Norwegian University of
Science and Technology

Checking of Nanoscale Transistor Models

Marie Helene Hernes

Nanotechnology
Submission date: July 2016
Supervisor: Trond Ytterdal, IET
Co-supervisor: Per Christian Andresen, Nordic Semiconductor
Christian Hergot, Nordic Semiconductor

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

The transistor is a vital building block of modem CMOS semiconductor fabrication. As process geometries scale further down in the nanometer range, the supply voltages are reduced while at the same time the need for high performance computing increases. Accurate modeling of the transistor and parasitic effects is key to ensure successful products.

This assignment consisting of two parts. The first part being a study of nano-scale transistor behaviour modelling and defining and creating quantitative checks to ensure modelling is correct and according to known transistor physics (e.g. subthreshold and weak inversion behaviour, gm- and channel-conductance discontinuities, leakage-currents, noise, intrinsic gain, layout-effects like WPE/LOD, multi-finger devices etc.)

The second part has a more practical view, with focus on SPICE level simulation of transistor characteristics, extraction of data and implementation of automatic pass/fail checks based on the analysis in the first part. The code for data handling and checking may be implemented in Perl, Python, C, C++ or Java, depending on the candidate's skills.

Checking of Nanoscale Transistor Models

TFE4925 Nanotechnology, Master's Thesis
Marie Helene Gunnsdatter Hernes

July 8, 2016

Sammendrag

En enhetskompaktmodell er en matematisk beskrivelse av en enhet, for eksempel en transistor, i en integrert krets. Kompaktmodeller er designet for å være en del av en større simulering sammen med en kretsmodell. Derfor vil feil i enhetsmodellen øke og spre seg gjennom hele simuleringen. Å bekrefte at enhetsmodellene oppfører seg som forventet er dermed en forutsetning for å simulere integrerte kretser. I denne oppgaven har et fullstendig sett av kvalitative referansetester blitt utarbeidet, disse blir presentert i denne rapporten.

I forbindelse med enhetsmodelltesting kan flere hundre kurver bli produsert og disse må vurderes og sorteres etter om de oppfører seg etter fysikkens lover. For å forenkle denne prosessen er det i denne oppgaven blitt implementert et program som finner diskontinuiteter i kurver.

Summary

A device compact model is a mathematical description of a device, e.g. a transistor, in an integrated circuit. Compact models are designed to be a part of a larger simulation, and work together with a circuit model. For this reason errors in the device model will multiply throughout the simulation. Device model validation is thus an important prerequisite for simulating integrated circuits. In this thesis a comprehensive set of qualitative benchmark tests is developed and presented.

In the course of device model testing, several hundred curves can be produced and have to be evaluated in terms of whether they behave as expected from the laws of physics. To facilitate this process a program that finds discontinuities in curves was implemented for this thesis.

Preface

This report is written as part of a master's degree in Nanoelectronics at NTNU, with course code TFE4925. It was completed the spring of 2016. My supervisors were Professor Trond Ytterdal (Department of Electronics and Telecommunications at NTNU) and Per Christian Andresen (Nordic Semiconductor). My co-supervisors were Christian Hergot and Giancarlo Castaneda, both at Nordic Semiconductor.

For this thesis, simulations were done with a foundry model for a standard commercially available 65nm CMOS process. Simulations were done using Cadence, Virtuoso and ADE XL.

Contents

1 Introduction 1
  1.1 Transistor Technology 1
  1.2 Transistor Modeling 2
  1.3 Compact Model Validation 3
  1.4 Deep Learning 4
  1.5 Main Contributions 4
2 Metal-Oxide-Semiconductor Field-Effect Transistor 5
  2.1 nMOSFET 6
    2.1.1 Conductance 7
  2.2 Carrier Transport in Semiconductors 8
  2.3 Poisson's Equation 8
    2.3.1 Gradual Channel Approximation 9
    2.3.2 Applied Bias 10
    2.3.3 Poisson's Equation in Terms of Band-Bending Potential 10
    2.3.4 Charge Distribution 12
  2.4 Pao-Sah Integral 13
    2.4.1 Drain Current 14
  2.5 Short-Channel Effects 15
  2.6 Noise 16
    2.6.1 Thermal Noise 16
    2.6.2 Flicker Noise 17
3 Compact Models 19
  3.1 Threshold Voltage Based Formulation 19
    3.1.1 Threshold Voltage Modeling 19
    3.1.2 Short Channel Effect 21
    3.1.3 Strong Inversion Drain Current Modeling 21
    3.1.4 Subthreshold Drain Current Modeling 22
  3.2 Surface Potential Models 23
    3.2.1 Surface Potential Equation 23
    3.2.2 Charge-Sheet Approximation 24
    3.2.3 Symmetric Linearization for Drain Current Modeling 24
  3.3 Inversion Charge-Based Models 26
    3.3.1 Symmetric Linearization of Charge Density 26
    3.3.2 Drain Current Modeling 27
  3.4 Measurement-Based Modeling 29
  3.5 TCAD 29
4 Compact Model Validation 31
  4.1 V_g-sweep 31
  4.2 V_{DS}-sweep 31
    4.2.1 Transcapacitances 31
  4.3 Gummel Symmetry 34

4.4 Length and Width Characteristics . . . 34
4.5 Leakage Current Characteristics . . . 34
4.6 Charge Conservation . . . 37
4.7 Non-Quasi-Static Operation . . . 37
4.8 Noise . . . 37
5 Deep Learning . . . 39
5.1 MNIST . . . 41
5.2 Restricted Boltzmann Machine . . . 41
6 Results and Discussion . . . 43
6.1 Benchmarking Compact Model Validation . . . 43
6.2 Quantitative Checking of Compact Models . . . 43
6.3 Test Assessment Automation . . . 44
6.3.1 Deep Learning Algorithm . . . 44
6.3.2 Python Script for Test Differentiation . . . 45
6.3.3 Comparison of Deep Learning Algorithm with Conventional Algorithm . . . 46
7 Conclusion . . . 47
8 Bibliography . . . 49
A Appendix List of Symbols . . . 55
B Appendix Source Code . . . 57

1 Introduction

1.1 Transistor Technology

The metal-oxide-semiconductor field-effect transistor (MOSFET) is a key component in the microelectronics industry. Transistors are used to amplify signals or as electronic switches. They are the building blocks of memory chips, processors and microcircuits and this makes them one of the most important technological advances of the 20th century.

The microelectronics industry is constantly pushing down the transistor size. Gate miniaturization has several advantages, the most obvious being an increased density of transistors on a chip. This in turn gives the chip increased functionality, assuming a constant chip size. In addition, down-scaling the devices increases switching speed and reduces power dissipation[1]. Also, the device threshold voltage is lowered, in theory allowing the supply voltage to be reduced. However, as the device switching behavior becomes poorer when the gate is shortened, thus the predicted lowering of supply voltage has not been achieved[2].

In 1965 Gordon Moore predicted that the density of transistors on a chip would double every year[3]. The prediction proved a useful tool and is now called Moore's Law. Later, this prediction was adjusted to a doubling every two years or 18 months, in any case a dramatic pace of minimization, and one that has become more or less a self-fulfilling prophecy. To ensure a continued and coordinated progression, the International Technology Roadmap for Semiconductors (ITRS) is agreed upon by a group of semiconductor industry experts[4]. The ITRS is a set of documents carrying opinions on the directions of research in different areas, notably size scales called nodes. The roadmap is meant to give an idea of when a certain capability will be needed in the market, so that suppliers are synchronized. The ITRS has been consistent with Moore's law, but as the 5 nm node approaches (due 2020–2021), down-scaling bulk silicon technology is increasingly difficult. Down-scaling is challenging both in terms of device fabrication and operation. As transistors get smaller and channels shorter, undesirable effects known as short channel effects (SCE) become observable. Notably, leakage current increases in the "off"-state of the transistor, which increases device power consumption. Alternative technologies are hence being researched.

1.2 Transistor Modeling

When MOSFET devices were introduced during the 1960s[5] it became necessary to model their behavior. A transistor model should be a mathematical description of the transistor's current-voltage (I-V) and capacitance-voltage (C-V) behavior and the charge carrier transport through the device[6]. The term compact device model is used to describe a device model that is designed to be implemented with a circuit model such as SPICE (Simulation Program with Integrated Circuit Emphasis)[7]. That is, a compact model should balance output parameter accuracy with computational power required to run simulations.

Early transistors were mainly operated in strong inversion. Considering the transistor to be turned on above a certain gate voltage, called the threshold voltage, and turned off below this voltage, a simple model is obtained[8]. Models based on this inference are called threshold voltage-based models. In this scheme, the current through the channel is considered to be zero for all voltages below the threshold voltage, therefore this method is only suitable for transistors operated in strong inversion.

When moderate and weak inversion scenarios became relevant, the threshold voltage-based models were updated with new equations to describe them. These regions have become more important in the last twenty years because mixing of analog and digital chips has become industry standard[9]. Additionally, operation voltage and channel length of devices have decreased. With this evolution, more equations have been added to threshold voltage-based models and they have consequently become more complex[10], needing many input variables.

A model that describes the device for all biasing conditions with the same physical state variable has several advantages over threshold voltage-based models. One approach is using the surface potential ϕ_s as the state variable. The surface potential is implicitly defined through the Pao-Sah integral[11] for the entire range of the MOSFET operation. However, this integral can only be solved numerically and thus requires extensive computation power. Combining the Pao-Sah integral with the charge-sheet approximation is one way of reducing computational power and time needed to execute simulations. In the charge-sheet approximation the inversion layer of the MOSFET is assumed to be infinitesimally thin, and hence constitutes a two-dimensional conducting plane[12]. Models that calculate ϕ_s at the two ends of the device channel are called surface potential-based models. ϕ_s is in turn used as a basis for determining parameters such as terminal charges, currents and derivatives.

A different approach to MOSFET modeling is the charge-based or inversion-charge-based model. As the ϕ_s-based models require computationally demanding mathematical solution techniques, expressing the device drain current in terms of the inversion charge density Q_i provides a slightly less accurate model which still has the advantage of being identically defined for all applied voltages[13]. This represents a middle ground between the less exact threshold voltage-based models and the more physically correct surface potential-based models. Q_i at the device channel ends is subsequently used to calculate the surface potential. Consequently the charge-based models are a variation of surface potential calculation, but different enough that it is conventionally its own category of compact models[10].

1.3 Compact Model Validation

Compact device models are designed to interface with an enveloping network of models that simulate the entire integrated circuit (IC). Errors in the device model will affect the related models and deteriorate the simulation results. Ergo checking of compact models is fundamental and should be done in a methodical and reproducible way. Today, compact models, IC-models and the actual ICs may be manufactured by different companies and foundries. This complex industry situation makes model standardization a prerequisite for an unimpeded IC-design and optimization process.

Compact model validation consists of choosing a set of benchmark tests, designating corresponding pass and fail criteria, and conducting the tests in an efficient and reproducible way. There is no such thing as a definite set of benchmark tests. As IC application evolves, new tests emerge to ensure model validity in all relevant model outputs. On the other hand, as compact models are updated, better solutions to some problem areas imply that some tests are no longer needed. When it comes to test execution and pass-fail determination, approaches vary.

As compact modeling became more complicated in the 1990s, the need for model validation and standardization became apparent. A set of benchmark tests for analog systems was produced by Tsividis and Suyama in 1994([14]). This set addressed some of the problems that arose when models designed for transistors in strong inversion were applied in the weaker inversion regimes required by analogue circuits. The Tsividis-Suyama tests are still used in compact model validation [15].

In 1995, the Compact Model Coalition (CMC) was established with the goal of developing and standardizing compact device models[16]. Ever since, this organization has provided for compact models in the public domain, such as BSIM and HiSIM. In 1997 a large set of benchmark tests was set as a standard by the CMC[17]. This set of tests contains 22 qualitative and 12 quantitative tests (encompassing the Tsividis-Suyama-tests) and is still used as a starting point for compact model validation. The last decade has seen an increase in radio-frequency (RF) use, altering the need of benchmark tests. Although the same tests are used, more focus is directed towards non-quasi-static operation[18] and validation of device capacitances[19].

1.4 Deep Learning

Deep learning, also known as machine learning or neural networks, allows computer models to recognize patterns at a higher level of abstraction than conventional methods[20]. Deep learning-models utilize several levels of processing networks to recognize patterns. Examples of usage include object detection in images[21], speech recognition[22] and language translation[23].

The first ideas on deep learning were greatly inspired by neural networks in brains, proposing connected nodes that act like synapses[24]. Simple and complex cells found in the cat's visual cortex were in the 1960s found to fire in response to certain visual impressions, with the complex cells having less sensitivity to spacial invariance than the simple cells[25]. This later inspired convolutional neural networks, networks that respond to overlapping areas in an image (or visual field) which make them better at disregarding object position[26].

