THE OPEN UNIVERSITY OF SRI LANKA
BACHELOR OF TECHNOLOGY (level 05)
ECX 5239
PHYSICAL ELECTRONICS
FINAL EXAMINATION 2009

DATE : 24th March 2010 **TIME : 14.00 – 17.00 hours**

Answer FIVE questions. Select ONE question from Section A and FOUR questions from Section B. State your assumptions (if any) clearly. Please note that all symbols carry their usual meaning.

SECTION A:

Select **ONE** question

1.

(a)

Explain, how Paulie's exclusion principle relates to the energy band model?

(b) (8 marks)

Find the fraction of electrons in the valance band that can be thermally promoted across the energy gap into the conduction band at 30^0K and 300^0K respectively for following situations:

(i) for an intrinsic semiconductor type 1 with an energy gap of E eV
(ii) for an intrinsic semiconductor type 2 with an energy gap of 5E eV

Compare the results obtained in each case and state what you can deduce from this comparison.

(c) (16 marks)

Measurements of the conductivity of a type 3 intrinsic semiconductor has shown that its conductivity varies with temperature as $\exp(-A_0T)$ where A_0 is a constant and T is the temperature given in Kelvin. Assuming that the hole mobility in this semiconductor is half the value of its electron mobility, derive an expression for its energy gap width.

(16 marks)

[Assume for all three semiconductror types 1, 2 and 3 mentioned in above (b) and (c) the effective concentration of available electron states in the conduction band at energy E_c and the effective concentration of available electron states in the valance band at energy E_v are equal and is given by N_0 m^{-3} and is independent of temperature for the temperature ranges considered in above situations (b) and (c)]

2.
Show that a semiconductor has a minimum conductivity at a given temperature when

$$n = n_i\sqrt{\frac{\mu_h}{\mu_e}} \quad \text{and} \quad p = n_i\sqrt{\frac{\mu_e}{\mu_h}}$$

(30 marks)

Find the intrinsic and minimum conductivities for germanium at a temperature such that $n_i = 2.5 \times 10^{19}\ m^{-3}$, $\mu_e = 0.38\ m^2/Vs$ and $\mu_h = 0.19\ m^2/Vs$.

(10 marks)

SECTION B:

Select **ANY FOUR** questions. All questions carry equal marks.

Questions in this section are related to extracted pages 17 to 32 (numbered here as 4 to 11) from a MSc thesis by Adrian Solis of Department of Electrical Engineering and Computer Science, MIT, June 2005. Title of the thesis is **MIT Device Simulation WebLab: Simulator for Microelectronic Devices.** Read these pages and answer following questions.

3.
Compare and contrast from the point of view of a typical Sri Lankan student doing Electrical Engineering the ease of access/use, cost and facilities available in Weblab with that of other simulator programs such as, local simulators, online simulators and virtual experiment facility on the WWW.

4.
According to the author
(a) What is a device type and what are the components of a device state?
(b) What is a device model?

5.
(a) List
(i) the inputs
(ii) the output
of a typical device simulator program.

(b) List the features available in the user interface of Weblabsim

6.
Explain what is meant by a 'device profile'. Is it possible to define various device profiles for a single device model? Explain.

7.
(a) According to the author, why is it desirable to simulate microelectronic devices?
(b) Explain briefly what is 'Weblabsim' and for which purpose it can be used.

8.
For 'the ideal pn-junction model'
(a) write the equations which defines the device type
(b) state the model parameters
(c) give the state vector
(d) state underlying assumptions of equations written in above (a)

Equations used in the course material

$e = 1.6 \times 10^{-19}$ C
$m_e = 9.108 \times 10^{-31}$ kg
$h = 6.625 \times 10^{-34}$ Js
$c = 3 \times 10^{8}$ ms^{-2}
$k = 1.38 \times 10^{-23}$ J/k
1 eV = 1.6×10^{-19} J
$\varepsilon_o = 8.854 \times 10^{-12}$ F/m
$\Gamma = N_- n_- e + N_+ n_+ e$

