9.24;
    Id[4][28] = 29.95,          Id[5][28] = 10.25;
    Id[4][29] = 31.40,          Id[5][29] = 11.25;
    Id[4][30] = 32.85,          Id[5][30] = 12.25;


    alpha_approx  = 0.0;
    alpha_range   = 5.0;


    gamma_approx  = 0.0;
    gamma_range   = 5.0;

    lambda_approx = 0.0;
    lambda_range  = 5.0;


looping:
   alpha_initial  = alpha_approx - alpha_range;
   alpha_final    = alpha_approx + alpha_range;
   alpha_step     = alpha_range  / 5.0;


   gamma_initial  = gamma_approx - gamma_range;
   gamma_final    = gamma_approx + gamma_range;
   gamma_step     = gamma_range  / 5.0;

   lambda_initial = lambda_approx - lambda_range;
   lambda_final   = lambda_approx + lambda_range;
   lambda_step    = lambda_range  / 5.0;


   for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
      for  (  gamma  =  gamma_initial;  gamma  <=  gamma_final;  gamma  +=
gamma_step)
         for  (  lambda  =  lambda_initial;  lambda  <=  lambda_final;  lambda  +=
lambda_step)
         {
            MSE[0] = 0.0;
            for (i = 1; i <= N; i++)
               for (j = 0; j < 31; j++)
               {
                  Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) *
tanh (alpha * Vds[j] / ( Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j]);
```

```
                        MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
                    }
                    if (MSE[0] < MIN && MSE[0] > 0.0)
                    {
                        MIN = MSE[0];
                        optalpha  = alpha;
                        optgamma  = gamma;
                        optlambda = lambda;
                    }
                printf ("alpha : %0.4f\n", alpha);
                printf ("gamma : %0.4f\n", gamma);
                printf ("lambda: %0.4f\n", lambda);
                printf ("MSE   : %0.4f\n", MSE[0]);
                printf ("MIN   : %0.4f\n", MIN);
            }
    alpha_approx = optalpha;
    alpha_range  = alpha_step;

    gamma_approx = optgamma;
    gamma_range  = gamma_step;

    lambda_approx = optlambda;
    lambda_range  = lambda_step;

    if (alpha_step > 0.01) goto looping;


//plotting:


    printf ("\nOptimized value of constants\n");
    printf ("--------------------------------\n");
    printf ("alpha : %0.4f\n", optalpha);
    printf ("gamma : %0.4f\n", optgamma);
    printf ("lambda: %0.4f\n", optlambda);


    getch();


    for (i = 0; i <= N ; i++)
    {
        obsRMS[i] = 0.0;
        calRMS[i] = 0.0;
        MSE[i]    = 0.0;
    }
```

```
  for (i = 1 ; i <= N ; i++)
  {
     for (j = 0; j < 31; j++)
     {
        Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / ( Vgs[i] - VT - optgamma * Vds[j])) * ( 1 + optlambda * Vds[j]);

        MSE[0]    += pow ((Id[i][j] - Id_cal[i][j]), 2);

        MSE[i]    += pow ((Id[i][j] - Id_cal[i][j]), 2);
        obsRMS[i] += pow (Id[i][j], 2);
        calRMS[i] += pow (Id_cal[i][j], 2);
        obsRMS[0] += pow (Id[i][j], 2);
        calRMS[0] += pow (Id_cal[i][j], 2);
     }

     MSE[i]     = sqrt ( MSE[i]   / 31.0);
     obsRMS[i]  = sqrt (obsRMS[i] / 31.0);
     calRMS[i]  = sqrt (calRMS[i] / 31.0);
  }

  MSE[0]    = sqrt ( MSE[0]    / ( N * 31.0));

  obsRMS[0] = sqrt ( obsRMS[0] / ( N * 31.0 ) );
  calRMS[0] = sqrt ( calRMS[0] / ( N * 31.0 ) );

  printf ("MSE calculated   : %0.4f\n", MSE[0]);
  printf ("Average RMS error: %0.4f\n", fabs ( calRMS[0] - obsRMS[0] ) );

  Vgss[0] = -3.0;

  for( i = 1 ; i < 36; i++)
     Vgss[i] = Vgss[i - 1] + 0.1;

  for( i = 1; i <= N ; i++ )
  {
     for( j = 0; j < 36; j++)
     {
        if((Vgss[j] - Vgs[i]) < Vds_step)
           n = j;
     }

     gm[n] = (Id[i][20] - Id[i+1][20])/(Vgs[i] - Vgs[i+1]);//20 is used for Vds = 2 v.
  }

  j = 20;
  for( i = 0; i < 36 ; i++ )
```

```
        gm_cal[i] = Idss *( 2 * ( 1 - Vgss[i] / (VT + optgamma * Vds[j])) * (- 1  / (VT +
optgamma * Vds[j])) * tanh (optalpha * Vds[j] / (Vgss[i] - VT - optgamma * Vds[j]))
+ pow((1 - Vgss[i] / (VT + optgamma * Vds[j])), 2) *  (1 - pow(tanh ( optalpha *
Vds[j] / (Vgss[i] - VT - optgamma * Vds[j])), 2)) * ( optalpha * (- Vds[j]) /
pow((Vgss[i] - VT - optgamma * Vds[j]), 2))) * ( 1 + optlambda * Vds[j] );

    i = 1;
    for ( j = 0; j < 31; j++)
    {
        gd[j+1]  = (Id[i][j+2]-Id[i][j])/(Vds_step*2);
        gd_cal[j] = Idss * (2 *( 1 - Vgs[i] / (VT + optgamma * Vds[j])) * (optgamma *
Vgs[i] / pow((VT + optgamma * Vds[j]), 2)) *  tanh (optalpha * Vds[j] / (Vgs[i] - VT
- optgamma  * Vds[j]))  * ( 1 + optlambda * Vds[j]) + pow((1 - Vgs[i] / (VT +
optgamma * Vds[j])), 2) * (1 - pow(tanh ( optalpha * Vds[j] / (Vgs[i] - VT -
optgamma * Vds[j])), 2)) * ( optalpha * (Vgs[i] - VT - optgamma * Vds[j]) +
(optalpha * optgamma * Vds[j])) / pow((Vgs[i] - VT - gamma * Vds[j]), 2) * ( 1 +
optlambda * Vds[j]) + pow((1 - Vgs[i] / (VT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / ( Vgs[i] - VT - optgamma * Vds[j])) * optlambda );
    }

    FILE *fp;
    fp = fopen("optimal.xls", "w");

    fprintf (fp,"\t\t\tIdss =%6.3f mA / mm\n\t\t\tVT = %6.3fV\n\n",Idss, VT);
    fprintf (fp,"\t\tMSE Calculated = %0.4f\t\t\tAv. RMS Error = %0.4f\n\n", MSE[0],
fabs ( calRMS[0] - obsRMS[0] ) );

    fprintf (fp,"\t\tMSE[1]\tMSE[2]\tMSE[3]\tMSE[4]\tMSE[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",MSE[1], MSE[2], MSE[3],
MSE[4], MSE[5]);

    fprintf (fp,"\t\tobsRMS[1]\tobsRMS[2]\tobsRMS[3]\tobsRMS[4]\tobsRMS[5]\n");
    fprintf  (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",obsRMS[1},  obsRMS[2],
obsRMS[3], obsRMS[4], obsRMS[5]);

    fprintf (fp,"\t\tcalRMS[1]\tcalRMS[2]\tcalRMS[3]\tcalRMS[4]\tcalRMS[5]\n");
    fprintf  (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",calRMS[1],  calRMS[2],
calRMS[3], calRMS[4], calRMS[5]);

    fprintf (fp,"\t\tError1\tError2\tError3\tError4\tError5\n");
    fprintf   (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",   fabs(calRMS[1]   -
obsRMS[1]), fabs(calRMS[2] - obsRMS[2]), fabs(calRMS[3] - obsRMS[3]),
fabs(calRMS[4] - obsRMS[4]), fabs(calRMS[5] - obsRMS[5]));

    fprintf (fp,"\tAlpha = %0.4f\t\tGamma = %0.4f\t\tLambda = %0.4f\n\n",optalpha,
optgamma, optlambda);
```