When error-efficient back propagation was introduced in deep learning algorithms, fewer training iterations were needed to obtain equally good results[27]. Error-efficient back propagation includes calculating the error difference before and after changing the node values in the model and thereby finding the most efficient next step for the model parameters.

More recently, increased computational power and better programs have enabled several applications such as object detection in large images of several million pixels[28]. One example is detecting the number of cells in mitosis in breast cancer scan images, an important parameter in breast cancer prognosis[29].

Concurrently deep learning programming languages and libraries have become available which has made deep learning more accessible. One example of this is Google's Tensorflow, a library for machine learning that utilizes graphs to represent visible and hidden nodes[30]. Also, various Python modules such as Theano[31] has simplified designing neural networks for personal computers.

1.5 Main Contributions

The main contribution of this work is an updated, comprehensive list of qualitative benchmark tests for compact device model validation. In addition, a function that iterates through a set of data points and finds curve discontinuities has been implemented. This in order to simplify the process of determining whether each individual curve generated by the device model abides by the laws of physics.

2 Metal-Oxide-Semiconductor Field-Effect Transistor

A transistor is an electrical device used to switch or amplify an electronic signal. In modern computer science, transistors represent the zeroes and ones that constitute the foundation of all computer technology.

A field effect transistor (FET) consists of a source and drain, connected by a gate[32]. Any voltage applied to the gate induces a channel between the source and drain and thus allows a current to flow between the two contacts.

To fabricate a FET, a doped semiconductor substrate with two oppositely doped wells is used. The two wells are given ohmic (metal) contacts and are called the source and drain. Between them, an oxide with the gate contact is positioned. When a voltage is applied to the device gate, an electric field protrudes from it into the substrate. Because the gate is isolated from the substrate by an oxide, the charge will build up in the gate, giving rise to a stationary electric field. This field will in turn attract charge carriers with the opposite charge of the substrate dopant charge carriers, which will create a layer of inverted charge in the channel, called an inversion layer. When the gate voltage exceeds the threshold voltage V_T, a current flows between the drain and source. This is called the field effect[33]. In figure 1 the MOSFET voltages and capacitances are illustrated.

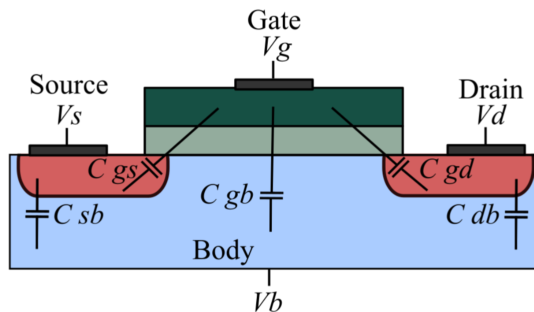

Figure 1: Voltages and transcapacitances of a MOSFET.

2.1 nMOSFET

Assume a p-doped substrate, known as a nMOSFET or n-channel MOSFET, illustrated in figure 2. When a positive voltage V_g is applied on the gate, positive charge carriers in the substrate will be pushed away from it, creating a depletion zone with no charge carriers in the substrate directly below the gate. This allows negative charge carriers from the source to enter this zone and flow through the transistor creating a current I_d between the source and drain, assuming that there is a potential difference between the source and drain. This constitutes an inversion channel. The minimum gate voltage at which current flows between the source and drain is called the threshold voltage, V_T. As the gate voltage increases, so does the source-drain current until it reaches a saturation current I_{sat}. This is illustrated in figure 3a.

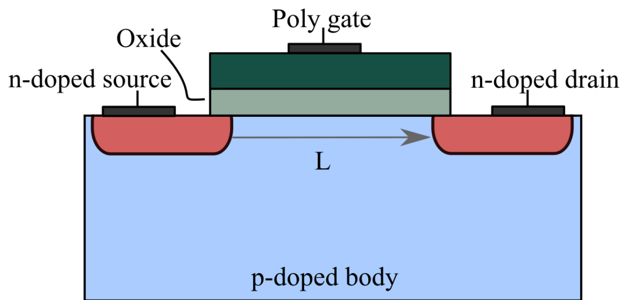

Figure 2: nMOSFET.

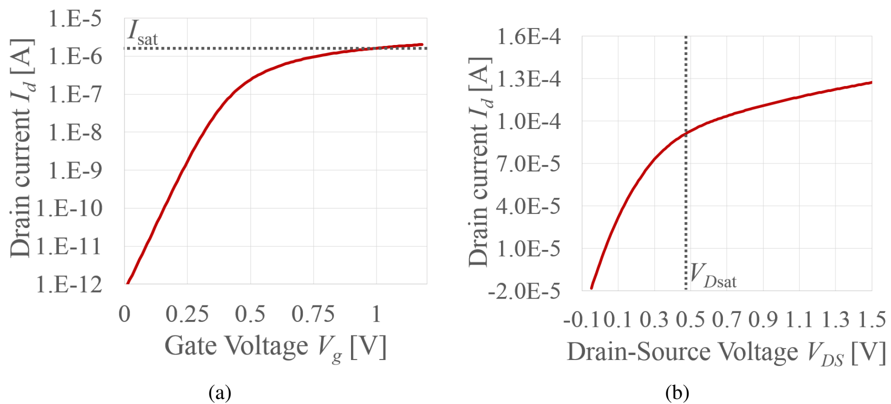

Figure 3: Current saturation in MOSFET.

To drive a current through the device channel, a potential between the source and drain V_{DS} is applied. However, if V_{DS} is too high, positive charge carriers are attracted to the drain end of the device channel and generate a depletion zone between the n-doped wells and inversion channel, and the p-doped substrate. This can in turn pinch off the inversion channel, which saturates the drain current. The depletion zone is illustrated in figure 4. The channel pinch-off happens when V_{DS} reaches $V_{D\text{sat}}$, given by[32]

$$V_{D\text{sat}} = \frac{V_g - V_T}{m} \tag{1}$$

where m is the bulk charge factor. The drain current dependency on V_{DS} is plotted in figure 3b. The linear region of V_{DS} is called the triode (or ohmic) region. The transistor is normally operated at a constant supply voltage V_{dd}.

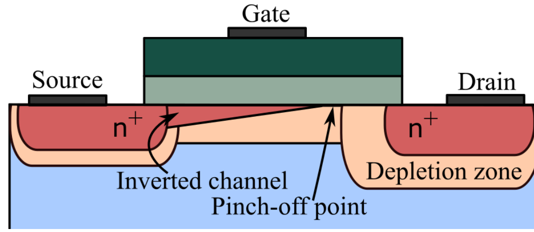

Figure 4: Pinch-off of the inverted channel due to a very high V_{DS}.

2.1.1 Conductance

The device response in terms of drain current when applying a gate voltage is called the transconductance or mutual conductance g_m and is defined as[34]

$$g_m = \left.\frac{\partial I_d}{\partial V_g}\right|_{V_{DS}} . \tag{2}$$

Similarly the device output conductance g_o is defined as the drain current response when the source-drain voltage is changed:

$$g_o = \left.\frac{\partial I_d}{\partial V_{DS}}\right|_{V_g} . \tag{3}$$

2.2 Carrier Transport in Semiconductors

Carrier transport in a semiconductor is the result of two mechanisms; diffusion and drift. As in all materials diffusion of charge carriers can give rise to a current. In addition a present electric field will produce a drift current[10].

$$I_d = I_{\text{diff}} + I_{\text{drift}} \tag{4}$$

At low E-fields, the relation between applied field E and drift velocity v is defined as charge carrier mobility μ:

$$v = \mu E. \tag{5}$$

At higher E-fields the charge carrier velocity becomes saturated. This happens because scattering events become more frequent as the electron velocity increases, thus the electrons are slowed down and their velocity can no longer increase. The saturated velocity is called drift velocity v_d and can be approximated as [32]

$$v_d \approx \frac{\mu_s E}{1 + \frac{E}{E_c}}. \tag{6}$$

Here μ_s is the surface carrier mobility and E_c is the critical electric field at which the velocity saturates.

2.3 Poisson's Equation

Poisson's equation is a differential equation that governs MOSFET operation because it relates charge density to electric potential[10]. It is based on Maxwell's equations and is stated as

$$\nabla^2 \phi = -\frac{\rho}{\varepsilon_s}. \tag{7}$$

Here, ϕ is the electrostatic potential, ρ is the charge density and ε_s is the semiconductor permittivity.

2.3.1 Gradual Channel Approximation

MOSFETs are 3D devices, but except for very small-scale geometry inferring that ϕ variations along the channel (in the y-direction) are much smaller than variations down through the substrate (in x-direction) is a useful assumption. (The coordinate system is illustrated in figure 5.)

$$\left|\frac{\partial^2 \phi}{\partial y^2}\right| \ll \left|\frac{\partial^2 \phi}{\partial x^2}\right| \tag{8}$$

Equation 8 is called the gradual channel approximation[10] and allows for separating Poisson's equation (equation 7) in the x- and y-direction. The surface potential-charge density relation is then Poisson's equation for the x-direction:

$$\frac{\mathrm{d}^2 \phi}{\mathrm{d}^2 x} = -\frac{\rho}{\varepsilon_s}. \tag{9}$$

Integrating equation 9 we obtain Gauss' law[35]:

$$E = \frac{1}{\varepsilon_s} \int \rho \, dx = \frac{Q_s}{\varepsilon_s} \tag{10}$$

where Q_s is the total induced charge, or space charge. This law is only valid for devices where space charge neutrality is upheld, that is when the semiconductor in sum has $\rho = 0$.

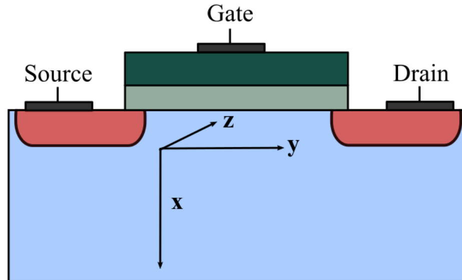

Figure 5: MOSFET showing the **x**, **y** and **z** coordinate system.

2.3.2 Applied Bias

When applying a bias at the oxide gate the potential drop will be shared between a voltage across the gate oxide V_{ox}, a surface potential at the gate oxide–semiconductor interface, and the flatband voltage V_{fb}[10]:

$$V_g = V_{ox} + \phi_s + V_{fb}. \tag{11}$$

The flatband voltage is the voltage required to make the energy bands in the semiconductor flat at the oxide-semiconductor interface[36]. By applying Gauss' law on both sides of the interface, we get

$$Q_s = -\varepsilon_s E_s \tag{12}$$

and

$$Q_g = \varepsilon_{ox} E_{ox} = V_{ox} C_{ox}, \tag{13}$$

where E_s is the surface electric field, Q_g is the gate charge and C_{ox} is the oxide capacitance per unit area. Assuming an ideal oxide, the induced charges across the oxide-semiconductor interface must balance each other; $Q_g = -Q_s$[10]. This gives

$$Q_s = -Q_g = -V_{ox} C_{ox}. \tag{14}$$

$$V_{ox} = -\frac{Q_s}{C_{ox}} \tag{15}$$

Substituting equation 15 into equation 11, V_g becomes

$$V_g = V_{fb} + \phi_s - \frac{Q_s}{C_{ox}}. \tag{16}$$

2.3.3 Poisson's Equation in Terms of Band-Bending Potential

To express Poisson's equation in terms of ϕ, ρ should be expressed as a function of electrostatic potential. Assuming that all dopants are ionized, ρ can be written as [10]

$$\rho(x) = q\left(p(x) - n(x) + N_d(x) - N_a(x)\right), \tag{17}$$

where q is the elementary charge, p and n are the free electron and hole concentrations and N_a and N_d are the acceptor and donor concentrations. Substituting ρ, Poisson's equation can be written as

$$\frac{d^2\phi}{dx^2} = \frac{-q}{\varepsilon_s}\left(p(x) - n(x) + N_d(x) - N_a(x)\right). \tag{18}$$

In an n-type semiconductor the Fermi potential ϕ_f lies above the intrinsic potential ϕ_i, hence the dopant concentration in the material can be shown to be[10].

$$N_d = n_i \exp\left(\frac{q(\phi_i - \phi_f)}{k_B T}\right). \tag{19}$$

Here n_i is the intrinsic carrier concentration, k_B is the Boltzmann constant and T is temperature. At room temperature the available thermal energy is sufficient to ionize all dopant atoms in the semiconductor material[10]. In nMOSFETs, it is thus safe to assume that $n = N_d$. Accordingly we can assume that the hole concentration in this n-type material, p_n, is given by

$$p_n = \frac{n_i^2}{N_d} = n_i \exp\left(\frac{q(\phi_i - \phi_f)}{k_B T}\right). \tag{20}$$

The bulk potential ϕ_b in n-type semiconductors is defined by re-arranging equation 19:

$$\phi_b = \phi_f - \phi_i = -\frac{k_B T}{q} \ln\left(\frac{N_d}{n_i}\right). \tag{21}$$

And in its general form:

$$\phi_b = \phi_i - \phi_f = \pm U_T \ln\left(\frac{N_b}{n_i}\right). \tag{22}$$

N_b is the dopant concentration, chosen according to material type and U_T is the thermal potential.