$P_n = nh/2\pi r$ $\quad \lambda = h/p$ $\quad \Delta x.\Delta p \geq h$ $\quad \Delta E.\Delta t \geq h$ $\quad R = \rho.l/A$ $\quad \rho = 1/\sigma$ $\quad J = \sigma E$

$\sqrt{\overline{c^2}} = \sqrt{(3KT/m_e)}$ $\quad J = -ne\bar{v}$
$\bar{v} = e\tau E/m_e$

$R_H = -(1/ne) = (1/pe)$
$\rho_n = n_i^2/N_D$ $\quad n_p = n_i^2/N_A$
$D_e/\mu_e = KT/e$ $\quad E = hc/\lambda$
$J_e = ne\mu_e E + eD_e (dn/dx)$
$n = N_c \exp -(E_c - E_F)/KT$
$n_i = \sqrt{N_c N_v} \exp -(Eg/2KT)$
$V_0 - V = (D_e/\mu_e). \ln(n_n/n_p)$
$I_S = A [(D_e e n_p/L_p) + (D_h e p_n/L_n)]$
$W_{n0}^2 = (2\varepsilon V_0/e). [N_A/(N_A N_D + N_D^2)]$

$\mu_e = e\tau/m_e$ $\quad \mu_h = e\tau/m_h$
$\lambda = \tau\sqrt{\overline{c^2}}$ $\quad D = (\lambda/3).\sqrt{\overline{c^2}}$
$\sigma = \mu_e n e + \mu_h p e$
$J_h = pe\mu_h E - eD_h (dp/dx)$
$p = N_v \exp -(E_F - E_c)/KT$
$V_0 = (KT/e). \ln(N_D N_A/n_i^2)$
$I = I_s [\exp (eV/KT) - 1]$
$W_n^2 = W_{n0}^2 (1 - V/V_0)$ $\quad W = W_n + W_p$
$W_{p0}^2 = (2\varepsilon V_0/e). [N_D/(N_A N_D + N_A^2)]$

Introduction

1.1 Motivation

A simulator is an important engineering tool. Engineers use simulators to design everything from bridges to electrical circuits. Simulators are especially useful in microelectronics, where development cycles are long and expensive. After a certain point in the development process, mistakes become increasingly costly to fix. Thus, it is essential for circuit and device engineers to verify the correctness of their designs before they are fabricated.

In addition to being standard fare in the every day work of engineers, simulators have tremendous educational value. With a device simulator, a student can examine how the device described by a particular model behaves when presented with various inputs. Although this exploration can be done with a real device, the appropriate equipment is often prohibitively expensive. Whereas device simulators are available at reasonable prices - in fact, some are even freely available on the web [1].

Furthermore, working with a device simulator allows students to vary conditions which are hard to control in real devices. For instance, he may want to know the effect of the substrate doping level *(NsvB)* on the threshold voltage *(VT)* of an ntype MOSFET. To do this experiment using real devices, he has to identify *NSUB* for a given transistor and obtain a series of transistors spanning a range of doping levels: a daunting task. A similar simulation is easily performed using SPICE [2].

Most programs can even be configured to graph the results. Visually examining how changes in a model parameter affect device characteristics makes it easier to develop an intuition into the physical effects that the parameter captures. A device simulator lets students explore device behavior in regimes that would otherwise be infeasible or unsafe to examine. A classic example is the response of a transistor to voltage conditions that lead to destructive breakdown. Unless the experimental setup is configured correctly, sending a device into destructive breakdown carries the risk of device damage. Moreover, repeatedly sending a device outside its normal mode of operation can degrade its characteristics. The resulting drift makes it harder to compare results from two experiments performed at different times.

Finally, integrating simulations and experiments is a powerful way to learn about a device. Students can compare measurements from real devices with predictions derived from the models they learn in class. By doing this, they gain a better understanding of the conditions under which these idealized models are applicable, and where the models fail.

1.2 The MIT Device Simulation WebLab

This thesis describes the design and development of the MIT Device Simulation WebLab ("WeblabSim"), an online simulator for exploring the behavior of devices. Using a Java applet interface, a user connected to the web can specify and run simulations of microelectronic devices. To define a simulation, he chooses a device model, and then configures the

behavior of the model's parameters and terminals. When submitted, a device simulator performs the simulation on computers located at MIT, and the results are then sent back to the user's applet for graphing and further analysis.