```
    fprintf
(fp,"Vds\tIds1(meas.)\tIds1(cal.)\tIds2(meas.)\tds2(cal.)\tIds3(meas.)\tIds3(cal.)\tIds4(
meas.)\tIds4(cal.)\tIds5(meas.)\tIds5(cal.)\n");
    for ( j = 0; j < 31; j++)
        fprintf(fp,"
%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t\n",
Vds[j],Id[1][j],Id_cal[1][j],Id[2][j],Id_cal[2][j],Id[3][j],Id_cal[3][j],Id[4][j],Id_cal[4][j
],Id[5][j],Id_cal[5][j]);
    fprintf (fp,"\nVgss\tgm(meas.)\tgm(cal.)\n");
    for ( j = 0; j < 36; j++)
        fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vgss[j], gm[j], gm_cal[j]);
    fprintf (fp,"\nVds\tgd(meas.)\tgd(cal.)\n");
    for ( j = 0; j < 31; j++)
        fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vds[j], gd[j], gd_cal[j]);

    fclose(fp);

    getch();
}
```

Program of SSIM Model for Device A-74-1

```
#include <stdio.h>
#include <conio.h>
#include <math.h>



#define Idss 192.0
#define VT -1.999999
#define N 5 // Number of curve to be simulated.



void main(void)
{
    int i, j, n;
    double Vgs[N+1],Id[N+1][31], gm[36], gd[36], Vds[31], Vgss[36];
    double Id_cal[N+1][31], gm_cal[36], gd_cal[36];
    double Vds_step;
    double MSE[N+1], MIN = 1e10;
    double obsRMS[N+1], calRMS[N+1];
    double optalpha, optbeta, optgamma, optlambda;
    double alpha, beta, gamma, lambda;
    double alpha_approx, alpha_range, alpha_initial, alpha_final, alpha_step;
    double beta_approx, beta_range, beta_initial, beta_final, beta_step;
```

```
   double  gamma_approx,    gamma_range,    gamma_initial,    gamma_final,
gamma_step;
   double lambda_approx, lambda_range, lambda_initial, lambda_final, lambda_step;


   Vds_step = 0.1;

   Vgs[1] =  0.0;
   Vgs[2] = -0.5;
   Vgs[3] = -1.0;
   Vgs[4] = -1.5;
   Vgs[5] = -2.0;

   for ( j = 0; j < 31 ;j++)
      Vds[j] = j* Vds_step;



   Id[1][0]  = 0.00,         Id[2][0]  = 0.00,          Id[3][0]  = 0.00;
   Id[1][1]  = 34.52,     Id[2][1]  = 21.71,      Id[3][1]  = 9.47;
   Id[1][2]  = 64.03,     Id[2][2]  = 38.98,      Id[3][2]  = 15.59;
   Id[1][3]  = 91.31,     Id[2][3]  = 56.24,      Id[3][3]  = 22.27;
   Id[1][4]  = 118.04,       Id[2][4]  = 71.27,        Id[3][4]  = 26.73;
   Id[1][5]  = 139.76,       Id[2][5]  = 80.73,        Id[3][5]  = 30.07;
   Id[1][6]  = 158.69,       Id[2][6]  = 88.53,        Id[3][6]  = 33.41;
   Id[1][7]  = 173.72,       Id[2][7]  = 94.65,        Id[3][7]  = 35.63;
   Id[1][8]  = 183.74,       Id[2][8]  = 99.11,        Id[3][8]  = 37.86;
   Id[1][9]  = 190.98,       Id[2][9]  = 102.45,          Id[3][9]  = 40.09;
   Id[1][10] = 197.10,       Id[2][10] = 105.79,          Id[3][10] = 41.76;
   Id[1][11] = 201.00,       Id[2][11] = 108.57,          Id[3][11] = 43.99;
   Id[1][12] = 204.90,       Id[2][12] = 111.36,          Id[3][12] = 45.66;
   Id[1][13] = 208.24,       Id[2][13] = 113.59,          Id[3][13] = 47.33;
   Id[1][14] = 211.02,       Id[2][14] = 115.81,          Id[3][14] = 49.55;
   Id[1][15] = 213.25,       Id[2][15] = 118.04,          Id[3][15] = 50.67;
   Id[1][16] = 215.48,       Id[2][16] = 120.27,          Id[3][16] = 52.90;
   Id[1][17] = 217.15,       Id[2][17] = 122.49,          Id[3][17] = 54.01;
   Id[1][18] = 219.38,       Id[2][18] = 124.16,          Id[3][18] = 55.68;
   Id[1][19] = 221.60,       Id[2][19] = 126.39,          Id[3][19] = 57.35;
   Id[1][20] = 223.27,       Id[2][20] = 128.06,          Id[3][20] = 59.02;
   Id[1][21] = 224.94,       Id[2][21] = 129.73,          Id[3][21] = 61.25;
   Id[1][22] = 226.61,       Id[2][22] = 131.40,          Id[3][22] = 62.92;
   Id[1][23] = 227.73,       Id[2][23] = 133.07,          Id[3][23] = 64.03;
   Id[1][24] = 229.40,       Id[2][24] = 135.30,          Id[3][24] = 65.70;
   Id[1][25] = 231.07,       Id[2][25] = 136.97,          Id[3][25] = 67.37;
   Id[1][26] = 232.18,       Id[2][26] = 138.64,          Id[3][26] = 69.04;
   Id[1][27] = 233.85,       Id[2][27] = 139.76,          Id[3][27] = 70.71;
   Id[1][28] = 234.97,       Id[2][28] = 141.43,          Id[3][28] = 72.38;
```

```
Id[1][29] = 236.08,        Id[2][29] = 143.10,        Id[3][29] = 74.61;
Id[1][30] = 237.19,        Id[2][30] = 144.77,        Id[3][30] = 76.28;


Id[4][0]  = 0.00,          Id[5][0]  = 0.00;
Id[4][1]  = 1.00,          Id[5][1]  = 0.15;
Id[4][2]  = 2.00,          Id[5][2]  = 0.30;
Id[4][3]  = 3.00,          Id[5][3]  = 0.45;
Id[4][4]  = 4.00,          Id[5][4]  = 0.59;
Id[4][5]  = 5.01,          Id[5][5]  = 0.74;
Id[4][6]  = 6.01,          Id[5][6]  = 0.89;
Id[4][7]  = 7.01,          Id[5][7]  = 1.04;
Id[4][8]  = 8.01,          Id[5][8]  = 1.19;
Id[4][9]  = 9.01,          Id[5][9]  = 1.34;
Id[4][10] = 10.02,         Id[5][10] = 1.49;
Id[4][11] = 10.08,         Id[5][11] = 1.64;
Id[4][12] = 11.58,         Id[5][12] = 1.78;
Id[4][13] = 12.36,         Id[5][13] = 1.93;
Id[4][14] = 13.14,         Id[5][14] = 2.08;
Id[4][15] = 13.92,         Id[5][15] = 2.23;
Id[4][16] = 15.03,         Id[5][16] = 2.56;
Id[4][17] = 16.15,         Id[5][17] = 2.90;
Id[4][18] = 17.26,         Id[5][18] = 3.23;
Id[4][19] = 18.38,         Id[5][19] = 3.57;
Id[4][20] = 19.49,         Id[5][20] = 3.90;
Id[4][21] = 20.71,         Id[5][21] = 4.57;
Id[4][22] = 21.94,         Id[5][22] = 5.24;
Id[4][23] = 23.16,         Id[5][23] = 5.90;
Id[4][24] = 24.39,         Id[5][24] = 6.57;
Id[4][25] = 25.61,         Id[5][25] = 7.24;
Id[4][26] = 27.06,         Id[5][26] = 8.24;
Id[4][27] = 28.51,         Id[5][27] = 9.24;
Id[4][28] = 29.95,         Id[5][28] = 10.25;
Id[4][29] = 31.40,         Id[5][29] = 11.25;
Id[4][30] = 32.85,         Id[5][30] = 12.25;
alpha_approx  = 0.0;
alpha_range   = 5.0;