Expressing n, p, N_a and N_d in terms of band-bending potentials ϕ will aid in finding Q_s. Here, we will have to include $\phi(x)$ in equation 20.

$$p(x) = n_i \exp\left(\frac{\phi_f - \phi_i - \phi(x)}{U_T}\right) = n_i \exp\left(\frac{\phi_f - \phi_i}{U_T}\right) \exp\left(\frac{\phi(x)}{U_T}\right) = N_a \exp\left(\frac{\phi(x)}{U_T}\right) \tag{23}$$

In the last step equation 21 has been used. The minority carrier concentration n_p can be expressed analogous to equation 20 as:

$$n_p = \frac{n_i^2}{p(x)} = \frac{n_i^2}{N_a} \exp\left(\frac{\phi(x)}{U_T}\right) \tag{24}$$

Substituting p from equation 23 and n from equation 24 into equation 17 we get

$$\rho(x) = q\left(N_a \exp\left(\frac{-\phi(x)}{U_T}\right) - \frac{n_i^2}{N_a} \exp\left(\frac{\phi(x)}{U_T}\right) - \frac{n_i^2}{N_a} - N_a\right) \tag{25}$$

In the last step the substrate is assumed to be uniformly doped. This rends the Poisson equation in terms of surface potential as

$$\frac{d^2\phi(x)}{dx^2} = -\frac{q}{\varepsilon_s}\left(N_a\left(\exp\left(\frac{-\phi(x)}{U_T}\right) - 1\right) - \frac{n_i^2}{N_a}\left(\exp\left(\frac{-\phi(x)}{U_T}\right) - 1\right)\right). \tag{26}$$

2.3.4 Charge Distribution

To obtain Q_s we want to solve Poisson's equation. By multiplying both sides of equation 26 by $2\frac{d\phi(x)}{dx}$ and integrating from 0 to $\phi(x)$, we get[10]

$$\begin{aligned}\left(\frac{d\phi(x)}{dx}\right)^2 &= E_s^2 \\ &= \frac{2qN_aU_T}{\varepsilon_s}\left(\left(\exp\left(\frac{-\phi(x)}{U_T}\right)+\frac{\phi(x)}{U_T}-1\right)+\frac{n_i^2}{N_a}\left(\exp\left(\frac{\phi(x)}{U_T}\right)-\frac{\phi(x)}{U_T}-1\right)\right)\end{aligned} \tag{27}$$

At the channel-gate oxide interface (in $x = 0$) $\phi(x) = \phi_s$ and $E = E_s$. This, combined with Gauss' law (equation 10) and equation 27 gives the charge per unit area on the surface:

$$Q_s = \pm\sqrt{2q\varepsilon_s N_a U_T}\left(\left(\exp\left(\frac{-\phi_s}{U_T}\right)+\frac{\phi_s}{U_T}-1\right)+\frac{n_i^2}{N_a}\left(\exp\left(\frac{\phi_s}{U_T}\right)-\frac{\phi_s}{U_T}-1\right)\right)^{\frac{1}{2}}. \tag{28}$$

2.4 Pao-Sah Integral

The Pao-Sah integral relates the drain current with the physical parameter ϕ_s and is valid for all biasing conditions. The mobile minority charge carrier density in the inversion layer is defined as[10]

$$Q_i = -q \int_0^{\infty} n \, dx. \tag{29}$$

To evaluate the integral in equation 29, we integrate over ϕ rather than x. In $x = 0$, $\phi = \phi_s$ and in $x = \infty$, $\phi = \phi_b$. (When $\phi > \phi_b$, the majority carrier concentration exceeds the minority carrier concentration and there is no longer a charge carrier inversion, meaning that the inversion layer ends.)

$$Q_i = -q \int_{\phi_s}^{\phi_b} n \frac{dx}{d\phi} \, d\phi. \tag{30}$$

n can be expressed by combining equations 20 and 21:

$$n = N_d \exp\left(\frac{\phi - \phi_b - V_{ch}}{U_T}\right). \tag{31}$$

V_{ch} is the channel potential due to applied V_{DS}, defined as the difference between the equilibrium Fermi energy levels in the source and drain. From Gauss' law (equation 10) we can find $\frac{dx}{d\phi} \cdot \frac{d\phi}{dx} = E$, thus

$$\frac{dx}{d\phi} = \frac{\varepsilon_s}{Q_s}. \tag{32}$$

Q_s we adapt from equation 28. In strong inversion the term $\exp\left(\frac{\phi_s}{U_T}\right)$ is much larger than both -1 and the term $\exp\left(\frac{-\phi_s}{U_T}\right)$, thus we disregard them and Q_s in strong inversion becomes:

$$\begin{aligned} Q_s &= -\sqrt{2q\varepsilon_s N_a U_T}\left(\phi_s + U_T \frac{n_i^2}{N_a} \exp\left(\frac{\phi_s - V_{ch}}{U_T}\right)\right)^{\frac{1}{2}} \\ &= -\sqrt{2q\varepsilon_s N_a U_T}\left(\phi_s + U_T \exp\left(\frac{\phi_s - 2\phi_b - V_{ch}}{U_T}\right)\right)^{\frac{1}{2}} \end{aligned} \tag{33}$$

where equation 24 was used in the last step. Now, substituting n from equation 31 and $\frac{dx}{d\phi}$ from equation 32 into the integral in equation 30 we get

$$Q_i = \sqrt{\frac{\varepsilon_s N_d q}{2}} \int_{\phi_s}^{\phi_b} \frac{\exp\left(\frac{\phi - 2\phi_b - V_{ch}}{U_T}\right)}{\left(\phi_s + U_T \exp\left(\frac{\phi_s - 2\phi_b - V_{ch}}{U_T}\right)\right)^{\frac{1}{2}}} d\phi. \tag{34}$$

2.4.1 Drain Current

In this section the drain current will be expressed in terms of Q_i. Start with the current density equation [10]:

$$J = J_n + J_p = -qn\mu_n F_n - qp\mu_p F_p \tag{35}$$

where J_n and J_p are the current densities of electrons and holes, μ_n and μ_p are their mobilities and F_n and F_p are their imrefs (quasi Fermi levels). Define the current continuity equations, that is simply define the recombination rates (r_r) and the generation (r_g) for electrons and holes;

$$\frac{\partial n}{\partial t} = \frac{1}{q}\nabla J_n - r_{rn} + r_{gn} \tag{36}$$

$$\frac{\partial p}{\partial t} = \frac{1}{q}\nabla J_n - r_{rp} + r_{gp} \tag{37}$$

Assuming that, on average, there is no generation or recombination of charge carriers, the continuity relations reduce to

$$\nabla J_n = 0 = \nabla J_p. \tag{38}$$

Taking nMOSFETs as an example, we presume that only minority carriers contribute to I_d. Assuming that current only flows along the channel, the quasi Fermi potential is constant in x-direction. In nMOSFETs the current density then becomes[10]

$$J_n = -qn\mu_n \frac{\partial F_n}{\partial y}. \tag{39}$$

Integrating the current density we get the drain current;

$$I_d = -W \int_0^\infty qn\mu_s \frac{\partial F_n}{\partial y} dx \tag{40}$$

W is the channel width and μ_s is the surface electron mobility, which is not equal to the bulk electron mobility μ_n. Substituting $\frac{\partial F_n}{\partial y}$ with V_{ch} we get

$$I_d = -W\mu_s \frac{dV_{ch}}{dy} \int_0^\infty qn\, dx. \tag{41}$$

Using the definition of Q_i from equation 29, equation 41 becomes:

$$I_d dy = W\mu_s Q_i dV_{ch} \tag{42}$$

This finally gives the current in terms of Q_i;

$$I_d = \mu_s \frac{W}{L} \int_0^{V_{DS}} Q_i\, dV_{ch} \tag{43}$$

And applying equation 29 we get Pao-Sah's integral:

$$I_d = \mu_s \frac{W}{L} \sqrt{\frac{\varepsilon_s N_d q}{2}} \int_0^{V_{DS}} \int_{\phi_s}^{\phi_b} \frac{\exp\left(\frac{\phi - 2\phi_b - V_{ch}}{U_T}\right)}{\left(\phi_s + U_T \exp\left(\frac{\phi_s - 2\phi_b - V_{ch}}{U_T}\right)\right)^{\frac{1}{2}}} d\phi\, dV_{ch} \tag{44}$$

2.5 Short-Channel Effects

When down-scaling the channel length of the MOSFET certain undesirable effects become noticeable. Short-channel effects arise when the channel length approaches the depletion layer's width, and this generates higher leakage current in the device[32].

A perfect switch has a direct transition from its "off"-state to its "on"-state in the threshold voltage. However, real transistors always have a small current between the absolute "off"- and "on"-states. This is due to the subthreshold slope of I_d, the SS. The subthreshold slope can be defined as[37]

$$SS = \frac{\mathrm{d}V_g}{\mathrm{d}\log I_d}. \tag{45}$$

The subthreshold slope is controlled by the leakage current in the device, that is the current flowing between the source and drain when the device is in its off state. It is thus a measure of the efficiency with which the gate voltage controls the channel current. The subtheshold slope is illustrated in figure 6.

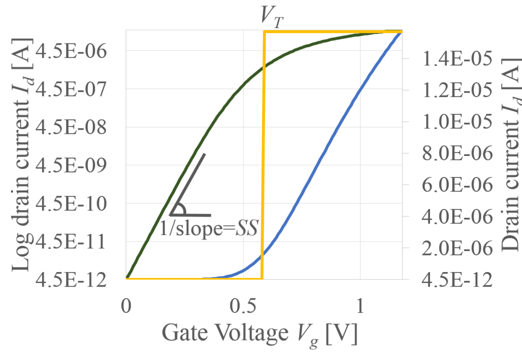

Figure 6: IV-curve for a perfect switch compared to a real transistor. Any real transistor will have a subthreshold slope, as indicated on the plot.

2.6 Noise

Noise is defined as an unwanted signal and will always be present in integrated circuits[38]. Noise can be divided into device noise, produced in the device, and external noise which is created by sources outside of the device such as capacitative and inductive interferences in the interconnect network[32]. The following will be focused on device noise.

Device noise comes from the random behavior of the charge carriers in the transistor, and the two main noise sources are thermal noise and flicker noise[10]. Examples of other sources of noise are any non-uniformity in the doping of the substrate, and leakage current from the source and drain (shot noise).

To characterize noise, the mean square value of the noise signal at narrow frequency bandwidths Δf can be analyzed. When the bandwidth approaches zero, this mean value is called the power spectral density S[10]. To find the total mean square noise within any bandwidth $[f_1, f_2]$ the power spectra density is summed up for each sub-bandwidth. For the mean square noise current $\overline{i_n^2}$ this is

$$\overline{i_n^2} = \int_{f_1}^{f_2} S_i(f)\, df. \tag{46}$$

For the mean square voltage noise $\overline{v_n^2}$ the equation becomes

$$\overline{v_n^2} = \int_{f_1}^{f_2} S_v(f)\, df. \tag{47}$$

2.6.1 Thermal Noise

Random thermal motion of charge carriers result in a uniform noise called thermal noise[39]. The voltage thermal noise generated across a resistor R can be written as[10]

$$\overline{v_t^2} = 4k_B T R. \tag{48}$$

To find the thermal noise in a MOSFET its channel is partitioned into small sections Δy, each with the resistance ΔR[10]:

$$\Delta R = \frac{\Delta y}{\mu W_{\text{eff}} Q_i}. \tag{49}$$

Here W_{eff} is the effective channel width. The voltage root mean square noise of each segment is then

$$\overline{\Delta v_t^2} = \frac{4k_B T \Delta f \Delta y}{\mu W_{\text{eff}} Q_i}. \tag{50}$$

This gives a current change of

$$\overline{\Delta i_t^2} = \left(\frac{W_{\text{eff}} \mu Q_i}{L_{\text{eff}}}\right)^2 \overline{\Delta v_t^2} = 4k_B T \frac{W_{\text{eff}} \mu Q_i}{L_{\text{eff}}^2} \Delta y \Delta f. \tag{51}$$

Integrating along the channel the total noise current power can be obtained:

$$\overline{\Delta i_t^2} = 4k_B T \Delta f \frac{\mu}{L_{\text{eff}}^2} \int_0^{L_{\text{eff}}} Q_i W_{\text{eff}}\, dy = 4k_B T \frac{\mu Q_i}{L_{\text{eff}}^2} \Delta f. \tag{52}$$

This gives the power spectral density as

$$S_i(f) = \frac{\overline{\Delta i_t^2}}{\Delta f} = 4k_B T \frac{\mu Q_i}{L_{\text{eff}}^2}. \tag{53}$$

2.6.2 Flicker Noise

The flicker noise is also known as $1/f$ noise and is inversely proportional to f. It originates in the fluctuating mobility function of charge carriers in the channel that again comes from electron traps in the gate dielectric[40]. Depending on the transistor W and L the number of electron traps present can be very small, resulting in a discrete noise signal called a random telegraph noise. For larger MOSFETs a large number of traps contribute to the flicker noise. The noise mean square can then be shown to be [32]

$$\overline{i_f^2} = k_B T \frac{K_f W_{\text{eff}}}{L_{\text{eff}}^2 C_{ox}} \left(\frac{I_d}{W_{\text{eff}}}\right)^{A_f} \frac{\Delta f}{f} \tag{54}$$

where K_f is a constant, proportional to the number of traps in the device oxide, and A_f is a constant for the importance of Coulomb scatting on the carrier mobility. Again the power spectral density becomes

$$S_i(f) = k_B T \frac{K_f W_{\text{eff}}}{L_{\text{eff}}^2 C_{ox}} \left(\frac{I_d}{W_{\text{eff}}}\right)^{A_f} \frac{1}{f}. \tag{55}$$

Since the flicker noise is reversely proportional to the frequency can it be disregarded for high frequencies, e.g. above 100 MHz[32].