WeblabSim exposes device simulators through a "virtual laboratory" interface. The system presents device models to the user as if they were actual devices, albeit with variable parameters that control their behavior. From the user's point of view, the terminals of the virtual device are attached to the ports of an analyzer, which he can then program as he wishes. When he is satisfied with his setup, he submits his experiment. He then receives the results of his experiment after it is seemingly performed at a remote location. The translation between the laboratory and simulator paradigms is seamless.

1.3 Related Work

1.3.1 Local Simulators

A stand-alone simulator can be considered to be the simplest virtual laboratory. In the field of microelectronics, there are various circuit simulators available. High end programs like HSPICETM[3] and PSpice®[4] offer high performance, extensive support for a wide variety of device models, and advanced analyses (e.g., sensitivity analysis, Monte Carlo analysis, smoke analysis). Simpler programs provide a more limited set of features, but for a lower price. WinSpice [1] is a circuit simulator based on Berkeley Spice3F4 [2] that is free for most uses. A license key, which activates more advanced features like parameterized sub-circuits and PSpice libraries, is available from the author for £45.1

Some circuit simulators [5, 6] are packaged with an integrated environment that lets users create, simulate and export circuit designs. These programs have a rich interface that employs a circuit building metaphor: users compose circuits by picking components and connecting them onscreen. Students can build circuits and try them out, just as they would in a real laboratory, but without expense or danger. Similar software exists for other fields of engineering; examples include Working ModelTM [7] for mechanics and CyclePad [8] for engineering thermodynamics.

1.3.2 Online Simulators

Several companies and individuals have developed circuit simulators that they have put on the web for online use. National Semiconductor runs the WEBENCH Electrical Simulator [9], an online electrical simulation tool for National components.

WEBENCH utilizes a standard SPICE simulation engine that is integrated with a graphical representation of the circuit schematic. The program displays the results in a waveform viewer, which allows multiple waveforms to be viewed together. This feature guides the user in optimizing his circuit by highlighting the differences between designs. Intersil has a similar online application called iSim [10]. Both programs restrict the user to running an AC analysis a set of predefined circuit layouts that use the company's technology. DigSim [11] is a digital logic simulator developed by Iwan van Rienen. DigSim is a Java applet that allows users to create digital schematics in a Computer Aided Drawing (CAD) environment. The

schematic can then be simulated to show how the circuit would perform if actually constructed. Mr. van Rienen has since stopped development on DigSim. However, a version of the program is still being maintained at the University of California in Berkeley [12].

1.3.3 Virtual Experiments on the World Wide Web

The Virtual Laboratory [13] at Johns Hopkins University is a collection of web-based science and engineering experiments developed for beginning students. The exercises were designed to expose high school seniors and college freshmen to experimentation, problem solving, data gathering, and scientific interpretation. Ordinarily, students were unable to practice these skills until they worked in a design laboratory in their junior or senior year. The activities cover a wide range of engineering topics, including logic circuits, heat transfer and sound propagation.

The Oxford Virtual Reality Group is hosting Virtual Chemistry [14], a three dimensional simulated laboratory for the teaching of chemistry. The site interface was modelled using virtual reality techniques. The user moves around the laboratory in a web browser window and takes part in experiments distributed around this virtual environment. The experiments feature videos and animations of the procedure being carried out, three-dimensional simulations of chemical processes and interactive sections that check the student's understanding of the material.

1.3.4 MIT Microelectronics WebLab

The MIT Microelectronics WebLab ("WebLab") [15, 16, 17, 18] is an online remote laboratory for characterizing microelectronic devices. Through a Java-enabled web browser, users all over the world can run experiments on real transistors, diodes, and other devices by means of a semiconductor parameter analyzer located at MIT.

WebLab has been very successful. Since its inception in 1998, over 1,900 students have used WebLab as part of graded assignments in classes at MIT, the National University of Singapore, and the Chalmers University of Technology in Sweden [19]. As an online remote laboratory in the microelectronics domain, WebLab has heavily influenced the WeblabSim user interface. The latest version of WebLab has a graphical representation of the experiment setup that users find to be intuitive [20, 21]. In fact, the features and user interface of WeblabSim were carefully designed to build on those of WebLab, so as to leverage WebLab's usability and familiarity.