beta_approx   = 0.0;
beta_range    = 1.0;

gamma_approx  = -0.5;
gamma_range   =  0.5;

lambda_approx = 0.5;
lambda_range  = 0.5;
```

```
looping:
   alpha_initial  = alpha_approx - alpha_range;
   alpha_final    = alpha_approx + alpha_range;
   alpha_step     = alpha_range  / 5.0;

   beta_initial  = beta_approx -  beta_range;
   beta_final    = beta_approx +  beta_range;
   beta_step     = beta_range  /  5.0;

   gamma_initial  = gamma_approx - gamma_range;
   gamma_final    = gamma_approx + gamma_range;
   gamma_step     = gamma_range  / 5.0;

   lambda_initial = lambda_approx - lambda_range;
   lambda_final   = lambda_approx + lambda_range;
   lambda_step    = lambda_range  / 5.0;


   for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
      for ( beta = beta_initial; beta <= beta_final; beta += beta_step)
         for  (  gamma  =  gamma_initial;  gamma  <=  gamma_final;  gamma  +=
gamma_step)
            for  (  lambda  =  lambda_initial;  lambda  <=  lambda_final;  lambda  +=
lambda_step)
            {
               MSE[0] = 0.0;
               for (i = 1; i <= N; i++)
                  for (j = 0; j < 31; j++)
                  {

                     Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2)
* tanh (alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] +
beta * Vgs[i]);

                     MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
                  }
                  if (MSE[0] < MIN && MSE[0] > 0.0)
                  {
                     MIN = MSE[0];
                     optalpha  = alpha;
                     optbeta   = beta;
                     optgamma  = gamma;
                     optlambda = lambda;
                  }
               printf ("alpha : %0.4f\n", alpha);
               printf ("beta  : %0.4f\n", beta);
               printf ("gamma : %0.4f\n", gamma);
```

```
                printf ("lambda: %0.4f\n", lambda);
                printf ("MSE   : %0.4f\n", MSE[0]);
                printf ("MIN   : %0.4f\n", MIN);
            }
    alpha_approx = optalpha;
    alpha_range  = alpha_step;

    beta_approx  = optbeta;
    beta_range   = beta_step;

    gamma_approx = optgamma;
    gamma_range  = gamma_step;

    lambda_approx = optlambda;
    lambda_range  = lambda_step;

    if (alpha_step > 0.01) goto looping;


//plotting:


    printf ("\nOptimized value of constants\n");
    printf ("--------------------------------\n");
    printf ("alpha : %0.4f\n", optalpha);
    printf ("beta  : %0.4f\n", optbeta);
    printf ("gamma : %0.4f\n", optgamma);
    printf ("lambda: %0.4f\n", optlambda);


    getch();


    for (i = 0; i <= N ; i++)
    {
       obsRMS[i] = 0.0;
       calRMS[i] = 0.0;
       MSE[i]    = 0.0;
    }

    for (i = 1 ; i <= N ; i++)
    {
       for (j = 0; j < 31; j++)
       {
          Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / (Vgs[i] - VT - optgamma * Vds[j])) * ( 1 + optlambda * Vds[j] +
optbeta * Vgs[i]);
```

```
            MSE[0]    += pow ((Id[i][j] - Id_cal[i][j]), 2);

            MSE[i]    += pow ((Id[i][j] - Id_cal[i][j]), 2);
            obsRMS[i] += pow (Id[i][j], 2);
            calRMS[i] += pow (Id_cal[i][j], 2);
            obsRMS[0] += pow (Id[i][j], 2);
            calRMS[0] += pow (Id_cal[i][j], 2);
        }

        MSE[i]    = sqrt ( MSE[i]   / 31.0);
        obsRMS[i]  = sqrt (obsRMS[i] / 31.0);
        calRMS[i]  = sqrt (calRMS[i] / 31.0);
    }

    MSE[0]    = sqrt ( MSE[0]    / ( N * 31.0));

    obsRMS[0] = sqrt ( obsRMS[0] / ( N * 31.0 ) );
    calRMS[0] = sqrt ( calRMS[0] / ( N * 31.0 ) );

    printf ("MSE calculated   : %0.4f\n", MSE[0]);
    printf ("Average RMS error: %0.4f\n", fabs ( calRMS[0] - obsRMS[0] ) );

    Vgss[0] = -3.0;

    for( i = 1 ; i < 36; i++)
        Vgss[i] = Vgss[i - 1] + 0.1;

    for( i = 1; i <= N ; i++ )
    {
        for( j = 0; j < 36; j++)
        {
            if((Vgss[j] - Vgs[i]) < Vds_step)
            n = j;
        }

        gm[n] = (Id[i][20] - Id[i+1][20])/(Vgs[i] - Vgs[i+1]);//20 is used for Vds = 2 v.
    }

    j = 20;
    for( i = 0; i < 36 ; i++ )
        gm_cal[i] = Idss * ( 2 *( 1 - Vgss[i] / (VT + optgamma * Vds[j])) * (- 1  / (VT +
optgamma * Vds[j])) * tanh (optalpha * Vds[j] / (Vgss[i] - VT - optgamma * Vds[j]))
* ( 1 + optlambda * Vds[j] + optbeta * Vgss[i]) + pow((1 - Vgss[i] / (VT + optgamma
* Vds[j])), 2) *  (1 - pow(tanh ( optalpha * Vds[j] / (Vgss[i] - VT - optgamma *
Vds[j])), 2)) * ( optalpha * (- Vds[j]) / pow((Vgss[i] - VT - optgamma * Vds[j]), 2)) *
( 1 + optlambda * Vds[j] + optbeta * Vgss[i]) + pow((1 - Vgss[i] / (VT + optgamma *
Vds[j])),2) * tanh (optalpha * Vds[j] / (Vgss[i] - VT - optgamma * Vds[j])) *
optbeta);

   i = 1;
   for ( j = 0; j < 31; j++)
   {
      gd[j+1]  = (Id[i][j+2]-Id[i][j])/(Vds_step*2);
      gd_cal[j] = Idss * (2 *( 1 - Vgs[i] / (VT + optgamma * Vds[j])) * (optgamma *
Vgs[i] / pow((VT + optgamma * Vds[j]), 2)) * tanh (optalpha * Vds[j] / (Vgs[i] - VT
- optgamma  * Vds[j])) * ( 1 + optlambda * Vds[j] + optbeta * Vgs[i]) + pow((1 -
Vgs[i] / (VT + optgamma * Vds[j])), 2) * (1 - pow( tanh ( optalpha * Vds[j] / (Vgs[i] -
VT - optgamma * Vds[j])), 2)) * (( optalpha * (Vgs[i] - VT - optgamma * Vds[j]) +
(optalpha * optgamma * Vds[j])) / pow ((Vgs[i] - VT - gamma * Vds[j]), 2)) * ( 1 +
optlambda * Vds[j] + optbeta * Vgs[i]) + pow((1 - Vgs[i] / (VT + optgamma *
Vds[j])),2) * tanh (optalpha * Vds[j] / ( Vgs[i] - VT - optgamma * Vds[j])) *
optlambda );
   }