3 Compact Models

Compact models are mathematical descriptions of circuit components, notably transistors, that are used together with other models in integrated circuit design and analysis. Thus, compact models are an indispensable component in computer-assisted design. Because compact models are designed to be used as part of a network of models, simple models requiring little computational power are preferred.

3.1 Threshold Voltage Based Formulation

As a first approach to transistor modeling, the threshold voltage is assumed to be a point where the current flowing through the device abruptly changes from zero to the "on"-current[10]. This approach requires that the threshold voltage be modeled, so V_T will be the first parameter explored. Subsequently other output parameters such as the charge carrier mobility, the drift velocity and the drain current can be modeled. Examples of threshold voltage based models are BSIM3[41], BSIM4[42] and MOS Model 9[43]. Most models in this chapter are adapted from BSIM3.

3.1.1 Threshold Voltage Modeling

Define the threshold voltage V_T as the voltage above which a channel inversion layer exists. When the applied gate voltage is below threshold, only leakage current is flowing through the channel and it can thus be approximated that $I_d \approx 0$. When applying a V_{DS}, a surface potential will arise in the device channel, depending on location; $\phi_s(y)$. Accordingly a channel potential exists along the channel such that[41]

$$V_{ch}(y) = \begin{cases} V_{SB} & \text{at } y = 0 \\ V_{SB} + V_{DS} & \text{at } y = l \end{cases} \tag{56}$$

where V_{SB} is the potential difference between the device source and body.(As a first approach, the substrate doping is assumed to be uniform, and the device geometry large enough that edge effects can be ignored.)

To obtain the total charge in the device, the Poisson equation with the gradual channel approximation (equation 9) is solved for ϕ_s. Defining strong inversion as the area where $\phi_s = 2\phi_b$, the inversion region is confined to $\phi_b < \phi_s < 2\phi_b$. In inversion, the total charge due to the channel potential as a function of location can be simplified to [10]

$$Q_s(y) = -\sqrt{2\varepsilon_s q N_b}\left(\phi_s(y) + U_T \exp\frac{\phi_s(y) - 2\phi_b - V_{ch}(y)}{U_T}\right)^{\frac{1}{2}} \tag{57}$$

In this case N_b is the acceptor concentration. Substituting this Q_s into equation 16 we get

$$V_g = V_{fb} + \phi_s(y) + \frac{\sqrt{2\varepsilon_s q N_b}}{C_{ox}} \left(\phi_s(y) + U_T \exp \frac{\phi_s(y) - 2\phi_b - V_{ch}(y)}{U_T} \right)^{\frac{1}{2}} \tag{58}$$

Under the condition $V_{DS} = 0$, $V_{ch} = V_{SB}$ (this follows from the boundary conditions in equation 56). In strong inversion $\phi_s = 2\phi_b$, consequently the surface potential is a constant along the channel;

$$\phi_s(y) = \phi_s = 2\phi_b + V_{SB}. \tag{59}$$

Subsequently, when $V_{DS} = 0$ equation 57 can be simplified to

$$\begin{aligned} Q_s(y) &= -\sqrt{2\varepsilon_s q N_b} \left((2\phi_b + V_{SB}) + U_T \exp \frac{(2\phi_b + V_{SB})}{U_T} \right)^{\frac{1}{2}} \\ &= -\sqrt{2\varepsilon_s q N_b \, (2\phi_b + V_{SB})}. \end{aligned} \tag{60}$$

Substituting Q_s from equation 60 into into equation 58, we get an expression for the threshold voltage.

$$V_T = V_g = V_{fb} + 2\phi_b + \frac{\sqrt{2\varepsilon_s q N_b}}{C_{ox}} \sqrt{2\phi_b + V_{SB}}. \tag{61}$$

Defining the body factor γ,

$$\gamma = \frac{\sqrt{2\varepsilon_s q N_a}}{C_{ox}} \tag{62}$$

the threshold voltage is

$$V_T = V_{fb} + 2\phi_b + \gamma\sqrt{2\phi_b + V_{SB}}. \tag{63}$$

3.1.2 Short Channel Effect

For short-channeled devices, the threshold voltage is lowered as a consequence of SCE. This must be incorporated in the threshold voltage model. The SCE impact on the threshold voltage can be shown to be [44]

$$\Delta V_T(\text{SCE}) = \frac{2(V_{bi} - \phi_s) + V_{SD}}{2\cosh\left(\frac{L_{\text{eff}}}{2l_t}\right) - 2} \tag{64}$$

where l_t is a characteristic length and V_{bi} is the built-in potential given as

$$V_{bi} = U_T \ln\left(\frac{N_a N_d}{n_i^2}\right). \tag{65}$$

l_t is defined as[44]

$$l_t = \sqrt{\frac{\varepsilon_s t_{ox} W_a}{\varepsilon_{ox}}}. \tag{66}$$

W_a is the average width of the depletion region along the channel and t_{ox} is the oxide thickness.

Further corrections to V_T include adjusting for non-uniformly doped samples, and introducing new parameters to increase model flexibility[10].

3.1.3 Strong Inversion Drain Current Modeling

In strong inversion, the drain current is modeled by[41]

$$I_d = WC_{ox}\left(V_g - V_T - A_{\text{bulk}}V(y)\right)v \tag{67}$$

A_{bulk} is a parameter designed to account for the bulk charge effect. A_{bulk} is very close to unity if the channel is small and increases with channel length[41]. $V(y)$ is the potential difference between the minority-carrier imref and the equilibrium Fermi potential in the bulk. Writing equation 67 in terms of E by substituting v from equation 6 we get

$$\frac{dV(y)}{dy} = E(y) = \frac{I_d}{\mu_s WC_{ox}(V_g - V_T - A_{\text{bulk}}V(y)) - \frac{I_d}{E_c}}. \tag{68}$$

Integrating this gives the drain current model for velocities below v_d:

$$I_d = \frac{\mu_s C_{ox} W V_{DS}}{2L}\frac{V_g - V_T - A_{\text{bulk}}}{1 + \frac{V_{DS}}{E_c L}}. \tag{69}$$

In the saturation regime I_d is only weakly dependent on V_{DS}. Here, expanding the current to a Taylor series is one way of analyzing the current[41].

3.1.4 Subthreshold Drain Current Modeling

In the subthreshold region the drain current can be expressed as[10]

$$I_d = \frac{\mu_n W v_t^2}{L} \sqrt{\frac{q\varepsilon_s N_b}{2\phi_s}} \left(1 - \exp\left(-\frac{V_{DS}}{v_t}\right)\right) \exp\left(\frac{SS(V_g - V_T - V_{\text{off}})}{v_t}\right) \tag{70}$$

where v_t is the mean thermal velocity and V_{off} is the offset voltage. To model the subthreshold slope, the channel length and the interface state density should be taken into account. In addition the coupling capacitances between the channel and drain and channel and source are parameters that must be extracted from the transistor.

3.2 Surface Potential Models

Surface potential based models use the Pao-Sah integral (section 2.4) to calculate the channel surface potential. Other output variables like terminal charges, currents and their derivatives are calculated from the surface potential. Examples of surface potential based models include PSP[45], HiSIM[46], BSIM-CMG[47] and MOS Model 11[48]. The equations in this chapter are predominantly adapted from PSP.

3.2.1 Surface Potential Equation

To find the surface potential equation, we start with Poisson's equation with the gradual channel approximation (equation 9) and neglect the hole current. The hole quasi Fermi level (imref) F_p is similarly negligible. The Boltzmann relation gives the hole concentration as a function of ϕ_s

$$p_b = N_a \exp\left(\frac{\phi_s}{U_T}\right) \tag{71}$$

where p_b is the majority carrier concentration in the neutral bulk region. For the hole concentration, the quasi Fermi potential can not be ignore thus the Boltzmann relation becomes

$$n = n_p \exp\left(\frac{\phi_s - F_p + F_n}{qU_T}\right). \tag{72}$$

Assuming the channel dopants to be completely ionized, $N_a = p_b - n_p$ and the charge density becomes

$$\rho = q\left(p_b\left(\exp\left(-\frac{\phi}{U_T}\right) - 1\right) - n_p\left(\exp\left(-\frac{F_p - F_n}{qU_T}\right)\exp\left(\frac{\phi}{U_T}\right) - 1\right)\right). \tag{73}$$

Setting the boundary condition

$$\frac{d\phi_s(0)}{dx} = 0 \tag{74}$$

it follows from Poisson's equation (equation 9) that

$$E_s^2 = -\frac{2}{\varepsilon_s}\int_0^{\phi_s} \rho d\phi. \tag{75}$$

Re-writing equation 16 with γ, defined in equation 62, we can obtain the following relation:

$$(V_g - V_{fb} - \phi_s)^2 = \gamma^2 U_T h. \tag{76}$$

Here the normalized square of the surface electric field h is defined as

$$h = \frac{\varepsilon_s E_s^2}{2qU_T p_b} = -\frac{1}{qU_T p_b}\int_0^{\phi_s} \rho d\phi. \tag{77}$$

The integral in equation 77 with ρ as in equation 73 cannot be solved unless $F_n = F_p$, which is not the case in most semiconductors. Thus further approximations must be done.

By defining a normalized imref splitting constant

$$k = \exp\left(\frac{F_p - F_n}{qU_T}\right) \tag{78}$$

different procedures of solving equation 77 can be tried out, e.g. disregarding the electron position dependence on the imref[34].

3.2.2 Charge-Sheet Approximation

The charge-sheet approximation assumes the inversion layer in the channel is infinitesimally thin, i.e. the potential does not vary through the thickness of the channel. This yields an implicit expression for the surface potential in terms of applied voltage.

In an n-channel MOSFET, the charge-sheet approximation relate the gate and channel potentials in the following way[49]

$$V_g - V_{fb} = \phi_S + \gamma + \sqrt{\phi_S} - \frac{Q_i}{C_{ox}} \tag{79}$$

$$Q_i = -\gamma C_{ox}\sqrt{U_T}\left(\sqrt{\frac{\phi_S}{U_T} + \exp\left(\frac{\phi_s - 2\phi_f - V_{ch}}{U_T}\right)} - \sqrt{\frac{\phi_s}{U_T}}\right). \tag{80}$$

3.2.3 Symmetric Linearization for Drain Current Modeling

Symmetric linearization is a systematic approach to simplify expressions in a surface potential-based model. Because the charge-sheet approximation is made, all charges in this section is per unit area.

Defining ϕ_m as the surface potential at the midpoint of the channel. An estimation of the bulk charge per unit channel area Q_b is then [50]

$$Q_b(\phi) \approx Q_b(\phi_m) + \left.\frac{dQ_b}{d\phi}\right|_{\phi=\phi_m}(\phi - \phi_m). \tag{81}$$

This gives the inversion charge per unit area as

$$Q_i(\phi) = C_{ox}(V_m + \alpha_m(\phi_m - \phi)) \tag{82}$$

where α_m is a midpoint surface potential constant

$$\alpha_m = 1 + \frac{\gamma}{2}(\phi_m - U_T)^{-1/2} \tag{83}$$

and V_m is the voltage at the channel midpoint;

$$V_m = \frac{Q_i(\phi_m)}{C_{ox}} = \frac{\gamma U_T \Delta(\phi_m, F_m)}{\sqrt{\phi_m - U_T + U_T\Delta(\phi_m, F_m)} + \sqrt{\phi_m - U_T}}. \tag{84}$$

F_m is the channel midpoint imref and

$$\Delta(\phi, F) = \exp\left(\frac{\phi - 2\phi_b - F_n}{U_T}\right) \tag{85}$$

can be the subject of further linearization:

$$\Delta(\phi_m, F_m) \approx \frac{1}{2}(\Delta(\phi_S, F_S) + \Delta(\phi_D, F_D)) - \frac{(\phi_D - \phi_S)^2}{4\gamma^2 U_T} \tag{86}$$

Here ϕ_S and ϕ_D are the surface potentials at the source and drain and F_S and F_D are the imrefs at the source and drain. Drain current in this model is then equally defined for for all values of ϕ[50]:

$$I_d = \mu C_{ox} \frac{W}{L}(V_m + \alpha_m U_T)\phi. \tag{87}$$

3.3 Inversion Charge-Based Models

In charge-based models, the inversion charge at the two ends of the channel is used to calculate the surface potential in the device. Some models that use this approach are AMC[51], EKV[52] and BSIM6[53]. Equations in this chapter are adapted from EKV and BSIM6.