Our ultimate goal is to integrate WebLab and WeblabSim into a single online laboratory. A student can then perform both measurements and simulations, and compare their results. We believe that the conjunction of an online remote laboratory and an online simulator will be a valuable resource in microelectronics education. WeblabSim's design and implementation has been so heavily influenced both by WebLab itself, and its underlying system blueprint, the iLab Batched Experiment Architecture (BEA) [22]. Indeed, as we will discuss in Chapter 2, WeblabSim has adopted the iLab BEA in its own design. Because of this, we have adopted the laboratory notations of WebLab and the iLab BEA for the equivalent concepts in WeblabSim. For example, we often talk of WeblabSim as an "online

laboratory", we call the system configuration the "laboratory configuration", and we refer to users' simulations as "experiments".

Chapter 2
The MIT Device Simulation WebLab

This chapter describes the problem addressed by the MIT Device Simulation WebLab, and its solution. It begins by analyzing the problem domain, resulting in a series of conceptual models on which we based the design of the system and the user interface. The following sections establish design goals based on the values of extensibility, reliability and efficiency, and then introduce a design blueprint that allows WeblabSim to meet these targets. It concludes with a high-level description of the system, which will be elaborated further in succeeding chapters.

2.1 Problem Analysis

The problem WeblabSim addresses, stated in general terms, is the design and implementation of an online device simulator. Solving this problem successfully requires understanding it. In this section, we carefully analyze the problem domain by identifying key abstractions and defining the relationships between them. This analysis will clarify the conceptual model behind the user interface presented above, and the design as it is described in the following chapters.

2.1.1 Device and Device Type

A *device* is a circuit component. Devices have at least two *terminals*, by which they are connected to the rest of a circuit. The voltage at a terminal and the current flowing into a terminal are numerical quantities that can be measured. Devices can have *observables*, which are named numerical quantities associated with the present state of the device. An example of an observable would be the dynamic resistance of a pn diode. Together, the values of the voltages and currents at the terminals of a device and the values of all the its observables comprise the device's state. We refer to these as the *components* of the device state.

A *device type* is a class of devices with similar behaviors. For instance, the class of devices with two terminals and that to the first order obey Ohm's law may be grouped into a device type called "resistor". All devices of a given type necessarily have the same number of terminals, and the same set of observables.

2.1.2 Device Model

A *device model* is a set of relations between the components of a device state.' The relations are typically expressed as equations involving the state components, constants, and

adjustable *parameters*. The equations that comprise a model may have been derived from first principles, or obtained empirically from experimental results.

A state satisfies a device model if and only if the components of the state satisfy the constraints imposed by the model's equations.2 Because the model equations mention specific components of a device state, a model is intimately associated with one device type. This corresponds to our intuition that a model describing a pn diode, for example, does not "work" for a resistor nor in general, for a different kind of device.

A device model constrains only the components of the device state that are involved in the model's equations. The observables whose values are determined by a model are the *model observables*. All other observables are left unconstrained; thus, it is not useful to consider their values when working with the particular device model.

Example: Ideal p-n Junction Model of a pn Diode [24]

Consider a silicon pn diode modelled as a one-dimensional semiconductor structure where the doping level abruptly changes from n-type, with a uniform donor concentration N_D, to p-type, with a uniform acceptor concentration N_A. Applying semiconductor physics, we find that there is a region around the junction (the *depletion region)* where the bulk charge density is non-zero. Deep inside the semiconductor, sufficiently far away from the junction, the charge density falls to zero. If this *quasineutral region* is much longer than the minority carrier diffusion length, the voltages and currents at the terminals of the diode are related by:

$$I_1 = A\left(\frac{q n_i^2 D_e}{N_A L_e} + \frac{q n_i^2 D_h}{N_D L_h}\right)\left(\exp\frac{q(V_1 - V_2)}{KT} - 1\right) \quad (2.1)$$