   FILE *fp;
   fp = fopen("optimal.xls", "w");

   fprintf (fp,"\t\t\tIdss =%6.3f mA / mm\n\t\t\tVT = %6.3fV\n\n",Idss, VT);
   fprintf (fp,"\t\tMSE Calculated = %0.4f\t\t\tAv. RMS Error = %0.4f\n\n", MSE[0],
fabs ( calRMS[0] - obsRMS[0] ) );

   fprintf (fp,"\t\tMSE[1]\tMSE[2]\tMSE[3]\tMSE[4]\tMSE[5]\n");
   fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",MSE[1], MSE[2], MSE[3],
MSE[4], MSE[5]);

   fprintf (fp,"\t\tobsRMS[1]\tobsRMS[2]\tobsRMS[3]\tobsRMS[4]\tobsRMS[5]\n");
   fprintf  (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",obsRMS[1],  obsRMS[2],
obsRMS[3], obsRMS[4], obsRMS[5]);

   fprintf (fp,"\t\tcalRMS[1]\tcalRMS[2]\tcalRMS[3]\tcalRMS[4]\tcalRMS[5]\n");
   fprintf  (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",calRMS[1],  calRMS[2],
calRMS[3], calRMS[4], calRMS[5]);

   fprintf (fp,"\t\tError1\tError2\tError3\tError4\tError5\n");
   fprintf   (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",   fabs(calRMS[1]   -
obsRMS[1]),  fabs(calRMS[2]  -  obsRMS[2]),  fabs(calRMS[3]  -  obsRMS[3]),
fabs(calRMS[4] - obsRMS[4]), fabs(calRMS[5] - obsRMS[5]));

   fprintf  (fp,"\tAlpha  =  %0.4f\t\tBeta  =  %0.4f\t\tGamma  =  %0.4f\t\tLambda  =
%0.4f\n\n",optalpha, optbeta, optgamma, optlambda);
```

```
   fprintf
(fp,"Vds\tIds1(meas.)\tIds1(cal.)\tIds2(meas.)\tds2(cal.)\tIds3(meas.)\tIds3(cal.)\tIds4(
meas.)\tIds4(cal.)\tIds5(meas.)\tIds5(cal.)\n");
   for ( j = 0; j < 31; j++)
      fprintf(fp,"
%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t\n",
Vds[j],Id[1][j],Id_cal[1][j],Id[2][j],Id_cal[2][j],Id[3][j],Id_cal[3][j],Id[4][j],Id_cal[4][j
],Id[5][j],Id_cal[5][j]);
   fprintf (fp,"\nVgss\tgm(meas.)\tgm(cal.)\n");
   for ( j = 0; j < 36; j++)
      fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vgss[j], gm[j], gm_cal[j]);
   fprintf (fp,"\nVds\tgd(meas.)\tgd(cal.)\n");
   for ( j = 0; j < 31; j++)
      fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vds[j], gd[j], gd_cal[j]);

   fclose(fp);

   getch();
}
```

Program of TSIM Model for Device A-74-1

```
#include <stdio.h>
#include <conio.h>
#include <math.h>


#define Idss 192.0
#define VT_plus_delVT -1.69
#define N 5 // Number of curve to be simulated.

void main(void)
{
   int i, j, n;
   double Vgs[N+1],Id[N+1][31], gm[36], gd[36], Vds[31], Vgss[36];
   double Id_cal[N+1][31], gm_cal[36], gd_cal[36];
   double Vds_step;
   double MSE[N+1], MIN = 1e10;
   double obsRMS[N+1], calRMS[N+1];
   double optalpha, optbeta, optgamma, optlambda;
   double alpha, beta, gamma, lambda;
   double alpha_approx, alpha_range, alpha_initial, alpha_final, alpha_step;
   double beta_approx, beta_range, beta_initial, beta_final, beta_step;
   double gamma_approx, gamma_range, gamma_initial, gamma_final,
gamma_step;
   double lambda_approx, lambda_range, lambda_initial, lambda_final, lambda_step;
```

```
Vds_step = 0.1;

Vgs[1] =  0.0;
Vgs[2] = -0.5;
Vgs[3] = -1.0;
Vgs[4] = -1.5;
Vgs[5] = -2.0;

for ( j = 0; j < 31 ;j++)
   Vds[j] = j* Vds_step;


Id[1][0]   = 0.00,           Id[2][0]   = 0.00,          Id[3][0]   = 0.00;
Id[1][1]   = 34.52,       Id[2][1]   = 21.71,       Id[3][1]   = 9.47;
Id[1][2]   = 64.03,       Id[2][2]   = 38.98,       Id[3][2]   = 15.59;
Id[1][3]   = 91.31,       Id[2][3]   = 56.24,       Id[3][3]   = 22.27;
Id[1][4]   = 118.04,         Id[2][4]   = 71.27,        Id[3][4]   = 26.73;
Id[1][5]   = 139.76,         Id[2][5]   = 80.73,        Id[3][5]   = 30.07;
Id[1][6]   = 158.69,         Id[2][6]   = 88.53,        Id[3][6]   = 33.41;
Id[1][7]   = 173.72,         Id[2][7]   = 94.65,        Id[3][7]   = 35.63;
Id[1][8]   = 183.74,         Id[2][8]   = 99.11,        Id[3][8]   = 37.86;
Id[1][9]   = 190.98,         Id[2][9]   = 102.45,          Id[3][9]   = 40.09;
Id[1][10] = 197.10,          Id[2][10] = 105.79,           Id[3][10] = 41.76;
Id[1][11] = 201.00,          Id[2][11] = 108.57,           Id[3][11] = 43.99;
Id[1][12] = 204.90,          Id[2][12] = 111.36,           Id[3][12] = 45.66;
Id[1][13] = 208.24,          Id[2][13] = 113.59,           Id[3][13] = 47.33;
Id[1][14] = 211.02,          Id[2][14] = 115.81,           Id[3][14] = 49.55;
Id[1][15] = 213.25,          Id[2][15] = 118.04,           Id[3][15] = 50.67;
Id[1][16] = 215.48,          Id[2][16] = 120.27,           Id[3][16] = 52.90;
Id[1][17] = 217.15,          Id[2][17] = 122.49,           Id[3][17] = 54.01;
Id[1][18] = 219.38,          Id[2][18] = 124.16,           Id[3][18] = 55.68;
Id[1][19] = 221.60,          Id[2][19] = 126.39,           Id[3][19] = 57.35;
Id[1][20] = 223.27,          Id[2][20] = 128.06,           Id[3][20] = 59.02;
Id[1][21] = 224.94,          Id[2][21] = 129.73,           Id[3][21] = 61.25;
Id[1][22] = 226.61,          Id[2][22] = 131.40,           Id[3][22] = 62.92;
Id[1][23] = 227.73,          Id[2][23] = 133.07,           Id[3][23] = 64.03;
Id[1][24] = 229.40,          Id[2][24] = 135.30,           Id[3][24] = 65.70;
Id[1][25] = 231.07,          Id[2][25] = 136.97,           Id[3][25] = 67.37;
Id[1][26] = 232.18,          Id[2][26] = 138.64,           Id[3][26] = 69.04;
Id[1][27] = 233.85,          Id[2][27] = 139.76,           Id[3][27] = 70.71;
Id[1][28] = 234.97,          Id[2][28] = 141.43,           Id[3][28] = 72.38;
Id[1][29] = 236.08,          Id[2][29] = 143.10,           Id[3][29] = 74.61;
Id[1][30] = 237.19,          Id[2][30] = 144.77,           Id[3][30] = 76.28;


Id[4][0]   = 0.00,           Id[5][0]   = 0.00;
Id[4][1]   = 1.00,           Id[5][1]   = 0.15;
```