3.3.1 Symmetric Linearization of Charge Density

With the Poisson equation and the gradual channel approximation (equation 9) as a starting point, the charge sheet approximation relates the gate and channel potentials to the inversion charge density as shown in equation 79. The inversion charge density dependence on the surface potential is almost linear when the gate voltage is kept constant. Defining a pinch-off surface potential ϕ_P as the surface potential when the inversion charge is zero[54], we get;

$$\phi_P = V_g - V_{fb} - \gamma^2 \left(\sqrt{\frac{V_g - V_{fb}}{\gamma^2} + \frac{1}{4}} - \frac{1}{2} \right) \tag{88}$$

the linearized inversion charge can be approximated symmetrically by using the inversion charge secant in the two data points $(0, \phi_P)$ and (Q_{i0}, ϕ_{s0}). For an arbitrary value of ϕ_{s0} that has yet to be defined, and its corresponding Q_{i0} calculated from equation 79, the linearized inversion charge is approximated as[54]

$$\frac{Q_i}{C_{ox}} = n_q \left(\phi_s - \phi_P \right). \tag{89}$$

Here an inversion linearization factor n_q is defined as

$$n_q = 1 + \frac{\gamma}{\sqrt{\phi_{s0}} + \sqrt{\phi_P}}. \tag{90}$$

ϕ_{s0} can be a function of the channel potential, or it can be set to a constant value $2\phi_f$ [54]. This gives the simplified inversion linearization factor

$$n_q = 1 + \frac{\gamma}{\sqrt{2\phi_f} + \sqrt{\phi_P}}. \tag{91}$$

Re-writing equation 80 and 89 with 91 a relation for V_{ch}, Q_i and ϕ_P is found:

$$\ln\left(-\frac{Q_i}{\gamma C_{ox}\sqrt{U_T}} \left(-\frac{Q_i}{\gamma C_{ox}\sqrt{U_T}} + 2\sqrt{\frac{Q_i}{n_q C_{ox} U_T} + \frac{\phi_P}{U_T}} \right) \right) - \frac{Q_i}{n_q C_{ox} U_T} = \frac{\phi_P - 2\phi_f}{U_T} - \frac{V_{ch}}{U_T}. \tag{92}$$

This relation for inversion charge density and surface potential is valid for both weak and strong inversion regimes[54].

3.3.2 Drain Current Modeling

The drain current is divided into drift current and diffusion current, which can be written as[53]

$$I_{\text{drift}} = -\mu W_{\text{eff}} Q_i \frac{d\phi_s}{dx} \tag{93}$$

$$I_{\text{diff}} = \mu W_{\text{eff}} U_T \frac{dQ_i}{dx} \tag{94}$$

Thus the total drain current becomes:

$$I_d = -\mu W_{\text{eff}} Q_i \frac{d\phi_s}{dx} + \mu W_{\text{eff}} U_T \frac{dQ_i}{dx}. \tag{95}$$

Substituting ϕ_s from equation 89,

$$I_d = \mu W_{\text{eff}} \left(-\frac{Q_i}{n_q C_{ox}} + U_T \right) \frac{dQ_i}{dx}. \tag{96}$$

Introducing a normalized inversion charge Q_n

$$Q_n = \frac{Q_i}{-2 n_q C_{ox} U_T} \tag{97}$$

and switching the spatial variable to ξ

$$\xi = \frac{x}{L} \tag{98}$$

the drain current can be written as

$$\begin{aligned} I_d &= \mu \frac{W_{\text{eff}}}{L_{\text{eff}}} \left(-\frac{-2 n_q C_{ox} U_T Q_n}{n_q C_{ox}} + U_T \right) \frac{d(-2 n_q C_{ox} U_T Q_n)}{d\xi} \\ &= -2 n_q C_{ox} k_{SS} U_T^2 (2Q_n + 1) \mu \frac{W_{\text{eff}}}{L_{\text{eff}}} \frac{dQ_n}{d\xi}. \end{aligned} \tag{99}$$

Here k_{SS} is a subthreshold slope degradation factor. Integrating from 0 to 1 (since quantities have been normalized) we get the total drain current

$$I_{d,\text{tot}} = \int_0^1 I_d d\xi = 2 n_q \mu \frac{W_{\text{eff}}}{L_{\text{eff}}} C_{ox} n U_T^2 ((Q_{nS} - Q_{nD})(Q_{nS} + Q_{nD} + 1)) \tag{100}$$

Here Q_{nS} and Q_{nD} are the normalized charges in the source and drain respectively.

For high levels of applied voltage, the charge carrier velocity in the channel becomes saturated. This reduces the drain current compared to equation 100. The effect of velocity saturation on carrier mobility can be expressed as [53]:

$$\mu_s = \frac{\mu}{\sqrt{1 + \left(\frac{1}{E_c} \frac{d\phi_s}{dx} \right)^2}}. \tag{101}$$

Substituting this μ_s into equation 99, we get

$$I_d = -2n_q C_{ox} n U_T^2 (2Q_n + 1) \frac{\mu}{\sqrt{1 + \left(\frac{1}{E_c} \frac{d\phi_s}{dx}\right)^2}} \frac{W_{\text{eff}}}{L_{\text{eff}}} \frac{dQ_n}{d\xi}. \tag{102}$$

Integrating gives the total current[53]

$$I_{d,\text{tot}} = \int_0^1 I_d d\xi = 2n_s \mu \frac{W_{\text{eff}}}{L_{\text{eff}}} C_{ox} n U_T^2 ((Q_{nS} - Q_{nD})(Q_{nS} Q_{nD} + 1)). \tag{103}$$

In addition to this velocity saturation adjustment to I_d, the device output conductance, that takes drain-induced barrier lowering, channel length modulation and short-channel effects into account, should be considered[53].

3.4 Measurement-Based Modeling

Modeling from data is without doubt the most accurate way of building a transistor model[55]. This is called an empirical model.

Measured data from devices, stored as tables of I-V relations, Q-V-relations etcetera, contain a discrete set of values for the given device. The simulator uses these values to interpolate and calculate the desired output variables.

Making physical measurements on devices is fundamental for empirical models. However this is not trivial in most modern IC contexts, and not compatible with modeling while the device design is not finished. Thus optimization using an iterative design process is not possible when using empirical models.

3.5 TCAD

Technology computer aided design (TCAD) is a method used for process and device design[56]. When talking about device modeling, TCAD can be used to model the MOSFET fabrication process, thus yielding a fairly accurate set of measurements for a given device. TCAD is particularly useful for failure analysis, and can be used to extract device statistics that can form a basis for quantitative compact model tests[57].

A typical TCAD simulation flow consists of starting with the process recipe. Every process step is simulated using precise parameters for the prospective layout. The resulting structure can then be implemented with a device model to produce device characteristics such as IV-curves. This approach gives less uniform models that can give insight to varied parameters such as oxide thickness inconsistencies.

A large impediment to widespread TCAD usage among compact model and IC designers is the amount of knowledge of process parameters required to accurately built the TCAD simulation. As the device fabrication is often handled by an external foundry, process parameters may not be easily accessible to model designers.

Another aspect of TCAD is that it requires a lot of computational power. IC manufacturing normally includes tens of steps in highly specialized machines, many of which have little or no means of observing the process step as it happens. Simulating all of this is an extensive undertaking and can not be a part of what would be considered compact modeling. Nonetheless TCAD can be used to generate device operation data that in turn can be used to quantitatively analyze a compact model. TCAD can thus be an alternative to doing actual measurements on the device for model validation purposes.

4 Compact Model Validation

This chapter is meant as an instructional guide to compact model validation. Relevant tests are presented along with example curves that highlight pass and fail conditions.

4.1 V_g-sweep

To inspect threshold voltage characteristics, a V_g-sweep should be conducted. Plotting I_d and g_m, the curves should have no kinks, glitches or discontinuities. The areas around the threshold and flatband voltages are where most models fail, hence these areas should be the main focus of scrutiny. Examples of acceptable I_d and g_m curves are shown in figure 7. The test should be conducted with increasing V_{DS} for varying L and W. Different values for V_b should also be examined.

As the transconductance-to-current ratio $\frac{g_m}{I_d}$ is an important quantity for analog design[17], it should be plotted to ensure no abrupt spikes in the moderate inversion region. The transconductance-to-output conductance ratio $\frac{g_m}{g_o}$ should also be checked. Examples are shown in figure 8. The curves shown are smooth and have no peak in weak inversion, so the model passes this test. (Note that in newer models such as BSIM4, this issue has largely been corrected and $\frac{g_m}{I_d}$ behavior in moderate inversion is satisfactory[42].)

4.2 V_{DS}-sweep

Modeling transistor behavior when V_{DS} approaches zero is notoriously difficult[17], and I_d and g_o should be investigated for non-differentiable points (curve kinks) by plotting these variables and their derivatives while sweeping V_{DS}. The section where V_{DS} approaches zero and the transition from triode to saturation are areas where models tend to fail and are thus the main areas of interest. Additionally, g_o should at no point be negative. Examples of acceptable curves are shown in figure 9, where the plots are smooth also when V_{DS} approach zero. Relevant parameters are V_g, V_b and L.

4.2.1 Transcapacitances

Transcapacitance behavior around $V_{DS} = 0$ should be plotted for small steps of V_{DS}. Expected behavior for transcapacitances include reciprocity in $V_{DS} = 0$, that is $C_{ij} = C_{ji}$[19], including requiring $C_{gs} = C_{gd} = C_{dg}$ and $C_{bs} = C_{bd} = C_{db}$[34]. In figure 10, C_{gs}, C_{dg} and C_{gd} are plotted. Around zero C_{gs} and C_{gd} have discontinuities, so the model can not be said to behave physically in this case.

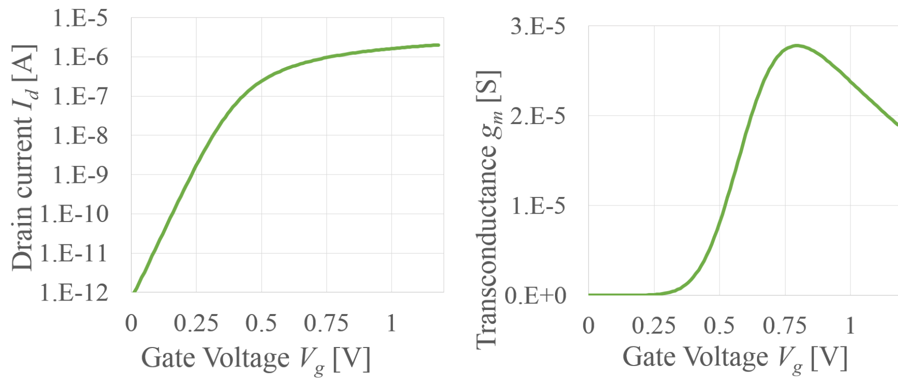

Figure 7: Threshold voltage characteristics. The curves have no discontinuities or kinks and are thus acceptable.

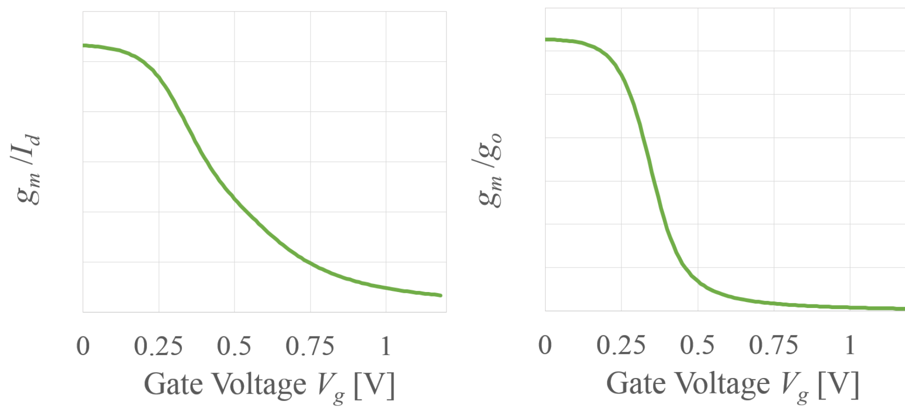

Figure 8: $\frac{g_m}{I_d}$ and $\frac{g_m}{g_o}$ plotted against V_g. The curves have no abrupt changes or spikes in the moderate inversion area and have thus passed the test.