$$I_2 = -I_1 \quad (2.2)$$

In Equation 2.1, V_1 (V_2) and I_1 (I_2) are the voltage and current at the p-type (ntype) terminal, respectively. A is the area of the junction, n_i is the intrinsic carrier concentration, q is the electronic charge, k is Boltzmann's constant, *Le, (Lh)* is the electron (hole) diffusion length, *De, (Dh)* is the electron (hole) diffusion coefficient, and T is the temperature of the diode. The values of q, k are fixed by the laws of nature. Only *ni, Le, Lh, De, Dh,* N_D, N_A, A and T may be adjusted. These nine quantities are the parameters. These parameters are not necessarily independent of each other. For example, the diffusion lengths and the diffusion coefficients above are roughly related to the doping level in the semiconductor. In practice, Equation 2.1 is often simplified by introducing an empirical parameter *Is* (the saturation current) which represents the combined effects of *ni, Le, Lh, De, Dh,* N_D, N_A and A. This gives

$$I_1 = f(I_S, T, T_{nom})\left(\exp\frac{q(V_1 - V_2)}{KT} - 1\right) \quad (2.3)$$

where T_{nom} is the temperature at which *Is* was measured, and f is a function that captures the temperature dependence of *Is*.
Formally, the state of a device of type "silicon pn diode" is captured by the vector [V_1,I_1, V_2,I_2].

The "ideal p-n junction model" for this device type is defined by Equations 2.4-2.5.

$$I_1 = f(I_S, T, T_{nom})\left(\exp\frac{q(V_1-V_2)}{kT} - 1\right) \quad (2.4)$$

$$I_2 = -I_1 \quad (2.5)$$

In this simple model, *Is*, *T* and T_{nom} are the model parameters. For this example, we chose to not include any model observables, limiting the ideal p-n junction model to the I-V characteristics of the diode. However, the model can be developed further to yield expressions for observables (e.g., dynamic resistance, junction capacitance).

2.1.3 Device Simulator

Conceptually, a *device simulator* is a computer program that takes as input a device model, a sequence of partially-specified device states (including values for the parameters of the device model), and a set of parameter values that control the behavior of the simulator. When the simulator runs, it "fills in" the partially-specified states such that each of the completed states satisfies the device model. The output of the simulator is the sequence of states corresponding to each partial state in the input.

We call the input to the simulator a ***simulation***, and call the output the result of the simulation. A simulator implements a device model if it accepts simulations that use the given model. Most simulators implement a variety of device models; conversely, a device model can be implemented by many simulators. For example, the Level 1 SPICE MOSFET model is supported by Berkeley Spice3F4, and by all other Spice compatible circuit simulators. The implementors of a device model can choose to give a default value to some parameters in the model, so that omitting the parameter's value is equivalent to specifying the default.

Model Simplifications

Concrete realizations of a device simulator often add additional functions to make the program more convenient to use. When a device simulator is exposed to users through WeblabSim, it is presented with the following features :

- The input sequences of partial states are not specified explicitly. Instead, the user provides control statements that generate the input sequence. This means that the user does not have to specify the inputs to the device in painstaking detail at every point in time. For example, rather than giving all the voltages at 0.1 V intervals from -1.0 V to 1.0 V, the user can simply declare that he wants "the voltage swept from -1.0 V to 1.0 V, in 0.1 V intervals." The next point lists the kinds of commands that can be given to the simulator.
- In generating the input, the simulator can
  - hold a parameter, terminal current or terminal voltage at a constant value (CONS function)
  - sweep a parameter, terminal current or terminal voltage over a range of linearly- or logarithmically-spaced values (VAR1 function)

- at each point in the parameter, current or voltage sweep described in 2.1.3 above, similarly sweep the value of another such quantity (VAR2 function)
- while sweeping the current or voltage at terminal *A*, set the current or voltage at a different terminal *B* to a linear function of the quantity begin swept at *A* (**VAR1P** function)

- . . The user can specify which currents and voltages he is interested in, and only those variables are downloaded
- The simulator can evaluate expressions (user-defined functions) which are functions of the currents and voltages in the results, and other defined expressions.

