A SCAPS script to calculate the width of the depletion layer

Marc Burgelman

20-1-2018

Correspondence address: Dept. of Electronics and Information Technology (ELIS), University of Gent, 'Belgium'.

E-mail: Marc.Burgelman@ugent.be

0. Version and file requirements

The SCAPS script discussed in this document uses:

- SCAPS version 3.3.09 of December 2020, or more recent.
- the SCAPS script file `width of SCL.script`

1. Introduction

In earlier SCAPS versions ≤ 3.3.08, the number of script variables was limited:

- 6 integer numbers: `{m}index`, where {m} stands for one of the six letters (x, y, z, u, v, w)
- 6 real (with a decimal point) numbers: `{m}value`, where {m} stands for one of the six letters (x, y, z, u, v, w)
- 6 real vectors: `{m}vector`, where {m} stands for one of the six letters (x, y, z, u, v, w).

From SCAPS ≥ 3.3.09 on, the number of these script variables is extended to 26: {m} can be any letter (without accents or so) of the Latin alphabet, thus one letter out of the set (a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z). As a result, there are changes in the Script Editor (to propose valid script commands while setting up a script) and in the Script Variable Panel, to display sets of 26 possible variables instead of 6: see the SCAPS User Manual.

The script discussed here is to illustrate these extended facilities. At the same time, a standard concept of semiconductor device physics is illustrated: the depletion layer or space charge layer (SCL).

2. Three ways to determine the space charge layer (SCL) width *W*

We take the simple definition file `simple pn.def`, and we increase the *n*-side doping density N_D a bit to make the junction more pronounced n^+p type. We will vary the *p*-side doping density N_A, and study $W(N_A)$.

2.1 Text book equation for SCL width *W*

Consult standard text books on semiconductor (device) physics, and find an equation for space charge layer width:

$$W = \sqrt{\frac{2\varepsilon_s\varepsilon_0}{q} N_{\text{eff}} \left(V_{bi} - V\right)} \tag{1}$$

where ε_0, ε_s and q have their usual meaning, V is the applied voltage, V_{bi} the diffusion or 'built-in' voltage, and N_{eff} is the 'effective' doping density that depends on the doping density N_A of the *p*-type layer and N_D of the *n*-type layer, and is mostly determined by the smallest of the two:

$$N_{eff} = \frac{N_A N_D}{N_A + N_D} \tag{2}$$

Eqs. (1) and (2) hold if the transition between neutral layer and space charge layer is abrupt, and is the doping densities N_A and N_D do not depend on position ('no grading', in SCAPS terminology). The built-in voltage V_{bi} is given by

$$V_{bi} = \frac{kT}{q} \ln\left(\frac{N_A N_D}{n_i^2} \right) \tag{3}$$

k and T have their usual meaning, and $kT/q \approx 25$ mV at room temperature; n_i 10^{10} cm^{-3} is the intrinsic carrier density, and is about 10^{10} cm^{-3} in silicon.

The implementation of Eqs. (1)-(3) in the script is straightforward, but laborious. The evaluation is done at zero voltage ($V = 0$) only. It would be a little more comfortable, if there existed script commands to take the square or the square root of a variable (scalar value or vector): now we must do with a multiplication and with the function …`power`, but even then it is laborious work… The script is annotated with comments, so that is (hopefully) understandable. For a simple job like this one, we already need a lot of script variables (values and vectors), certainly > 6! The correspondence between the physical notations in the above equations (e.g. V_{bi}, N_{eff}, …) and the SCAPS script variables avector… zvector and avalue… zvalue is listed in Table 1 at the bottom of this document.

2.2 Determine *W* from the capacitance *C* simulation

This is the standard way to 'measure' depletion layer width W: one is measuring the capacitance (per area) C, and deduces W from C with

$$W = \frac{\varepsilon_s \varepsilon_0}{C} \tag{4}$$

This Eq. (4) has a broader validity than Eq. (1), from which it is usually derived in elementary text books. The space charge layer still should be 'abrupt', but for Eq. (4) there is no need that N_A and N_D be constant, they can vary with position x (the can be *graded* in SCAPS terminology): $N_A(x)$ and $N_D(x)$.