```
    Id[4][2]  = 2.00,          Id[5][2]  = 0.30;
    Id[4][3]  = 3.00,          Id[5][3]  = 0.45;
    Id[4][4]  = 4.00,          Id[5][4]  = 0.59;
    Id[4][5]  = 5.01,          Id[5][5]  = 0.74;
    Id[4][6]  = 6.01,          Id[5][6]  = 0.89;
    Id[4][7]  = 7.01,          Id[5][7]  = 1.04;
    Id[4][8]  = 8.01,          Id[5][8]  = 1.19;
    Id[4][9]  = 9.01,          Id[5][9]  = 1.34;
    Id[4][10] = 10.02,         Id[5][10] = 1.49;
    Id[4][11] = 10.08,         Id[5][11] = 1.64;
    Id[4][12] = 11.58,         Id[5][12] = 1.78;
    Id[4][13] = 12.36,         Id[5][13] = 1.93;
    Id[4][14] = 13.14,         Id[5][14] = 2.08;
    Id[4][15] = 13.92,         Id[5][15] = 2.23;
    Id[4][16] = 15.03,         Id[5][16] = 2.56;
    Id[4][17] = 16.15,         Id[5][17] = 2.90;
    Id[4][18] = 17.26,         Id[5][18] = 3.23;
    Id[4][19] = 18.38,         Id[5][19] = 3.57;
    Id[4][20] = 19.49,         Id[5][20] = 3.90;
    Id[4][21] = 20.71,         Id[5][21] = 4.57;
    Id[4][22] = 21.94,         Id[5][22] = 5.24;
    Id[4][23] = 23.16,         Id[5][23] = 5.90;
    Id[4][24] = 24.39,         Id[5][24] = 6.57;
    Id[4][25] = 25.61,         Id[5][25] = 7.24;
    Id[4][26] = 27.06,         Id[5][26] = 8.24;
    Id[4][27] = 28.51,         Id[5][27] = 9.24;
    Id[4][28] = 29.95,         Id[5][28] = 10.25;
    Id[4][29] = 31.40,         Id[5][29] = 11.25;
    Id[4][30] = 32.85,         Id[5][30] = 12.25;

    alpha_approx  = 0.0;
    alpha_range   = 5.0;

    beta_approx   = 0.0;
    beta_range    = 1.0;

    gamma_approx  =-0.5;
    gamma_range   = 0.5;

    lambda_approx = 0.5;
    lambda_range  = 0.5;


looping:
    alpha_initial  = alpha_approx - alpha_range;
    alpha_final    = alpha_approx + alpha_range;
    alpha_step     = alpha_range / 5.0;
    beta_initial   = beta_approx -  beta_range;
    beta_final     = beta_approx +  beta_range;
    beta_step      = beta_range  /  5.0;

    gamma_initial  = gamma_approx - gamma_range;
    gamma_final    = gamma_approx + gamma_range;
    gamma_step     = gamma_range  / 5.0;

    lambda_initial = lambda_approx - lambda_range;
    lambda_final   = lambda_approx + lambda_range;
    lambda_step    = lambda_range  / 5.0;


    for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
        for ( beta = beta_initial; beta <= beta_final; beta += beta_step)
            for ( gamma = gamma_initial; gamma <= gamma_final; gamma +=
gamma_step)
                for ( lambda = lambda_initial; lambda <= lambda_final; lambda +=
lambda_step)
                {
                    MSE[0] = 0.0;
                    for (i = 1; i <= N; i++)
                        for (j = 0; j < 31; j++)
                        {

                            Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT_plus_delVT + gamma
* Vds[j])),2) * tanh (alpha * Vds[j] / (Vgs[i] - VT_plus_delVT - gamma * Vds[j])) * (
1 + lambda * Vds[j] + beta * Vgs[i]);

                            MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
                        }
                        if (MSE[0] < MIN && MSE[0] > 0.0)
                        {
                            MIN = MSE[0];
                            optalpha  = alpha;
                            optbeta   = beta;
                            optgamma  = gamma;
                            optlambda = lambda;
                        }
                    printf ("alpha : %0.4f\n", alpha);
                    printf ("beta  : %0.4f\n", beta);
                    printf ("gamma : %0.4f\n", gamma);
                    printf ("lambda: %0.4f\n", lambda);
                    printf ("MSE   : %0.4f\n", MSE[0]);
                    printf ("MIN   : %0.4f\n", MIN);
                }
    alpha_approx = optalpha;
    alpha_range  = alpha_step;

    beta_approx  = optbeta;
    beta_range   = beta_step;

    gamma_approx = optgamma;
    gamma_range  = gamma_step;

    lambda_approx = optlambda;
    lambda_range  = lambda_step;

    if (alpha_step > 0.001) goto looping;


//plotting:


    printf ("\nOptimized value of constants\n");
    printf ("------------------------------\n");
    printf ("alpha : %0.4f\n", optalpha);
    printf ("beta  : %0.4f\n", optbeta);
    printf ("gamma : %0.4f\n", optgamma);
    printf ("lambda: %0.4f\n", optlambda);


    getch();


    for (i = 0; i <= N ; i++)
    {
      obsRMS[i] = 0.0;
      calRMS[i] = 0.0;
      MSE[i]    = 0.0;
    }

    for (i = 1 ; i <= N ; i++)
    {
      for (j = 0; j < 31; j++)
      {
        Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT_plus_delVT + optgamma *
Vds[j])),2) * tanh (optalpha * Vds[j] / (Vgs[i] - VT_plus_delVT - optgamma *
Vds[j])) * ( 1 + optlambda * Vds[j] + optbeta * Vgs[i]);

        MSE[0]   += pow ((Id[i][j] - Id_cal[i][j]), 2);

        MSE[i]   += pow ((Id[i][j] - Id_cal[i][j]), 2);
```

```
            obsRMS[i] += pow (Id[i][j], 2);
            calRMS[i] += pow (Id_cal[i][j], 2);
            obsRMS[0] += pow (Id[i][j], 2);
            calRMS[0] += pow (Id_cal[i][j], 2);
        }

        MSE[i]    = sqrt ( MSE[i]   / 31.0);
        obsRMS[i]  = sqrt (obsRMS[i] / 31.0);
        calRMS[i]  = sqrt (calRMS[i] / 31.0);
    }

    MSE[0]    = sqrt ( MSE[0]   / ( N * 31.0));

    obsRMS[0] = sqrt ( obsRMS[0] / ( N * 31.0 ) );
    calRMS[0] = sqrt ( calRMS[0] / ( N * 31.0 ) );

    printf ("MSE calculated   : %0.4f\n", MSE[0]);
    printf ("Average RMS error: %0.4f\n", fabs ( calRMS[0] - obsRMS[0] ) );

    Vgss[0] = -3.0;

    for( i = 1 ; i < 36; i++)
        Vgss[i] = Vgss[i - 1] + 0.1;

    for( i = 1; i <= N ; i++ )
    {
        for( j = 0; j < 36; j++)
        {
            if((Vgss[j] - Vgs[i]) < Vds_step)
            n = j;
        }

        gm[n] = (Id[i][20] - Id[i+1][20])/(Vgs[i] - Vgs[i+1]);//20 is used for Vds = 2 v.
    }

    j = 20;
    for( i = 0; i < 36 ; i++ )
        gm_cal[i] = Idss * ( 2 *( 1 - Vgss[i] / (VT_plus_delVT + optgamma * Vds[j])) *
(- 1  / (VT_plus_delVT + optgamma * Vds[j])) * tanh (optalpha * Vds[j] / (Vgss[i] -
VT_plus_delVT - optgamma * Vds[j])) * ( 1 + optlambda * Vds[j] + optbeta *
Vgss[i]) + pow((1 - Vgss[i] / (VT_plus_delVT + optgamma * Vds[j])), 2) *  (1 -
pow(tanh ( optalpha * Vds[j] / (Vgss[i] - VT_plus_delVT - optgamma * Vds[j])), 2))
* ( optalpha * (- Vds[j]) / pow((Vgss[i] - VT_plus_delVT - optgamma * Vds[j]), 2)) *
( 1 + optlambda * Vds[j] + optbeta * Vgss[i]) + pow((1 - Vgss[i] / (VT_plus_delVT +
optgamma * Vds[j])),2) * tanh (optalpha * Vds[j] / (Vgss[i] - VT_plus_delVT -
optgamma * Vds[j])) * optbeta);
```

```
    i = 1;
    for ( j = 0; j < 31; j++)
    {
      gd[j+1]  = (Id[i][j+2]-Id[i][j])/(Vds_step*2);
      gd_cal[j] = Idss * (2 *( 1 - Vgs[i] / (VT_plus_delVT + optgamma * Vds[j])) *
(optgamma * Vgs[i] / pow((VT_plus_delVT + optgamma * Vds[j]), 2)) * tanh
(optalpha * Vds[j] / (Vgs[i] - VT_plus_delVT - optgamma  * Vds[j])) * ( 1 +
optlambda * Vds[j] + optbeta * Vgs[i]) + pow((1 - Vgs[i] / (VT_plus_delVT +
optgamma * Vds[j])), 2) * (1 - pow( tanh ( optalpha * Vds[j] / (Vgs[i] -
VT_plus_delVT - optgamma * Vds[j])), 2)) * (( optalpha * (Vgs[i] - VT_plus_delVT
- optgamma * Vds[j]) + (optalpha * optgamma * Vds[j])) / pow ((Vgs[i] -
VT_plus_delVT - gamma * Vds[j]), 2)) * ( 1 + optlambda * Vds[j] + optbeta *
Vgs[i]) + pow((1 - Vgs[i] / (VT_plus_delVT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / ( Vgs[i] - VT_plus_delVT - optgamma * Vds[j])) * optlambda );
    }