If the device simulator actually does not support all these features, the WeblabSim system emulates those that are missing by processing the input and output of the simulator. To keep the system simple, we associate each device model with exactly one device simulator. This means that WeblabSim treats the Level 1 MOSFET model implemented by Berkeley Spice3F4 and the Level 1 MOSFET model implemented by WinSpice to be different device models, even though they may be functionally equivalent. This makes it much easier to determine which simulator to use to run a given simulation: we simply use the simulator associated with the device model specified in the simulation. This means that the user's choice of device model while configuring a simulation effectively determines the following: the simulator that runs the simulation, the device model to use, and the default values for any parameters that the user omits.

We avoid issues caused by incompatibilities between different simulator versions by defining different versions of the same simulator to be different simulators altogether. Thus, to the WeblabSim system, WinSpice version 1.05.01 and WinSpice version 1.05.04 are two different simulators.

2.1.4 Device Profile

A laboratory administrator may want to customize device models before presenting them to users. Device models can be quite complicated. For instance, the Level 1 junction diode model in HSPICETM has more than 23 parameters [25]. It could be desirable from an educational standpoint to limit the number of parameters that users see at a given time. For instance, if an exercise focuses on the ideal model of a silicon pn diode, students should not have to set parameters like the breakdown voltage or the flicker noise coefficient. Even though the simulator often provides default values for these parameters, the mere presence of these parameters in the device setup dialog clutters the interface and could distract the user from the task at hand.

A device profile is a customized device model. By defining a device profile, the laboratory administrator can:

* define a descriptive name (e.g., "ideal silicon pn diode") that will guide users in selecting a profile to use in their simulation
* re-define the default values for model/simulator parameters
* hide model/simulator parameters. Hidden parameters are not presented to the user in the device setup dialog. However, if the administrator specifies a default value, the parameter's default value is still passed to the device simulator.

* tell the system to withhold (omit) certain model/simulator parameters from the simulator. Omitted parameters are not passed to the device simulator at all, even though a default value was assigned to them at the device model or device profile level. 5
* set the maximum number of data points in a simulation involving this device profile

Many device profiles can be defined from a single device model. Supposing the WinSpice v1.05.04 junction diode model is installed, the administrator can create device profiles like "ideal silicon pn diode", "silicon pn diode with series resistance", "silicon pn diode with breakdown", and "germanium pn diode" from this device model by setting the appropriate parameters correctly.

Example: A Germanium pn Diode

The junction diode model implemented by WinSpice v1.05.04 has 15 parameters, nine of which are relevant to the diode's DC characteristics (including temperature dependence) [26]. If we focus on a semiconductor diode and ignore breakdown, only the parameters in Table 2.1 need to be considered.

name	description	Units	default
IS	Saturation current	A	1.0×10^{-14}
N	Ideality factor	-	1.0
RS	Series resistance	ohm	0.01
EG	Band gap energy	eV	1.11
TNOM	Param. Measu. temp	^{0}C	27
TEMP	Simulation temperature	^{0}C	27

Table 2.1: WinSpice junction diode model parameters that control the DC characteristics of a pn diode.

A simple device profile that emulates the DC characteristics of a germanium pn diode can be defined by:

1. setting the band gap energy to 0.661 eV (EG=0.661),
2. increasing the default value of the saturation current to reflect the higher intrinsic carrier concentration and larger carrier diffusion coefficients in germanium (e.g., IS=1.50E-5),
3. setting TNOM=27 and documenting that whatever value the user provides for IS is the saturation current as measured at 27 °C, and
4. hiding EG, TNOM and the parameters not present in Table 2.1, so that the user cannot change their values.

This example illustrates the differences between the concepts of a device, a device type, a device model and a device profile. A germanium pn diode is a device. Its characteristics suggest that we classify it as a "pn diode", which is its device type. PN diodes can be modeled as an "ideal p-n junction", as described in section 2.1.2; we can use this device model for the germanium diode by giving appropriate values for the model parameters. If users had to provide these values every time they wanted to simulate a germanium diode, they will soon find this process tedious and error prone. To make it easier for WeblabSim users, we can define a device profile named "germanium pn diode" that has the relevant parameters preset to the correct values.