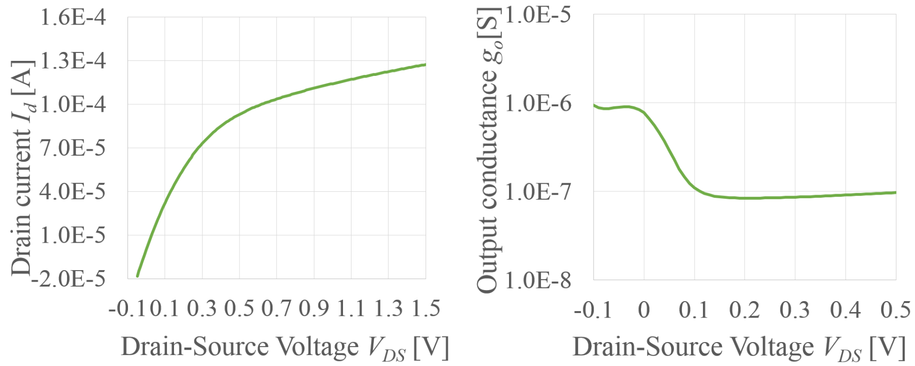

Figure 9: Curves for V_{DS}-sweep characteristics. As the curves look smooth around $V_{DS} = 0$, the test is passed.

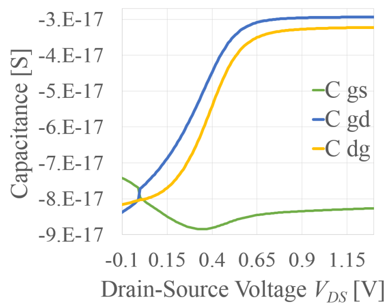

Figure 10: Some transcapacitances plotted with V_{DS}. All three curves should meet in $V_{DS} = 0$. In this model the trancapacitances are not continuous around zero, hence the modeled behavior of the transcapacitances is not physical for these data points.

4.3 Gummel Symmetry

To investigate asymmetry for the source and drain in devices manufactured to be symmetric, the device can be tested with $V_d = V_x$ and $V_s = -V_x$. This is called a Gummel symmetry test. Gummel tests are intended for uncovering asymmetries in the device model, but can also highlight other problematic behavior. I_d and derivatives of I_d are plotted around $V_x = 0$ for values of V_g. Such plots are shown in figure 11. Even though the plots are all symmetrical around zero, the derivatives plotted in figure 11b are not differentiable in $V_x = 0$, thus this test reveals a model weakness.

4.4 Length and Width Characteristics

Length and width should be swept, and the I_d, g_m and g_o should be plotted along with their first derivatives for a set of temperatures. Example curves for drain current are given in figure 12, and for conductances in figure 13. Both curves are without discontinuities and have thus passed the qualitative test.

Because various equations are used to modulate different effects in the model, some model "stitching" can be discovered in length and width sweeps. One example of this is modulating for short channel effects. This is only done below certain values and can thus introduce abrupt changes in the output variables. Sweeping length and width parameters reveal where stitches or joints are situated. In figure 14 this is shown. The drain current looks smooth but plotting its derivative exposes the model stitches at $W = 300$ and 600 nm.

4.5 Leakage Current Characteristics

The temperature should be swept while the gate voltage is kept at zero and V_{DS} and L are varied. I_d should be split into I_{DS} and I_{DB}, and these two graphs should increase with temperature. I_{sat} and V_{bi} should also be be verified; these two parameters should diminish when T increases.

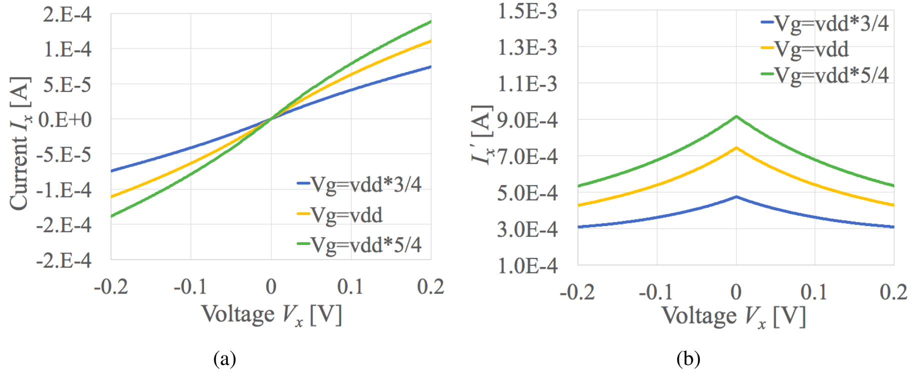

Figure 11: Gummel symmetry plots for varying V_g. (a) I_x. (b) $\frac{dI_x}{dV_x}$. The derivative is not differentiable in $V_x = 0$, so the model behavior is not ideal. However, the plot is symmetric around zero.

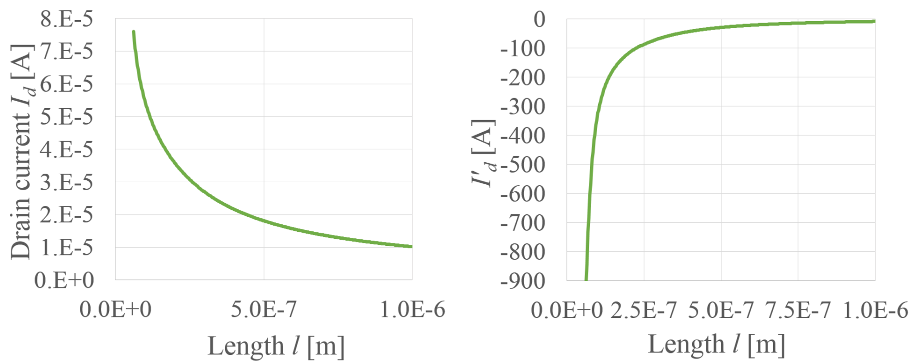

Figure 12: I_d plotted with L. The curves look smooth, so the model is not problematic in this area.

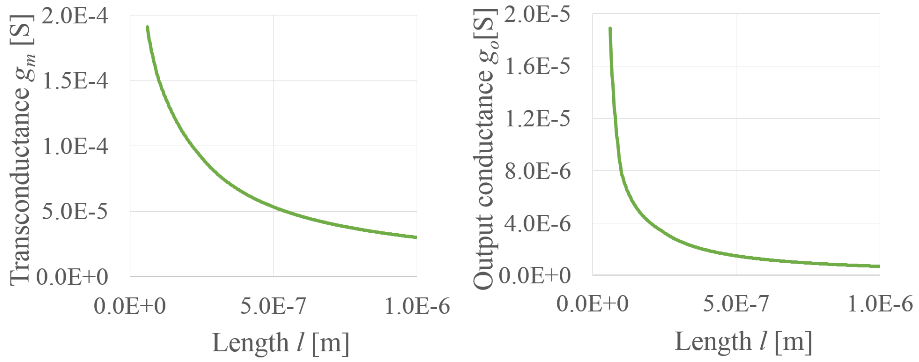

Figure 13: Conductance characteristics for length-sweep. The transconductance is differentiable in all points, hence the model is in accordance with the laws of physics.

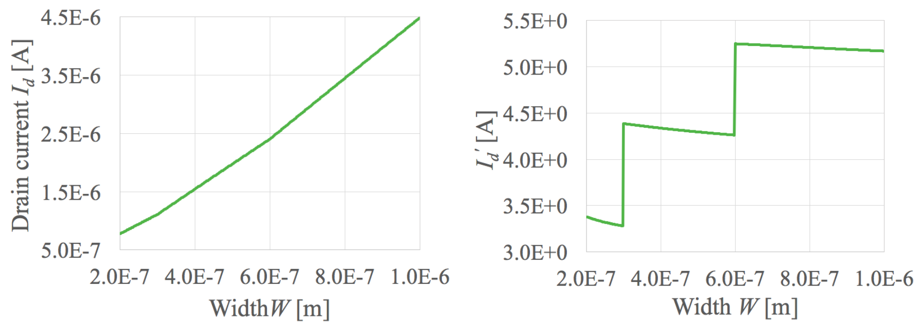

Figure 14: I_d plotted with W. In this model, stitching happens at 300 nm and 600 nm.

4.6 Charge Conservation

The combination of numerical integration methods used by the device model can lead to non-physical charge behavior, i.e. charge conservation issues[58]. However, as compact models started using both charge equations and capacitance equations, along with imposing self-consistent boundary conditions on the charges, this problem has been ameliorated and is no longer an important concern[42]. (This calculation is possible when using terminal charges as state variables instead of terminal voltages.)

Charge conservation can be verified by plotting V_g, V_b and V_d against time. If the charge is conserved, these three plots should look identical, in particular they should have no phase difference[58].

4.7 Non-Quasi-Static Operation

Non-quasi-static (NQS) operation is important to model when utilizing RF circuits, especially when using long-channel devices. For low frequencies in a long-channel device kept at $V_{DS} = 0$ the transresistances can be defined as[34]

$$R_{DG} = \frac{1 + \frac{C_s}{C_{ox}}}{6g_o} \tag{104}$$

$$R_{GB} = \frac{1}{12g_o}. \tag{105}$$

Both parameters should yield smooth curves when plotted against V_g. As V_g approaches zero the resistances should not go to infinity.

4.8 Noise

When doing noise simulation tests, V_{DS} should be kept constant and V_g at V_{dd}. Thermal and flicker noise is separated by frequency; at low frequencies the flicker noise dominates. Thermal noise is not frequency dependent and can be investigated for higher frequencies, e.g. above 100 MHz[32].

Flicker noise should be analyzed for low frequencies with varied W, and the noise power spectral density should decrease with increasing device width. Altering the value of V_g should not affect the noise signal.

5 Deep Learning

Deep learning, also known as machine learning or neural networks, is an approach to artificial intelligence where multiple layers of nodes are implemented to recognize patterns in data[20]. One example is classifying images by what they contain.

Deep learning algorithms use layers of image representation to introduce levels of abstraction to the pattern-recognition process. Starting with the input data, each layer accentuates what parts of the image are significant for classification, and which variations are less relevant. The weighting is not done by a programmer, but by the program itself. This makes deep learning algorithms versatile, as similar algorithms can be applied to different forms of input data and to accomplish different goals.

Deep learning algorithms typically consists of three stages; training, testing and validation[59]. After the training phase, a validation set of examples are often used to fine-tune the model parameters. Validation can be used to find the best number of hidden nodes or the ideal value for the learning rate. Ultimately the model is tested and the model properties are assessed.

Learning in algorithms is divided into supervised and unsupervised learning. For supervised learning, the input data used in the training phase contains labels assigning each image to a predefined class. Unsupervised learning, on the other hand, takes images with no labels as input data and establishes connections directly from them[60].

One way of implementing a supervised deep learning algorithm is computing a function that measures a score in assigning images to categories. The algorithm's objective is to maximize this score by adjusting its internal parameters. To do this, a gradient vector is computed that finds the resulting difference in model score, resulting from making a small increase or decrease in every possible parameter. The weights and biases are consequently regulated so that the output error is low on average. One iteration is called an epoch. The nodes and weights constitute a network of Markov Chains, and are illustrated in figure 15.

To simplify computation, the training dataset can be divided into smaller batches (minibatches) and the adjustments to weights and balances can be calculated limited within each minibatch. An average of required adjustments is calculated and updated every time a new minibatch has been processed. This is repeated until the calculated average stabilizes[20]. Since weights are calculated for every existing pair of nodes in the training set, splitting it into minibatches decreases the number of operations carried out by the program substantially.

For a large number of nodes, e.g. a large set of images, each containing a thousands of pixels, the memory required for deep learning grows exponentially. Calculating and storing gradients, nodes and layers of a deep learning algorithm can require extensive computational power. Deep learning algorithms are often designed to run on multiple graphics processing units (GPUs)[31].

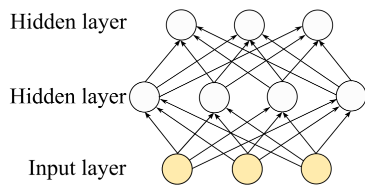

Figure 15: Nodes and layers in a neural network.

5.1 MNIST

A very central example in deep learning is the Mixed National Institute of Standards and Technology (MNIST) database[61]. This set contains pictures of handwritten numbers, standardized in terms of format so that all images have the same size (28x28 pixels). All pixels are either white or black (binary images) and each image is labeled according to the number depicted (0 to 9). The digits are size-normalized and centered in the image. The MNIST database today contains 70 000 images[62], conventionally split into training sets and test sets.