    FILE *fp;
    fp = fopen("optimal.xls", "w");

    fprintf (fp,"\t\t\tIdss =%6.3f mA / mm\n\t\t\tVT_plus_delVT = %6.3fV\n\n",Idss,
VT_plus_delVT);
    fprintf (fp,"\t\tMSE Calculated = %0.4f\t\t\tAv. RMS Error = %0.4f\n\n", MSE[0],
fabs ( calRMS[0] - obsRMS[0] ) );

    fprintf (fp,"\t\tMSE[1]\tMSE[2]\tMSE[3]\tMSE[4]\tMSE[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",MSE[1], MSE[2], MSE[3],
MSE[4], MSE[5]);

    fprintf (fp,"\t\tobsRMS[1]\tobsRMS[2]\tobsRMS[3]\tobsRMS[4]\tobsRMS[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",obsRMS[1], obsRMS[2],
obsRMS[3], obsRMS[4], obsRMS[5]);

    fprintf (fp,"\t\tcalRMS[1]\tcalRMS[2]\tcalRMS[3]\tcalRMS[4]\tcalRMS[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",calRMS[1], calRMS[2],
calRMS[3], calRMS[4], calRMS[5]);

    fprintf (fp,"\t\tError1\tError2\tError3\tError4\tError5\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n", fabs(calRMS[1] -
obsRMS[1]), fabs(calRMS[2] - obsRMS[2]), fabs(calRMS[3] - obsRMS[3]),
fabs(calRMS[4] - obsRMS[4]), fabs(calRMS[5] - obsRMS[5]));

    fprintf (fp,"\tAlpha = %0.4f\t\tBeta = %0.4f\t\tGamma = %0.4f\t\tLambda =
%0.4f\n\n",optalpha, optbeta, optgamma, optlambda);

    fprintf
(fp,"Vds\tIds1(meas.)\tIds1(cal.)\tIds2(meas.)\tds2(cal.)\tIds3(meas.)\tIds3(cal.)\tIds4(
meas.)\tIds4(cal.)\tIds5(meas.)\tIds5(cal.)\n");
```

```
    for ( j = 0; j < 31; j++)
       fprintf(fp,"
%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t\n",
Vds[j],Id[1][j],Id_cal[1][j],Id[2][j],Id_cal[2][j],Id[3][j],Id_cal[3][j],Id[4][j],Id_cal[4][j
],Id[5][j],Id_cal[5][j]);
    fprintf (fp,"\nVgss\tgm(meas.)\tgm(cal.)\n");
    for ( j = 0; j < 36; j++)
       fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vgss[j], gm[j], gm_cal[j]);
    fprintf (fp,"\nVds\tgd(meas.)\tgd(cal.)\n");
    for ( j = 0; j < 31; j++)
       fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vds[j], gd[j], gd_cal[j]);

    fclose(fp);

    getch();
}
```

Program of New Reduced-Time Algorithm for Device A-74-1

```
#include <stdio.h>
#include <conio.h>
#include <math.h>


#define Idss 192.0
#define VT -1.999999
#define N 5 // Number of curve to be simulated.
void main(void)
{
   int i, j, n;
   double Vgs[N+1],Id[N+1][31], gm[36], gd[36], Vds[31], Vgss[36];
   double Id_cal[N+1][31], gm_cal[36], gd_cal[36];
   double Vds_step;
   double MSE[N+1], MIN = 1e10;
   double obsRMS[N+1], calRMS[N+1];
   double optalpha, optbeta, optgamma, optlambda;
   double alpha, beta, gamma, lambda;
   double alpha_approx, alpha_range, alpha_initial, alpha_final, alpha_step;
   double beta_approx, beta_range, beta_initial, beta_final, beta_step;
   double gamma_approx, gamma_range, gamma_initial, gamma_final,
gamma_step;
   double lambda_approx, lambda_range, lambda_initial, lambda_final, lambda_step;

   Vds_step = 0.1;
```

```
Vgs[1] =  0.0;
Vgs[2] = -0.5;
Vgs[3] = -1.0;
Vgs[4] = -1.5;
Vgs[5] = -2.0;

for ( j = 0; j < 31 ;j++)
   Vds[j] = j* Vds_step;


Id[1][0]   = 0.00,         Id[2][0]   = 0.00,           Id[3][0]   = 0.00;
Id[1][1]   = 34.52,      Id[2][1]   = 21.71,       Id[3][1]   = 9.47;
Id[1][2]   = 64.03,      Id[2][2]   = 38.98,       Id[3][2]   = 15.59;
Id[1][3]   = 91.31,      Id[2][3]   = 56.24,       Id[3][3]   = 22.27;
Id[1][4]   = 118.04,        Id[2][4]   = 71.27,       Id[3][4]   = 26.73;
Id[1][5]   = 139.76,        Id[2][5]   = 80.73,       Id[3][5]   = 30.07;
Id[1][6]   = 158.69,        Id[2][6]   = 88.53,       Id[3][6]   = 33.41;
Id[1][7]   = 173.72,        Id[2][7]   = 94.65,       Id[3][7]   = 35.63;
Id[1][8]   = 183.74,        Id[2][8]   = 99.11,       Id[3][8]   = 37.86;
Id[1][9]   = 190.98,        Id[2][9]   = 102.45,         Id[3][9]   = 40.09;
Id[1][10] = 197.10,         Id[2][10] = 105.79,          Id[3][10] = 41.76;
Id[1][11] = 201.00,         Id[2][11] = 108.57,          Id[3][11] = 43.99;
Id[1][12] = 204.90,         Id[2][12] = 111.36,          Id[3][12] = 45.66;
Id[1][13] = 208.24,         Id[2][13] = 113.59,          Id[3][13] = 47.33;
Id[1][14] = 211.02,         Id[2][14] = 115.81,          Id[3][14] = 49.55;
Id[1][15] = 213.25,         Id[2][15] = 118.04,          Id[3][15] = 50.67;
Id[1][16] = 215.48,         Id[2][16] = 120.27,          Id[3][16] = 52.90;
Id[1][17] = 217.15,         Id[2][17] = 122.49,          Id[3][17] = 54.01;
Id[1][18] = 219.38,         Id[2][18] = 124.16,          Id[3][18] = 55.68;
Id[1][19] = 221.60,         Id[2][19] = 126.39,          Id[3][19] = 57.35;
Id[1][20] = 223.27,         Id[2][20] = 128.06,          Id[3][20] = 59.02;
Id[1][21] = 224.94,         Id[2][21] = 129.73,          Id[3][21] = 61.25;
Id[1][22] = 226.61,         Id[2][22] = 131.40,          Id[3][22] = 62.92;
Id[1][23] = 227.73,         Id[2][23] = 133.07,          Id[3][23] = 64.03;
Id[1][24] = 229.40,         Id[2][24] = 135.30,          Id[3][24] = 65.70;
Id[1][25] = 231.07,         Id[2][25] = 136.97,          Id[3][25] = 67.37;
Id[1][26] = 232.18,         Id[2][26] = 138.64,          Id[3][26] = 69.04;
Id[1][27] = 233.85,         Id[2][27] = 139.76,          Id[3][27] = 70.71;
Id[1][28] = 234.97,         Id[2][28] = 141.43,          Id[3][28] = 72.38;
Id[1][29] = 236.08,         Id[2][29] = 143.10,          Id[3][29] = 74.61;
Id[1][30] = 237.19,         Id[2][30] = 144.77,          Id[3][30] = 76.28;


