An Idea For A Numerical Solution Of The Semiconductor Equations

John Barrett

April 2, 2009

1 Introduction

One of the most important problems in Electrical Engineering is the solution of the semiconductor equations, which govern such devices as diodes and, more importantly, transistors. The semiconductor equations are five coupled partial differential equations:

$$J_{n\cdot x}(x,t) = q\mu_n(x)n(x,t)\varepsilon_x(x,t) + qD_n(x)\frac{\partial n(x,t)}{\partial x} \tag{1}$$

$$J_{p\cdot x}(x,t) = q\mu_p(x)p(x,t)\varepsilon_x(x,t) - qD_p(x)\frac{\partial p(x,t)}{\partial x} \tag{2}$$

$$\frac{\partial n(x,t)}{\partial t} = \frac{1}{q}\frac{\partial J_{n\cdot x}(x,t)}{\partial x} + R_{gen\cdot n}(x,t) - R_{rec\cdot n}(x,t) \tag{3}$$

$$\frac{\partial p(x,t)}{\partial t} = -\frac{1}{q}\frac{\partial J_{p\cdot x}(x,t)}{\partial x} + R_{gen\cdot p}(x,t) - R_{rec\cdot p}(x,t) \tag{4}$$

$$\epsilon_{Si}\frac{\partial \varepsilon_x(x,t)}{\partial x} = q[p(x,t) - n(x,t) + N_d^+(x) - N_a^-(x)] \tag{5}$$

A quick glance at these equations is enough to give engineers a headache from countless hours trying to rework and solve them under various conditions in college. There are a few simplifications that can be made under certain conditions, such as inserting equations (1) and (2) in equations (3) and (4) to get a set of three partial differential equations (will be possibly inserted later, as the doping profiles I put into my code cause a great instability at the region change from p-type to n-type). Another simplification, which is made while taking the classes ECE 162 and ECE 216, is that most of the semiconductor analysis is made under thermal equilibrium or steady state conditions, both resulting in the partial derivatives with respect to time vanish, just allowing the spatial currents and carrier concentrations to change. Even looking at equations (1)-(5), they are already simplified to the one-dimensional model, which is good enough for getting a basic understanding of most semiconductors. This paper will attempt to establish a good model to solve these equations using numerical integration techniques that we have covered in class for systems of ordinary differential equations. The methods used here are all explicit, and therefore not unconditionally stable (as seen in the preliminary simulations of my code using MATLAB).

2 Background Theory

In order to tackle the semiconductor equations, one must have an understanding of the variables of the equations and the units that these variables take, as well as reasonable quantities of these variables. There are several pages of equations all related to equations (1)-(5), but without knowing what the variables mean, proper application of the equations cannot be done.

The most important fact to remember when dealing with the semiconductor equations is the fact that all lengths are done using centimeters instead of meters (Why? I do not know). When viewing my code, seen in the appendix, that would explain why the x vector (the vector that corresponds to the one-dimensional axis of the semiconductor) has a length of 100e-4, which in the unit system, represents 100 microns. Time is still represented in seconds, so in the code in the appendix, MATLAB only looks at the first microsecond after the p-type semiconductor is joined with the n-type semiconductor.

There are only two independent variables in the system. They are the dopant concentrations N_a and N_d. When N_a and N_d are written as functions of x, the doping profile emerges. Most semiconductors have a constant or linear doping profile, leading to an abrupt or linearly graded junction, respectively. The code in the appendix can be modified so either profile can be analyzed. N_a represents the concentration of electron acceptors, that when implanted into the silicon, become ionized at room temperature leading to the production of a hole near the acceptor ion. N_d represents the electron donor concentration, which has an extra valence electron that can move around the silicon lattice structure. Both of these concentrations have units of $\frac{\text{atoms}}{\text{cm}^3}$ and nearly all of the acceptor and donor atoms are ionized at room temperature (which is convenient since we need semiconductors to operate at standard room temperature). The constant doping and linear doping profiles are seen in figures (1) and (2) on the top of the following page.

Rest of the variables that appear in the semiconductor equations are dependent on these dopant profiles. The main dependent variables to look at are the electron concentration (n), the hole concentration (p), the current densities of both the electrons and holes ($J_{n \cdot x}$ and $J_{p \cdot x}$), respectively, the electric field (ε_x) and the potential (ψ). To begin the solution, $n(x, 0)$ and $p(x, 0)$ are set to N_d and N_a, with $n(0, t) = n(0, 0)$, $n(L, t) = n(L, 0)$, $p(0, t) = p(0, 0)$ and $p(L, t) = p(L, 0)$, which sets up the initial boundary value problem with ohmic boundaries (L is the length of the semiconductor area). Ohmic boundaries just mean that the electron and hole concentrations remain at their thermal equilibrium values for all time. It is easily inferred that the units of n and p are the same units as N_d and N_a.