In the script, we simulate C(0 V) by ordering a C-V simulation from V = 0 V to V = 0 V (2 points is the minimum…).

2.3 Define 'edges' of the SCL, and determine *W* from these

Since SCAPS can simulate all internal semiconductor properties, including $n(x)$ and $p(x)$, we can easily check the assumption of abrupt depletion, determine approximatively the edges of the SCL by simple inspection and then deduce W from that. This is shown in Fig. 1. The functions $p(x)$ and $n(x)$ are retrieved in the script, with commands `get energybands.p`

and `get energybands.n`. The positions where $p(x_{left}) = N_A/2$, and $n(x_{right}) = N_D/2$ are found with the `math interpolate` command (the factor ½ can be varied in the script,you can play with it). And the SCL width is then determined as $W = x_{right} - x_{left}$, with the `math scalarsubtract` command. Keep the Manual at hand to see how these commands work!

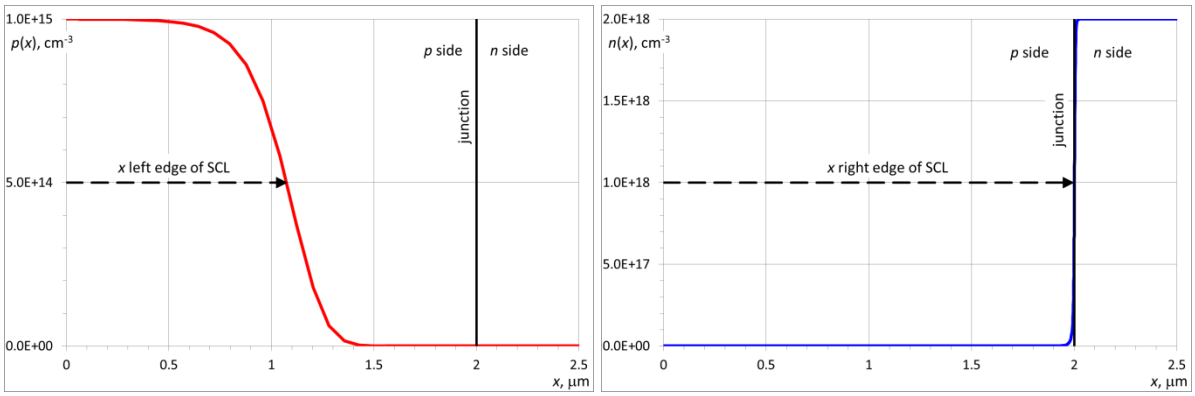

Fig. 1 The simulated hole density $p(x)$ (red, left) and electron density $n(x)$ (blue, right). The right side of the SCL (transition of $n(x)$ from N_D to 0) almost coincides with the junction at 2 µm, and is very abrupt. The left side (transition of $p(x)$ from N_A to 0) is much less abrupt, even rather gradual. We define the edges of the SCL as the x positions where $p(x) = N_A/2$ and $n(x) = N_D/2$ (the factor ½ is somewhat an arbitrary criterion, the user can play with it).

2.4 Numerical precaution ('trick' to avoid convergence failure here

In this script, we vary N_A(p-layer) from 10^{12} cm^{-3} to 10^{18} cm^{-3}. In this range, the depletion layer W varies from about 20 µm to 50 nm (the script should be set-up and have been run before we can know that ☹, but we can also consult at text book with a graph of $W(N_A)$ – such graphs are easily found, thus it is much faster than working out the script, and our imaginary semiconductor of `simple pn.def` is to very different from silicon… ☺).

In a first try, one could take the p-layer thickness d > 20 µm, so that it always can accommodate for the space charge layer of thickness W. However, we observe (by trying out…) that convergence failure occurs at high N_A, when the p-layer is too thick. Therefore, we will vary d with N_A so that it is always a little bit, but not too much, larger than W. A variation of d from 100 µm to 1 µm, as shown in Fig. 2, does the job.