5.2 Restricted Boltzmann Machine

A restricted Boltzmann machine (RBM) is a form of neural network where binary pixels are connected to binary feature detectors by symmetrically weighted connections[62]. Here the pixels constitute the visible nodes ($\mathbf{v}$), while the feature detectors are hidden nodes ($\mathbf{h}$), in sum two layers. Designating $\mathbf{b}$ as biases and $\mathbf{w}$ as weights, the RBM energy is

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in \text{pixels}} b_i v_i - \sum_{j \in \text{features}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}. \tag{106}$$

Here v_i and h_j are the binary states of pixel i and feature j. This function defines a probability to every possible image. Given a training image, all values of $\mathbf{h}$ are set to 0 or 1 with a probability of $\sigma(b_j + \sum_i v_i w_{ij})$, where σ is the logistic function:

$$\sigma(x) = \frac{1}{1 + \exp(x)}. \tag{107}$$

With this starting point a distorted image is produced, often called a confabulation[62]. In this image, the probability of pixel v_i having the value 1 is $\sigma(b_i + \sum_j h_j w_{ij})$. Next the weight vector is updated with the change δ_{ij} in each variable w_{ij}, where

$$\delta w_{ij} = l_r \left(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{conf}} \right). \tag{108}$$

l_r is called the learning rate and $\langle v_i h_j \rangle$ represents the fraction of times where pixel i and feature detector j are on together for the data and confabulations. Similarly the biases are updated. Thus the machine learns by using error-efficient back propagation, without following the the exact gradient of the log probability of the training data.

6 Results and Discussion

6.1 Benchmarking Compact Model Validation

A set of benchmark tests was developed and is presented in section 4. In addition to Gummel symmetry, charge conservation, non-quasi-static operation and noise tests, V_g, V_{DS}, L, W and T should be swept with parameters such as V_{DS}, V_g, V_b, L, W, and T.

By structuring tests after sweep variable, the number of different tests in the validation process is lowered. In this report a set of nine qualitative tests are proposed, compared to 12[17] or 22[15] different tests, sorted mainly by output signal. Automating test runs would presumably be less time-consuming when the test set contains fewer tests, however the number of curves to evaluate would not change.

When testing a compact model the intended application should always be kept in mind. One example is if the user plans to optimize the transistor's L and W for a specific parameter. In this case the model files must not contain stitching in the relevant interval. At the very least the model benchmarking process must expose and convey the specifics of this mechanism.

When ultimately deciding whether a compact model has passed or failed the model validation, a large number of plots and specifications will have to be considered. Even if a model fails under certain conditions such as very high temperatures or very long channel, it can still probably be used for typical modeling. Again, knowing the model's weaknesses becomes critical.

6.2 Quantitative Checking of Compact Models

For quantitative checking of models, simulation data should be compared to physical measurements. Operation parameters such as temperature and voltages should be identical for the two sets of data in order to make direct comparisons. This data could either be measured in a lab, or obtained directly from the model foundry. For this thesis it was not possible to do direct measurements on the transistor.

Compact device models are typically released alongside select measurements. However, these measurements often represent only select characteristics of the device. For this thesis, the measurements provided by the model manufacturer were not sufficient to do any meaningful quantitative comparison of the device to the compact model.

One could imagine a quantitative comparison of the device model with TCAD-generated data. This would require even more comprehensive information about the transistors; i.e. the full process recipe.

6.3 Test Assessment Automation

According to section 4, in order to be validated a compact model should go through nine different tests, all with a number of varying parameters and corners. Introducing any form of automation would simplify this model validation process. In the following, two different approaches to assessing each individual curve resulting from the benchmarking process, are proposed.

6.3.1 Deep Learning Algorithm

An attempt was made to write a deep learning program that could separate a passed test from a failed test. This was to be done by image classification.

As the compact model simulation produces a csv-file with x- and y-coordinates, the data had to be transformed into a binary image before being evaluated. Each image contained 160000 pixels (400x400), which in retrospect is far too many to be analyzed on a personal computer in a reasonable amount of time. This is especially true for deep learning programs that require a number of hidden nodes equal to the number of visible nodes (pixels). Csv-files were read and converted into binary vectors containing one variable for each pixel in the image. The images were divided into two categories; pass and fail. As a first approach the training database contained 20 images, 5 passed and 15 failed, all generated by running a Gummel symmetry test (see section 4.3) and extracting $\frac{dI_x}{dx}$. A binary image of a failed test run is shown in figure 16.

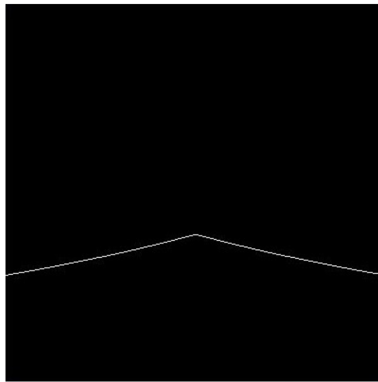

Figure 16: Example of binary images used in RBM. Because of the sharp kink, this test was deemed failed.

Using Theano[31], a restricted Boltzmann machine was implemented based on an RBM for MNIST classification. The training was done in minibatches, but ultimately the algorithm was not able to separate passed and failed images.

One reason the RBM was unsuccessful might be that the objects to be detected, i.e. the curves, were not centered on the y-axis of the images. Spacial variance is probably something that should be handled by a more advanced neural network than an RBM. Alternatively, both centering and normalization of the plots could be handled by the data loading function, thereby simplifying the learning process at the cost of a more complex data loading function.

6.3.2 Python Script for Test Differentiation

To detect failed tests, a simple algorithm to find discontinuities in curves was implemented. By comparing neighboring points, curve discontinuities above a given threshold value are detected. The source code for this program is given in Appendix B.

$\frac{dI_d}{dV_{DS}}$-curves were exported from a V_{DS}-sweep (see section 4.2). Since this test normally fails around $V_{DS} = 0$, the data was zoomed in on this area. To discover irregularities in these plots, the gradient was analyzed for discontinuities. Examples of data used are shown in figure 17. As seen in figure 17b, the discontinuity in $V_{DS} = 0$ is divided into three by two intermediate points. This means that a failed curve has three differences between neighboring data points that are greater than the threshold value.

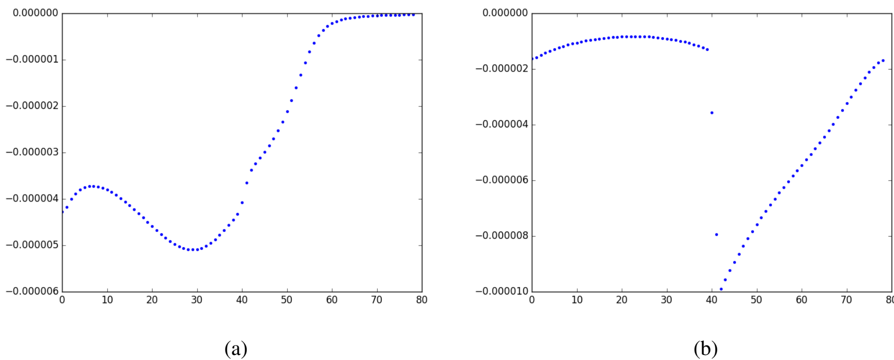

Figure 17: $\frac{d^2I_d}{dV_{DS}^2}$-plots. Number on the x-axis indicate index, and $V_{DS} = 0$ in $x = 40$. (a) A passed curve with a slight disturbance around $V_{DS} = 0$. (b) A failed test with a large discontinuity in $V_{DS} = 0$.

The threshold value is calculated by the program. This is done by finding the average value of the difference between neighbors in the vector, and multiplying this value by 2.5. Since the threshold value is a multiple of the average difference value, it will be large for failed curves and small for passed curves. For passed curves there will thus be a large number of points where the difference between neighboring points exceeds the threshold value. For failed curves the threshold value will only be exceeded in the discontinuity.

6.3.3 Comparison of Deep Learning Algorithm with Conventional Algorithm

Even though Python modules for deep learning such as Theano exist, the fact remains that programming a neural network is a complex task[63]. In addition to being time-consuming to write, deep learning networks typically use more computational power than conventional functions. Working on multiple GPUs or connecting to a more powerful processor than a personal computer is not particularly difficult, but it adds another level of complexity to the implementation of the program.

Deep learning networks have the advantage of versatility; any form of image classification can be done by the same program. On the other hand, in order to be employed in a new application, the program needs a large set of pre-classified images to learn from. Building this database will be time-consuming, but with a good neural network the same kind of curves with the same kind of defects could probably be classified on the basis of the same training database. Normalizing the curves would further facilitate using the same deep learning program on curves from different model validation tests.

According to the presented arguments, programming conventional algorithms for evaluating curves or images would be the most economical approach for small sets of images, while deep learning is the preferred method for very large sets of images or when complicated features should be detected. Curves resulting from compact model validation are normally not particularly complex. The number of images to evaluate will depend on how the model validation is done, but following the approach in Section 4, an estimate of 450 images can be made. (Nine tests should be conducted with an average of two parameters and five generated plots each. If the number of investigated corners is five, the total number of images to evaluate is 450.) This number in itself is probably not high enough to justify a deep learning program for curve assessment. However, if compact models are validated on a regular basis building a curve database for deep learning may be a good investment.

In any case it is worth noting that automating curve evaluation will only determine whether each curve has passed or failed the test according to pre-set conditions. Unexpected unphysical behavior can not be discovered and thus represents a blind-spot. The argument could be made that a selection of crucial curves should be controlled manually after the curves have gone through the validation program.

7 Conclusion

A set of benchmark tests for compact model validation was developed. The set consists of nine qualitative tests, comprising tests sweeping V_g, V_{DS}, L, W, and Gummel symmetry, leakage current, charge conservation, non-quasi-static operation and noise tests.

To evaluate individual curves generated in the device model benchmark tests, two approaches were investigated. The first, programming a deep learning network, was deemed too complex and was thus forfeited. A simpler approach where discontinuities in the curve was discovered by comparing values of neighboring data points was implemented.

8 Bibliography

[1] R. H. Dennard, F. Geansslen, H. Yu, V. Rideout, E. Bassous, and A. LeBlanc, "Design of ion-implanted MOSFET's with very small physical dimensions," *IEEE Journal on Solid-State Circuits. (Special Issue on Micropower Electronics)*, vol. 9, pp. 256–268, October 1974.

[2] S. Salahuddin and S. Datta, "Use of negative capacitance to provide voltage amplification for low power nanoscale devices," *Nano Letters*, vol. 8, no. 2, pp. 405–410, 2008.

[3] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, pp. 114–117, April 1965.

[4] "The international technology roadmap for semiconductors home page." http://www.itrs2.net/, accessed April 2016.

[5] C.-T. Sah, "Evolution of the MOS-transistor - from conception to VLSI," *IEEE*, vol. 76, pp. 1280–1326, October 1988.

[6] C. C. McAndrew, "Practical modeling for circuit simulation," *IEEE Journal of Solid-State Circuits*, vol. 33, pp. 439–448, March 1998.

[7] L. W. Nagel and D. Pederson, "SPICE (simulation program with integrated circuit emphasis)," Tech. Rep. UCB/ERL M382, EECS Department, University of California, Berkeley, April 1973.

[8] H. K. J. Ihantola and J. L. Moll, "Design theory of a surface field-effect transistor," *Solid State Electronics*, vol. 7, pp. 423–430, 1964.

[9] A. Abidi, A. Rofougaran, G. Chang, J. Rael, J. Chang, M. Rofougaran, and P. Chang, "The future of CMOS wireless transceivers," in *Solid-State Circuits Conference, 1997. Digest of Technical Papers. 43rd ISSCC., 1997 IEEE International*, pp. 118–119, February 1997.

[10] S. K. Saha, *Compact Models for Integrated Circuit Design*. CRC Press, 2015.

[11] C.-T. Sah and H. C. Pao, "Effects of diffusion current on characteristics of metal-oxide (insulator)-semiconductor transistors," *Solid-State Electronics*, vol. 9, pp. 927–937, October 1966.

[12] J. R. Brews, "A charge-sheet model of the MOSFET," *Solid State Electronics*, vol. 21, pp. 345–355, February 1978.

[13] M. A. Maher and C. A. Mead, "A physical charge-controlled model for MOS transistors," *Advanced Research in VLSI: Proceedings of the 1987 Conference.*, pp. 211–229, 1987.

[14] Y. Tsividis and K. Suyama, "MOSFET modeling for analog circuit CAD: problems and prospects," *IEEE Journal of Solid-State Circuits*, vol. 29, pp. 210–216, March 1994.

[15] K. K. Das, S. G. Walker, and M. Bhushan, "An integrated CAD methodology for evaluating MOSFET and parasitic extraction models and variability," *Proceedings of the IEEE*, vol. 95, pp. 670–687, March 2007.

[16] B. Brooks, K. Green, J. Krick, T. Vrotsos, and D. Weiser, “Standardization and validation of compact models,” *Technical Proceedings of the 2002 International Conference on Modeling and Simulation of Microsystems*, vol. 1, no. 13, 2002.