Id[4][0]   = 0.00,          Id[5][0]   = 0.00;
Id[4][1]   = 1.00,          Id[5][1]   = 0.15;
Id[4][2]   = 2.00,          Id[5][2]   = 0.30;
Id[4][3]   = 3.00,          Id[5][3]   = 0.45;
```

```
    Id[4][4]  = 4.00,           Id[5][4]  = 0.59;
    Id[4][5]  = 5.01,           Id[5][5]  = 0.74;
    Id[4][6]  = 6.01,           Id[5][6]  = 0.89;
    Id[4][7]  = 7.01,           Id[5][7]  = 1.04;
    Id[4][8]  = 8.01,           Id[5][8]  = 1.19;
    Id[4][9]  = 9.01,           Id[5][9]  = 1.34;
    Id[4][10] = 10.02,          Id[5][10] = 1.49;
    Id[4][11] = 10.08,          Id[5][11] = 1.64;
    Id[4][12] = 11.58,          Id[5][12] = 1.78;
    Id[4][13] = 12.36,          Id[5][13] = 1.93;
    Id[4][14] = 13.14,          Id[5][14] = 2.08;
    Id[4][15] = 13.92,          Id[5][15] = 2.23;
    Id[4][16] = 15.03,          Id[5][16] = 2.56;
    Id[4][17] = 16.15,          Id[5][17] = 2.90;
    Id[4][18] = 17.26,          Id[5][18] = 3.23;
    Id[4][19] = 18.38,          Id[5][19] = 3.57;
    Id[4][20] = 19.49,          Id[5][20] = 3.90;
    Id[4][21] = 20.71,          Id[5][21] = 4.57;
    Id[4][22] = 21.94,          Id[5][22] = 5.24;
    Id[4][23] = 23.16,          Id[5][23] = 5.90;
    Id[4][24] = 24.39,          Id[5][24] = 6.57;
    Id[4][25] = 25.61,          Id[5][25] = 7.24;
    Id[4][26] = 27.06,          Id[5][26] = 8.24;
    Id[4][27] = 28.51,          Id[5][27] = 9.24;
    Id[4][28] = 29.95,          Id[5][28] = 10.25;
    Id[4][29] = 31.40,          Id[5][29] = 11.25;
    Id[4][30] = 32.85,          Id[5][30] = 12.25;


    alpha_approx  = 0.0;
    alpha_range   = 5.0;

    beta_approx   = 0.0;
    beta_range    = 5.0;

    gamma_approx  = 0.0;
    gamma_range   = 5.0;

    lambda_approx = 0.0;
    lambda_range  = 5.0;

//Range Selection:

    alpha  = 0.0;
    beta   = 0.0;
    gamma  = 0.0;
    lambda = 0.0;
```

```
//Alpha:

    alpha_initial  = alpha_approx - alpha_range;
    alpha_final    = alpha_approx + alpha_range;
    alpha_step     = alpha_range  / 10.0;

    MSE[1] = 1e10;

    for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
    {
       MSE[0] = 0.0;
       for (i = 1; i <= N; i++)
          for (j = 0; j < 31; j++)
          {
             Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) * tanh
(alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] + beta *
Vgs[i]);

             MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
          }
       if (MSE[0] < MSE[1])
          MSE[1] = MSE[0],
          optalpha = alpha;
    }
    alpha_range  = alpha_step;
    alpha_approx = optalpha;


//Beta:

    alpha = optalpha;

    beta_initial  = beta_approx - beta_range;
    beta_final    = beta_approx + beta_range;
    beta_step     = beta_range  / 10.0;

    MSE[1] = 1e10;

    for ( beta = beta_initial; beta <= beta_final; beta += beta_step)
    {
       MSE[0] = 0.0;
       for (i = 1; i <= N; i++)
          for (j = 0; j < 31; j++)
          {
             Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) * tanh
(alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] + beta *
Vgs[i]);
```

```
                MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
            }
        if (MSE[0] < MSE[1])
            MSE[1] = MSE[0],
            optbeta = beta;
    }
    beta_range  = beta_step;
    beta_approx = optbeta;

//Gamma:

    beta = optbeta;

    gamma_initial  = gamma_approx - gamma_range;
    gamma_final    = gamma_approx + gamma_range;
    gamma_step     = gamma_range  / 10.0;

    MSE[1] = 1e10;

    for ( gamma = gamma_initial; gamma <= gamma_final; gamma += gamma_step)
    {
        MSE[0] = 0.0;
        for (i = 1; i <= N; i++)
            for (j = 0; j < 31; j++)
            {
                Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) * tanh
(alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] + beta *
Vgs[i]);

                MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
            }
        if (MSE[0] < MSE[1])
            MSE[1] = MSE[0],
            optgamma = gamma;
    }
    gamma_range  = gamma_step;
    gamma_approx = optgamma;

//Lambda:

    gamma = optgamma;

    lambda_initial  = lambda_approx - lambda_range;
    lambda_final    = lambda_approx + lambda_range;
    lambda_step     = lambda_range  / 10.0;
```

```
  MSE[1] = 1e10;

  for ( lambda = lambda_initial; lambda <= lambda_final; lambda += lambda_step)
  {
    MSE[0] = 0.0;
    for (i = 1; i <= N; i++)
      for (j = 0; j < 31; j++)
      {
        Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2) * tanh
(alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] + beta *
Vgs[i]);

        MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
      }
    if (MSE[0] < MSE[1])
      MSE[1] = MSE[0],
      optlambda = lambda;
  }
  lambda_range  = lambda_step;
  lambda_approx = optlambda;


looping:
  alpha_initial  = alpha_approx - alpha_range;
  alpha_final    = alpha_approx + alpha_range;
  alpha_step     = alpha_range  / 5.0;
  beta_initial  = beta_approx -  beta_range;
  beta_final     = beta_approx +  beta_range;
  beta_step      = beta_range  /  5.0;

  gamma_initial  = gamma_approx - gamma_range;
  gamma_final    = gamma_approx + gamma_range;
  gamma_step     = gamma_range  / 5.0;

  lambda_initial = lambda_approx - lambda_range;
  lambda_final   = lambda_approx + lambda_range;
  lambda_step    = lambda_range  / 5.0;


  for ( alpha = alpha_initial; alpha <= alpha_final; alpha += alpha_step)
    for ( beta = beta_initial; beta <= beta_final; beta += beta_step)
      for  (  gamma  =  gamma_initial;  gamma  <=  gamma_final;  gamma  +=
gamma_step)
        for  (  lambda  =  lambda_initial;  lambda  <=  lambda_final;  lambda  +=
lambda_step)
        {
          MSE[0] = 0.0;
```

```
                    for (i = 1; i <= N; i++)
                        for (j = 0; j < 31; j++)
                        {

                            Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + gamma * Vds[j])),2)
* tanh (alpha * Vds[j] / (Vgs[i] - VT - gamma * Vds[j])) * ( 1 + lambda * Vds[j] +
beta * Vgs[i]);


                            MSE[0] += pow ((Id[i][j] - Id_cal[i][j]), 2);
                        }


                        if (MSE[0] < MIN && MSE[0] > 0.0)
                    {
                        MIN = MSE[0];
                        optalpha  = alpha;
                        optbeta   = beta;
                        optgamma  = gamma;
                        optlambda = lambda;
                    }
                    printf ("alpha : %0.4f\n", alpha);
                    printf ("beta  : %0.4f\n", beta);
                    printf ("gamma : %0.4f\n", gamma);
                    printf ("lambda: %0.4f\n", lambda);
                    printf ("MSE   : %0.4f\n", MSE[0]);
                    printf ("MIN   : %0.4f\n", MIN);
                }
    alpha_approx = optalpha;
    alpha_range  = alpha_step;

    beta_approx  = optbeta;
    beta_range   = beta_step;

    gamma_approx = optgamma;
    gamma_range  = gamma_step;

    lambda_approx = optlambda;
    lambda_range  = lambda_step;

    if (alpha_step > 0.001) goto looping;