We aim to present the dependence of W on the p-layer acceptor density N_A, but the analytical Eqs. (1) and (2) learn that W depends on N_{eff} rather than directly on N_A, and (very roughly said) $N_A \approx \min(N_A, N_D)$. Thus, we can only expect a real influence of N_A on W when $N_A \gg N_D$, or at least $N_A \geq N_D$. We have therefore set the donor density N_D in the n-layer of `simple pn.def` to 1×10^{18} cm^{-3}, instead of keeping the $N_D = 10^{17}$ cm^{-3} value in the definition file.

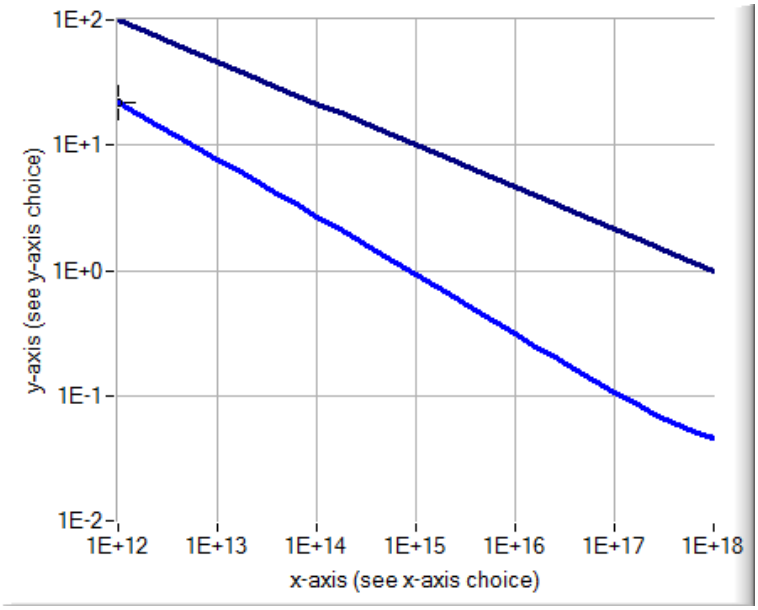

Fig. 2 The variation of N_A (the *x*-axis range, in cm^{-3}) and the thickness *d* of the *p*-layer (**black** line; in μm) set in the script. This ensures that *d* is always larger, but not too much larger, than the SCL width *W* (blue line; in μm). The blue $W(N_A)$ line is a result of the script calculation.

2.5 Comparison of *W* by the three methods

Fig. 3. shows the Script Graphs panel after executing the script. The three $W(N_A)$ plots, obtained by the 3 methods above, almost coincide, and the relative difference between the three methods is always < 10%, and around or lower than 1 % over much of the N_A-range: let us call this very acceptable ☺.

3. Conclusions

- Setting up a working SCAPS script is not so easy-going as one could hope for…
- It is highly advised to insert `show scriptvariable` commands at many places in the script: one can follow if all is working and to desire, and above all, you have the chance to stop the script execution. When all is well, you can out-comment most or all `show scriptvariable` commands. Also, can be more safe to start with a lower number of N_A points, e.g. 3 instead of the actual 31 would be enough to judge the correct working of the script, and much faster.
- When you will need many script variables to implement your script, it can be useful to make a table of the script names of variables (such as aindex, pvalue, svector…) and the data/properties these contain in your problem. For this script, this was done in Table 1 below.
- The basic equation (1) for SCL width *W* was set up by Shockley and contemporary semiconductor pioneers, based on simple assumptions on 'abrupt depletion'. They did not have computers, programs or even calculators at their disposal, they were actually in the course of inventing the components (transistors, diodes) that would be used many years later to make computers… And yet their insight and intuition was remarkably accurate!