[17] “IEEE Recommended Practices Number P1485 on: Test Procedures for Micro-Electronic MOSFET circuit Simulator Model Validation.” Micro Electronic MOSFET Task Group, IEEE Standard Working Draft., September 2006.

[18] A. Dutta, S. Sirohi, T. Ethirajan, H. Agarwal, Y. S. Chauhan, and R. Q. Williams, “BSIM6 – benchmarking the next-generation MOSFET model for RF applications,” in *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, pp. 421–426, January 2014.

[19] G. D. J. Smit, A. J. Scholten, and D. B. M. Klaassen, “RF benchmark tests for compact MOS models,” in *2010 IEEE Radio Frequency Integrated Circuits Symposium*, pp. 593–596, May 2010.

[20] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, 2015.

[21] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[22] G. Hinton, L. Deng, D. Yu, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, G. Dahl, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition,” *IEEE Signal Processing Magazine*, vol. 29, pp. 82–97, November 2012.

[23] S. Jean, K. Cho, R. Memisevic, and Y. Bengio, “On using very large target vocabulary for neural machine translation,” *CoRR*, 2014.

[24] D. O. Hebb, *The Organization of Behavior*. Wiley, 1949.

[25] D. H. Hubel and T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex,” *J Physiol*, vol. 160, pp. 106–154, January 1962.

[26] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[27] S. Dreyfus, “The computational solution of optimal control problems with time lag,” *IEEE Transactions on Automatic Control*, vol. 18, pp. 383–385, August 1973.

[28] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85 – 117, 2015.

[29] L. Roux, D. Racoceanu, N. Loménie, M. Kulikova, H. Irshad, J. Klossa, F. Capron, C. Genestie, G. Naour, and M. Gurcan, “Mitosis detection in breast cancer histological images; An ICPR 2012 contest,” *Journal of Pathology Informatics*, vol. 4, no. 1, p. 8, 2013.

[30] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015.

[31] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[32] C. C. Hu, *Modern Semiconductor Devices for Integrated Circuits*. Prentice Hall, 2009.

[33] W. Shockley, "A unipolar "field-effect" transistor," *Proceedings of the IRE*, vol. 40, pp. 1365–1376, November 1952.

[34] G. Gildenblat, *Compact Modeling: Principles, Techniques and Applications*. Springer, 2010.

[35] C. F. Gauss, *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. Sumtibus F. Perthes and I.H. Besser, 1809.

[36] K. Piskorski and H. M. Przewlocki, "The methods to determine flat-band voltage VFB in semiconductor of a MOS structure," in *MIPRO, 2010 Proceedings of the 33rd International Convention*, pp. 37–42, May 2010.

[37] I. Ferain, C. A. Colinge, and J.-P. Colinge, "Multigate transistors as the future of classical metal–oxide–semiconductor field-effect transistors," *Nature*, vol. 479, pp. 310–316, November 2011.

[38] R. P. Jindal, "Compact noise models for MOSFETs," *IEEE Transactions on Electron Devices*, vol. 53, pp. 2051–2061, September 2006.

[39] A. G. Jordan and N. A. Jordan, "Theory of noise in metal oxide semiconductor devices," *IEEE Transactions on Electron Devices*, vol. 12, pp. 148–156, March 1965.

[40] K. K. Hung, P. K. Ko, C. Hu, and Y. C. Cheng, "A unified model for the flicker noise in metal-oxide-semiconductor field-effect transistors," *IEEE Transactions on Electron Devices*, vol. 37, pp. 654–665, March 1990.

[41] X. Xi, W. Liu, X. Jin, M. Chan, C. Hu, J. Chen, M.-C. Jeng, Z. Liu, Y. Cheng, K. Chen, K. Hui, J. Huang, R. Tu, and P. Ko, *BSIM3v3.3 MOSFET Model User's Manual*. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, 2005.

[42] N. Paydavosi, T. H. Morshed, D. Lu, W. Yang, M. Dunga, X. Xi, J. He, W. Liu, X. Jin, J. Ou, M. Chan, Ali, Niknejad, and C. Hu, *BSIM4 v4.8.0 MOSFET Model - User's Manual*. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, 2013.

[43] NXP, *MOS Model, level 903*, 2011.

[44] Z. H. Liu, C. Hu, J. H. Huang, T. Y. Chan, M. C. Jeng, P. K. Ko, and Y. C. Cheng, "Threshold voltage model for deep-submicrometer MOSFETs," *IEEE Transactions on Electron Devices*, vol. 40, pp. 86–95, January 1993.

[45] X. Li, W. Wu, G. Gildenblat, G. Smit, A. Scholten, D. Klaassen, and R. van Langevelde, *PSP 102.3*. Arizona State University and NXP Semiconductors Research, 2008.

[46] H. J. Mattausch, M. Miura-Mattausch, N. Sadachika, M. Miyake, and D. Navarro, "The HiSIM compact model family for integrated devices containing a surface-potential MOSFET core," in *15th International Conference on Mixed Design of Integrated Circuits and Systems, 2008. MIXDES 2008.*, pp. 39–50, June 2008.

[47] S. Khandelwal, J. Duarte, A. S. Medury, V. Sriramkumar, N. Paydavosi, D. Lu, C.-H. Lin, M. Dunga, S. Yao, T. Morshed, A. Niknejad, S. Salahuddin, and C. Hu, *BSIM-CMG 110.0.0 Multi-Gate MOSFET Compact Model Technical Manual*. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, 2015.

[48] NXP, *MOS Model 11, level 1101*, 2011.

[49] Y. Tsividis, *Operation and modeling of the MOS transistor*. New York; Oxford University Press, 2nd ed ed., 1999.

[50] T. Chen and G. Gildenblat, "Symmetric bulk charge linearization in charge-sheet MOSFET model," *Electronics Letters*, vol. 37, pp. 791–793, July 2001.

[51] C. Galup-Montoro, M. C. Schneider, A. I. A. Cunha, F. R. de Sousa, H. Klimach, and O. F. Siebel, "The advanced compact MOSFET (ACM) model for circuit analysis and design," in *2007 IEEE Custom Integrated Circuits Conference*, pp. 519–526, September 2007.

[52] A. Bazigos, M. Bucher, F. Krummenacher, J. M. Sallese, A. S. Roy, and C. Enz, *EKV3 MOSFET Compact Model Documentation, Model Version 301.02*. Technical University of Crete, 2008.

[53] H. Agarwal, C. Gupta, S. Khandelwal, J. P. Duarte, Y. S. Chauhan, S. Salahuddin, and C. Hu, *BSIM6.1.1 MOSFET Compact Model Technical Manual*. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, 2015.

[54] J.-M. Sallese, M. Bucher, F. Krummenacher, and P. Fazan, "Inversion charge linearization in MOSFET modeling and rigorous derivation of the EKV compact model," *Solid-State Electronics*, vol. 47, no. 4, pp. 677–683, 2003.

[55] D. E. Root, "Future device modeling trends," *IEEE Microwave Magazine*, vol. 13, pp. 45–59, Nov 2012.

[56] S. Li and Y. Fu, *3D TCAD Simulation for Semiconductor Processes, Devices and Optoelectronics*. Springer New York, 2012.

[57] H. Gossner, C. Russ, F. Siegelin, J. Schneider, K. Schruefer, T. Schulz, C. Duvvury, C. R. Cleavelin, and W. Xiong, "Unique ESD failure mechanism in a MuGFET technology," in *2006 International Electron Devices Meeting*, pp. 1–4, December 2006.

[58] P. Yang, B. D. Epler, and P. K. Chatterjee, "An investigation of the charge conservation problem for MOSFET circuit simulation," *IEEE Journal of Solid-State Circuits*, vol. 18, pp. 128–138, February 1983.

[59] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer New York, 2009.

[60] G. Hinton, P. Dayan, B. Frey, and R. M. Neal, "The 'wake-sleep' algorithm for unsupervised neural networks," *Science*, vol. 268, pp. 1158–61, May 1995.

[61] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Back-propagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[62] G. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[63] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative study of caffe, neon, theano, and torch for deep learning," *CoRR*, 2015.

Appendix A List of Symbols

ϕ_s Surface Potential

ϕ_f Substrate Fermi potential

ϕ_i Substrate intrinsic potential

ϕ_b Bulk potential

ϕ_m Surface potential at channel midpoint

ϕ_S Surface potential at device source

ϕ_D Surface potential at device drain

ϕ_P Pinch-off surface potential

V_g Gate voltage

V_b Body voltage

V_T Threshold voltage

V_{DS} Drain-source voltage

V_{SB} Source-body voltage

$V_{D\text{sat}}$ Drain-source voltage at saturation

V_{dd} Positive supply voltage

V_{fb} Flatband voltage.

V_{ox} Oxide voltage

V_{ch} Channel voltage (electron quasi-Fermi potential)

V_{bi} Built-in potential

V_{off} Offset voltage

V_m Voltage at channel midpoint

$\overline{v_n^2}$ Mean square noise voltage

U_T Thermal potential

Q_i Inversion charge density

Q_s Space charge

Q_g Gate charge

Q_b Bulk charge per unit area

Q_g Gate induced charge

Q_n Normalized inversion charge

Q_{nS} Normalized charge in source

Q_{nD} Normalized charge in drain

I_d Drain current

I_{sat} Saturation current

I_{diff} Diffusion current

I_{drift} Drift current

$\overline{i_n^2}$ Mean square noise current

C_{gs} Gate-source transcapacitance

C_{sb} Source-body transcapacitance

C_{gb} Gate-body transcapacitance

C_{gd} Gate-drain transcapacitance

C_{db} Drain-body transcapacitance

C_{ox} Capacitance of oxide per unit area

C_s Capacitance of semiconductor per unit area

g_m Transconductance

g_o Output conductance

E_s Surface electric field

E_c Critical electric field

v_d Charge carrier drift velocity

v_t Thermal velocity

ρ Charge density

p_n Hole concentration in n-type semiconductor

p_b Majority carrier bulk hole concentration

n_i Intrinsic carrier concentration

n_p Minority carrier electron concentration

N_a Acceptor concentration

N_d Donor concentration

N_b Bulk dopant concentration

T Absolute temperature

J_n Electron current density

J_p Hole current density

F_n Electron imref

F_p Hole imref

F_m Imref at channel midpoint

F_S Imref at device source

F_D Imref at device drain

r_{rn} Electron recombination rate

r_{rp} Hole recombination rate

r_{gn} Electron generation rate

r_{gn} Hole recombination rate

y Spatial variable along the device channel

x Spatial variable from the gate down to the substrate

ξ Normalized spatial variable x

L Channel length

L_{eff} Effective channel length

l_t Characteristic length

W Channel width

W_{eff} Effective channel width

W_a Average depletion region width

t_{ox} Oxide thickness

μ_s Surface mobility

μ_n Electron mobility

μ_p Hole mobility

ε_s Permittivity of semiconductor

ε_{ox} Permittivity of oxide

SS Subthreshold slope

f Frequency

S_i Power spectral density of the current noise

S_v Power spectral density of the voltage noise

R Electric resistance

R_{DG} Drain-gate transresistance

R_{GB} Gate-body transresistance

q Elementary charge

k_B Boltzmann's constant

m Bulk charge factor

K_f Electron trap oxide density constant

A_f Carrier mobility Coulomb scattering dependency

γ Body factor

A_{bulk} Bulk charge effect parameter

h The normalized square of the surface electric field

k Normalized imref splitting constant

α_m Midpoint surface potential constant

n_q Inversion charge linearization factor

k_{SS} Subthreshold slope degradation factor

Appendix B Source Code

This is the python code described in section 6.3.2.

```
import matplotlib.pyplot as pyplot
import numpy
import os
from numpy import genfromtxt

#test_file returns 1 for a passed test and 0 for a failed test.
def test_file(filename):
    my_data = genfromtxt(filename, delimiter=',',usecols=numpy.arange(0,2))
    #my_data=my_data[90:169,1]          #If the file contains more data points
                                        #than necessary it can be cut here.

    my_data=numpy.gradient(my_data)
    differences=numpy.absolute(numpy.diff(my_data))

    #Set the threshold value to 2.5* the difference average.
    avrg=numpy.average(differences)
    threshold=avrg*2.5
    values_over_t= differences > threshold
    number_over=numpy.sum(values_over_t)
    val=1
    #If there are no large discontinuities in the dataset, the threshold
    #value is relatively small and thus there will be many points where the
    #difference between neighboring points is greater than the threshold value.
    if(number_over<=3):
        val=0
        ##If the user wants to plot all failing tests:
        #pyplot.plot(my_data, linestyle='None', marker='.')
        #pyplot.show()
    return val

#test_directory iterates through all files in a directory and tests them
#it returns a vector with their results
def test_directory(directoryname):
    results=[]
    listing= os.listdir(directoryname)
    for infile in listing:
        filepath=directoryname+"/"+infile
        result=test_file(filepath)
        results.append(result)
    return results

#a=test_directory("C:/Python27/Scripts/data")
#print a
```