//plotting:


    printf ("\nOptimized value of constants\n");
    printf ("--------------------------------\n");
    printf ("alpha : %0.4f\n", optalpha);
    printf ("beta  : %0.4f\n", optbeta);
    printf ("gamma : %0.4f\n", optgamma);
    printf ("lambda: %0.4f\n", optlambda);


    getch();


    for (i = 0; i <= N ; i++)
    {
        obsRMS[i] = 0.0;
        calRMS[i] = 0.0;
        MSE[i]    = 0.0;
    }

    for (i = 1 ; i <= N ; i++)
    {
        for (j = 0; j < 31; j++)
        {
            Id_cal[i][j] = Idss * pow((1 - Vgs[i] / (VT + optgamma * Vds[j])),2) * tanh
(optalpha * Vds[j] / (Vgs[i] - VT - optgamma * Vds[j])) * ( 1 + optlambda * Vds[j] +
optbeta * Vgs[i]);

            MSE[0]    += pow ((Id[i][j] - Id_cal[i][j]), 2);
            MSE[i]    += pow ((Id[i][j] - Id_cal[i][j]), 2);
            obsRMS[i] += pow (Id[i][j], 2);
            calRMS[i] += pow (Id_cal[i][j], 2);
            obsRMS[0] += pow (Id[i][j], 2);
            calRMS[0] += pow (Id_cal[i][j], 2);
        }

        MSE[i]     = sqrt ( MSE[i]   / 31.0);
        obsRMS[i]  = sqrt (obsRMS[i] / 31.0);
        calRMS[i]  = sqrt (calRMS[i] / 31.0);
    }

    MSE[0]    = sqrt ( MSE[0]    / ( N * 31.0));

    obsRMS[0] = sqrt ( obsRMS[0] / ( N * 31.0 ) );
    calRMS[0] = sqrt ( calRMS[0] / ( N * 31.0 ) );

    printf ("MSE calculated   : %0.4f\n", MSE[0]);
    printf ("Average RMS error: %0.4f\n", fabs ( calRMS[0] - obsRMS[0] ) );

    Vgss[0] = -3.0;
    for( i = 1 ; i < 36; i++)
        Vgss[i] = Vgss[i - 1] + 0.1;

    for( i = 1; i <= N ; i++ )
    {
        for( j = 0; j < 36; j++)
        {
            if((Vgss[j] - Vgs[i]) < Vds_step)
            n = j;
        }

        gm[n] = (Id[i][20] - Id[i+1][20])/(Vgs[i] - Vgs[i+1]);//20 is used for Vds = 2 v.
    }

    j = 20;
    for( i = 0; i < 36 ; i++ )
        gm_cal[i] = Idss * ( 2 *( 1 - Vgss[i] / (VT + optgamma * Vds[j])) * (- 1  / (VT +
optgamma * Vds[j])) * tanh (optalpha * Vds[j] / (Vgss[i] - VT - optgamma * Vds[j]))
* ( 1 + optlambda * Vds[j] + optbeta * Vgss[i]) + pow((1 - Vgss[i] / (VT + optgamma
* Vds[j])), 2) *  (1 - pow(tanh ( optalpha * Vds[j] / (Vgss[i] - VT - optgamma *
Vds[j])), 2)) * ( optalpha * (- Vds[j]) / pow((Vgss[i] - VT - optgamma * Vds[j]), 2)) *
( 1 + optlambda * Vds[j] + optbeta * Vgss[i]) + pow((1 - Vgss[i] / (VT + optgamma *
Vds[j])),2)  *  tanh  (optalpha  *  Vds[j]  /  (Vgss[i]  -  VT  -  optgamma  *  Vds[j]))  *
optbeta);

    i = 1;
    for ( j = 0; j < 31; j++)
    {
        gd[j+1]   = (Id[i][j+2]-Id[i][j])/(Vds_step*2);
        gd_cal[j] = Idss * (2 *( 1 - Vgs[i] / (VT + optgamma * Vds[j])) * (optgamma *
Vgs[i] / pow((VT + optgamma * Vds[j]), 2)) *  tanh (optalpha * Vds[j] / (Vgs[i] - VT
- optgamma   * Vds[j]))   * ( 1 + optlambda * Vds[j] + optbeta * Vgs[i]) + pow((1 -
Vgs[i] / (VT + optgamma * Vds[j])), 2) * (1 - pow( tanh ( optalpha * Vds[j] / (Vgs[i] -
VT - optgamma * Vds[j])), 2)) * (( optalpha * (Vgs[i] - VT - optgamma * Vds[j]) +
(optalpha * optgamma * Vds[j])) / pow ((Vgs[i] - VT - gamma * Vds[j]), 2)) * ( 1 +
optlambda  *  Vds[j]  +  optbeta  *  Vgs[i])  +  pow((1  -  Vgs[i]  /  (VT  +  optgamma  *
Vds[j])),2)  *  tanh  (optalpha  *  Vds[j]  /  (  Vgs[i]  -  VT  -  optgamma  *  Vds[j]))  *
optlambda );
    }

    FILE *fp;
    fp = fopen("optimal.xls", "w");
    fprintf (fp,"\t\t\tIdss =%6.3f mA / mm\n\t\t\tVT = %6.3fV\n\n",Idss, VT);
    fprintf (fp,"\t\tMSE Calculated = %0.4f\t\t\tAv. RMS Error = %0.4f\n\n", MSE[0],
fabs ( calRMS[0] - obsRMS[0] ) );
```

```
    fprintf (fp,"\t\tMSE[1]\tMSE[2]\tMSE[3]\tMSE[4]\tMSE[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",MSE[1], MSE[2], MSE[3],
MSE[4], MSE[5]);

    fprintf (fp,"\t\tobsRMS[1]\tobsRMS[2]\tobsRMS[3]\tobsRMS[4]\tobsRMS[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",obsRMS[1], obsRMS[2],
obsRMS[3], obsRMS[4], obsRMS[5]);

    fprintf (fp,"\t\tcalRMS[1]\tcalRMS[2]\tcalRMS[3]\tcalRMS[4]\tcalRMS[5]\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n",calRMS[1], calRMS[2],
calRMS[3], calRMS[4], calRMS[5]);

    fprintf (fp,"\t\tError1\tError2\tError3\tError4\tError5\n");
    fprintf (fp,"\t\t%0.4f\t%0.4f\t%0.4f\t%0.4f\t%0.4f\n\n", fabs(calRMS[1] -
obsRMS[1]), fabs(calRMS[2] - obsRMS[2]), fabs(calRMS[3] - obsRMS[3]),
fabs(calRMS[4] - obsRMS[4]), fabs(calRMS[5] - obsRMS[5]));

    fprintf (fp,"\tAlpha = %0.4f\t\tBeta = %0.4f\t\tGamma = %0.4f\t\tLambda =
%0.4f\n\n",optalpha, optbeta, optgamma, optlambda);

    fprintf
(fp,"Vds\tIds1(meas.)\tIds1(cal.)\tIds2(meas.)\tds2(cal.)\tIds3(meas.)\tIds3(cal.)\tIds4(
meas.)\tIds4(cal.)\tIds5(meas.)\tIds5(cal.)\n");
    for ( j = 0; j < 31; j++)
        fprintf(fp,"
%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t%0.2f\t\n",
Vds[j],Id[1][j],Id_cal[1][j],Id[2][j],Id_cal[2][j],Id[3][j],Id_cal[3][j],Id[4][j],Id_cal[4][j
],Id[5][j],Id_cal[5][j]);
    fprintf (fp,"\nVgss\tgm(meas.)\tgm(cal.)\n");
    for ( j = 0; j < 36; j++)
        fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vgss[j], gm[j], gm_cal[j]);
    fprintf (fp,"\nVds\tgd(meas.)\tgd(cal.)\n");
    for ( j = 0; j < 31; j++)
        fprintf(fp," %0.2f\t%0.2f\t%0.2f\n",Vds[j], gd[j], gd_cal[j]);

    fclose(fp);

    getch();
}
```