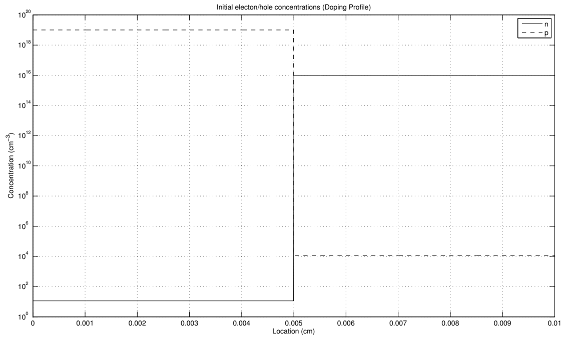

Figure 1: Abrupt Junction

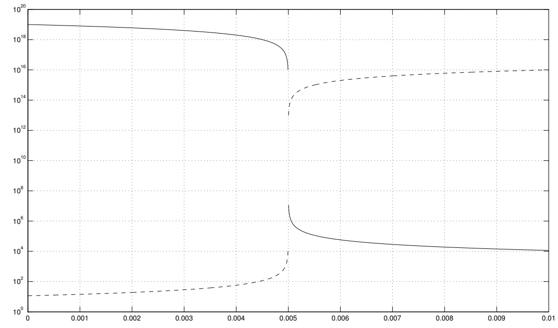

Figure 2: Linearly-graded Junction

Looking at the current densities ($J_{n \cdot x}$ and $J_{p \cdot x}$), the current caused by charges moving in the electric field (drift) and the current caused by the introduction of a concentration gradient (diffusion) are added together, note the signs take into consideration the polarity of the charges moving to keep current convention. The units of current density are amperes per square centimeter ($\frac{\text{A}}{\text{cm}^2}$) and is derived directly from solving and adding equations (1) and (2). The electric field is found by solving equation (5) or integrating the total charge and scaling by a known constant. Potential is found by integrating the negative of the electric field.

There are several other parameters seen in equations (1)-(5). These parameters are dependent only on the doping concentration as a function of space (with the exception of the recombination and generation rates seen in equations (3) and (4) as R_{rec} and R_{gen}). The space-only dependent parameters are the ion carrier mobility (μ), diffusion constant (D), the minority carrier recombination and generation lifetimes (τ_{rec} and τ_{gen}) and the diffusion length (L). The carrier mobility has the units $\frac{\text{cm}^2}{V \cdot s}$ and is used in the equation to determine the total drift current density. The diffusion constants have units of $\frac{\text{cm}^2}{s}$ and are used to calculate the portion of the current due to diffusion. The recombination lifetimes have units of seconds and are used in the calculation of the recombination and generation rates. The diffusion length is in units of centimeters, but in the code, the diffusion length is not needed to be explicitly determined.

The recombination and generation rates are determined by finding the deviation of the electron and hole concentrations from thermal equilibrium and dividing by the recombination or generation time. Recombination is a more favorable thermodynamic process, so recombination is faster by, approximately, a factor of 50 to 100. However, when recombination lifetimes are determined, only the minority carrier lifetimes are looked at. An assumption made in this paper is that the minority and majority carrier lifetimes are the same, which is usually invalid, but calculation of the majority carrier lifetimes are extremely difficult if even possible at all. All the equations relating the parameters and variables are seen below:

$$
\begin{aligned}
\mu_{p,P}(N_a) &= 49.7 + \frac{418.3}{1 + (\frac{N_a}{1.6\times 10^{17}})^{0.70}} \\
\mu_{n,P}(N_a) &= 232 + \frac{1180}{1 + (\frac{N_a}{8.0\times 10^{16}})^{0.90}} \\
\mu_{p,N}(N_d) &= 130 + \frac{370}{1 + (\frac{N_d}{8.0\times 10^{17}})^{1.25}} \\
\mu_{n,N}(N_d) &= 92 + \frac{1268}{1 + (\frac{N_d}{1.3\times 10^{17}})^{0.91}} \\
D &= \frac{\mu k_B T}{q} \\
R_{gen,P} &= -\frac{n_P(x,t) - n_P^{\circ}}{\tau_{gen,P}} \\
R_{rec,P} &= \frac{n_P(x,t) - n_P^{\circ}}{\tau_{rec,P}} \\
R_{gen,N} &= -\frac{p_N(x,t) - p_N^{\circ}}{\tau_{gen,N}} \\
R_{rec,N} &= \frac{p_N(x,t) - p_N^{\circ}}{\tau_{rec,N}} \\
\tau_{rec\cdot p,N} &= \frac{1}{7.80\times 10^{-13} N_d^+ + 1.80\times 10^{-31}(N_d^+)^2} \\
\tau_{rec\cdot n,P} &= \frac{1}{3.45\times 10^{-12} N_a^- + 9.50\times 10^{-32}(N_a^-)^2} \\
L &= \sqrt{D\cdot\tau} \\
\frac{\partial^2\psi(x,t)}{\partial x^2} &= -\frac{\rho(x,t)}{\epsilon_{Si}} \\
\rho(x,t) &= q[p(x,t) - n(x,t) + N_d^+(x) - N_a^-(x)
\end{aligned}
$$

There are also other parameters in the relation and semiconductor equations that have not been mentioned. These other values, q, ϵ_{Si}, k_B, T and n_i are constants of the materials or physical nature. The amount of charge on an electron, q, has been determined, to a first approximation, to be 1.6×10^{-19} coulombs. The permittivity of silicon ϵ_{Si} has been shown to be $11.7\times\epsilon_0$ and $\epsilon_0 = 8.85\times 10^{-14}$ farads per centimeter. The Boltzmann constant k_B, in this paper, is used as 1.38×10^{-23} joule seconds. T is defined to be the absolute temperature, around 300 for room temperature, and is measured using the Kelvin scale of temperature. The last parameter is n_i, which is the concentration of free electrons in a silicon crystal. At room temperature, the value of this constant is 1.07×10^{10}.

3 Numerical Analysis

In order to solve this system, equations (1)-(5) were discretized into the following equations:

$$J_{n\cdot x}(x,t) = q\mu_n(x)n(x,t)\varepsilon_x(x,t) + qD_n(x)\frac{n(x+dx,t)-n(x,t)}{dx} \tag{6}$$

$$J_{p\cdot x}(x,t) = q\mu_p(x)p(x,t)\varepsilon_x(x,t) - qD_p(x)\frac{p(x+dx,t)-p(x,t)}{dx} \tag{7}$$

$$\frac{n(x,t+dt)-n(x,t)}{dt} = \frac{1}{q}\frac{J_{n\cdot x}(x+dx,t)-J_{n\cdot x}(x-dx,t)}{2dx} + R_{gen\cdot n}(x,t) - R_{rec\cdot n}(x,t) \tag{8}$$

$$\frac{p(x,t+dt)-p(x,t)}{dt} = -\frac{1}{q}\frac{J_{p\cdot x}(x+dx,t)-J_{p\cdot x}(x-dx,t)}{2dx} + R_{gen\cdot p}(x,t) - R_{rec\cdot p}(x,t) \tag{9}$$

$$\epsilon_{Si}\frac{\varepsilon_x(x+dx,t)-\varepsilon_x(x,t)}{dx} = q[p(x,t)-n(x,t)+N_d^+(x)-N_a^-(x)] \tag{10}$$

It would seem like equations (6) and (7) should be used to update the electron and hole concentrations spatially, but that is an incorrect thought process. Equations (6) and (7) should only be used to determine the current densities needed for equations (8) and (9), where the electron and hole concentrations are then updated (as stated earlier, it may be more accurate to just plug in equations (6) and (7) into equations (8) and (9), but I will look at that before the final paper if I am still having convergence issues at the junction between the p and n regions). With these equations ready to go, a solution for a basic abrupt junction was attempted, but the method had issues at the junction, where there is a large jump in dopant concentrations. This means I need another boundary condition at or near the junction, but the code has not been modified to include this detail, known as the law of the junction. The only variable not solved for in the code is the potential, which will need to be added into the solver when I begin to look at different potentials applied across the semiconductor.

4 Appendix

MATLAB code for solution and plot generation

```
kB=1.38e-23;
T=300;
q=1.6e-19;
ni=1.07e10;
esi=11.7*8.85e-14;

dx=5e-6;
x=0:dx:100e-4;
dt=1e-9;
t=0:dt:1e-6;
Na=zeros(1,length(x));
Nd=zeros(1,length(x));
p=zeros(length(t),length(x));
```

```
n=zeros(length(t),length(x));
B=1e19;
As=1e16;
for i=1:length(x)/2
    Na(i)=B-(B-ni)/0.005*x(i);
    p(1,i)=B-(B-ni)/0.005*x(i);
    Nd(length(x)-i)=As-(As-ni)/0.005*x(length(x)-i);
    n(1,length(x)-i)=As-(As-ni)/0.005*x(length(x)-i);
end
Nd(length(x))=As;
n(1,length(x))=As;
for i=1:length(x)/2
    n(1,i)=ni^2/p(1,i);
    p(1,length(x)-i)=ni^2/n(1,length(x)-i);
end
p(1,length(x))=ni^2/n(1,length(x));
p0=p(1,:);
n0=n(1,:);
p(:,1)=p(1,1);
p(:,length(x))=p(1,length(x));
n(:,1)=n(1,1);
n(:,length(x))=n(1,length(x));
unP=232+1180./(1+(Na./8e16).^0.90);
upP=49.7+418.3./(1+(Na./1.6e17).^0.70);
unN=92+1268./(1+(Nd./1.3e17).^0.91);
upN=130+370./(1+(Na./8e17).^1.25);
up=zeros(1,length(unP)+length(upP));
un=zeros(1,length(unN)+length(upN));
up=unP+upP;
un=unN+upN;
Dp=(up*kB*T)/q;
Dn=(un*kB*T)/q;
trecN=zeros(1,length(x));
trecP=zeros(1,length(x));
for i=1:length(x)
    if Nd>0
        trecN(i)=1./(7.8e-13*Nd+1.8e-31*Nd.^2);
    end
    if Na>0
        trecP(i)=1./(3.45e-12*Na+9.5e-32*Na.^2);
    end
end
tgenN=50*trecN;
tgenP=50*trecP;
```

```
psi=zeros(length(t),length(x));
field=zeros(length(t),length(x));
Jnx=zeros(length(t),length(x));
Jpx=zeros(length(t),length(x));

for i=1:length(t)-1
    for j=1:length(x)-1
        field(i,j+1)=field(i,j)+dx*esi*q*(p(i,j)-n(i,j)+Nd(j)-Na(j));
    end
    Jnx(i,1)=q*un(1)*n(i,1)*field(i,1)+q*Dn(1)*(n(i,2)-n(i,1))/dx;
    Jpx(i,1)=q*up(1)*p(i,1)*field(i,1)-q*Dp(1)*(p(i,2)-p(i,1))/dx;
    a1=length(Jnx(i,:));
    a2=length(Jpx(i,:));
    Jnx(i,a1)=q*un(a1)*n(i,a1)*field(i,a1)+q*Dn(a1)*(n(i,a1)-n(i,a1-1))/dx;
    Jpx(i,a2)=q*up(a2)*p(i,a2)*field(i,a2)-q*Dp(a2)*(p(i,a2)-p(i,a2-1))/dx;
    for j=1:length(x)-1
        Jnx(i,j+1)=q*un(j+1)*n(i,j+1)*field(i,j+1)+q*Dn(j+1)*...
            (n(i,j+1)-n(i,j))/dx;
        Jpx(i,j+1)=q*up(j+1)*p(i,j+1)*field(i,j+1)-q*Dp(j+1)*...
            (p(i,j+1)-p(i,j))/dx;
    end
    for j=2:length(x)-1
        if trecN>0
            n(i+1,j)=n(i,j)+dt*(((Jnx(i,j+1)-Jnx(i,j-1))/(2*dx*q))-(n(i,j)...
                -n0(j))/trecN(j));
        else
            n(i+1,j)=n(i,j)+dt*(((Jnx(i,j+1)-Jnx(i,j-1))/(2*dx*q)));
        end
        if trecP>0
            p(i+1,j)=p(i,j)+dt*(((Jpx(i,j+1)-Jpx(i,j-1))/(-2*dx*q))-...
                (p(i,j)-p0(j))/trecP(j));
        else
            p(i+1,j)=p(i,j)+dt*(((Jpx(i,j+1)-Jpx(i,j-1))/(-2*dx*q)));
        end
    end
end
figure (1)
semilogy(x,p0,'-k',x,n0,'--k');
grid on
```

5 References

- Burden and Faires. Numerical Analysis: 8th Edition.

- Massoud, H. “ECE162 Class Notes.” Chapters 3-5, Fall 2007.
- Massoud, H. “ECE216 Lecture Slides.”Lectures 4-11 Fall 2008.
- Press, et. al. Numerical Recipes in C++: The Art of Scientific Computing.