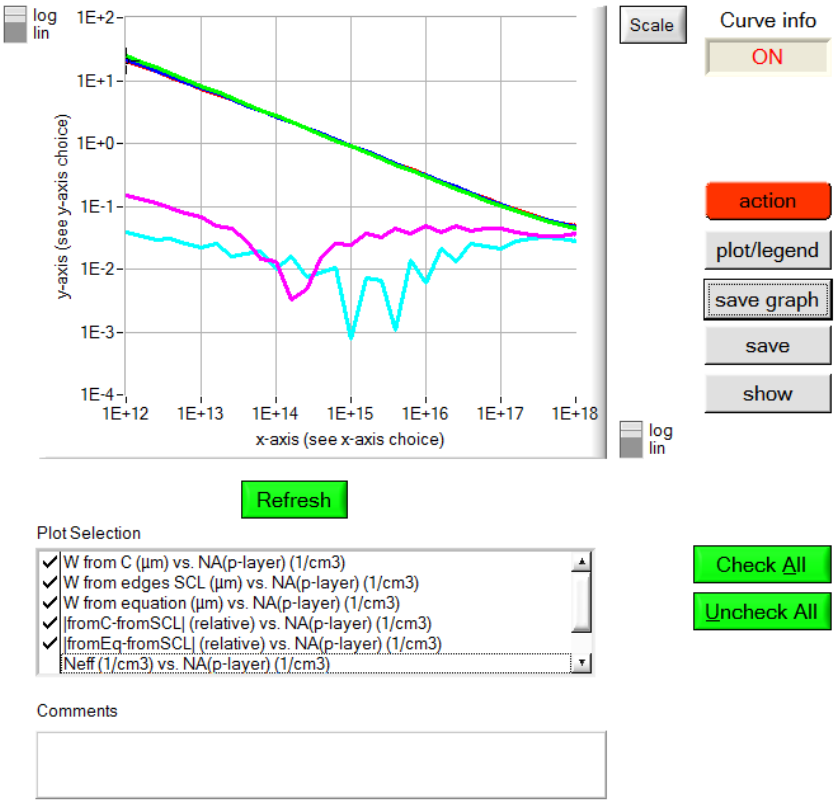

Fig. 3 Script graph results. $W(N_A)$ in a log-log plot (W is in µm, N_A in cm^{-3}). The 3 W plots (red, blue and green line) almost coincide. Also $|W_{\text{from } C} - W_{\text{from SCL}}|$ (cyan) and $|W_{\text{analytic Eq.}} - W_{\text{from SCL}}|$ (magenta) are plotted, in relative units. The deviation is always less than 10%, and less than 1 % over most of the range.

Table 1 Meaning of script vectors and script values used in the script `width of SCL.script`… there are many more variables used than the (only) 6 allowed in earlier versions. But there are no *index*-variables used (such as aindex or qindex).

avector		**a**value	
bvector		**b**value	
cvector	C of C-V simulation; for last N_A value	**c**value	C(0 Volt)
dvector	$d(N_A)$, thickness of p-layer	**d**value	conversion factor cm → µm
evector	n of $n(x)$ simulation; for last N_A value	**e**value	ε_0
fvector		**f**value	conversion from C(nF/cm^2) and ε_0(F/cm) to W(µm)
gvector		**g**value	factor $\frac{2\varepsilon_0\varepsilon_s}{q}$ in analytic eq.

hvector	p of $p(x)$ simulation; for last N_A value	**h**value	=½, exponent in analytic eq.
ivector	$V_{bi}(N_A)$ analytic calculation	**i**value	n_i and later n_i^2
jvector		**j**value	
kvector		**k**value	kT/q = 0.025 mV (room T)
lvector	left edge of SCL (μm) (N_A)	**l**value	left edge x of SCL (μm)
mvector	$N_{eff} = \frac{N_A N_D}{N_A + N_D}$ analyt. calc. (for all N_A)	**m**value	$N_{eff} = \frac{N_A N_D}{N_A + N_D}$ (for one N_A)
nvector	N_A, varied here $10^{12} \rightarrow 10^{18}$ cm^{-3}	**n**value	N_D of n-layer, and later yvalue*N_D
ovector		**o**value	
pvector		**p**value	N_A of p-layer, and later yvalue*N_A
qvector		**q**value	q, elementary charge
rvector	right edge of SCL (μm) (N_A)	**r**value	ε_s , relative dielect. const.
svector		**s**value	
tvector	$\frac{\lvert W_{\text{from } C} - W_{\text{from } SCL}\rvert}{W_{\text{from } SCL}}(N_A)$	**t**value	
uvector	V of C-V simulation; for last N_A value	**u**value	
vvector	W from 1/C (N_A)	**v**value	V_{bi} for one N_A, analytic eq.
wvector	W from edges SCL (N_A)	**w**value	W from edges SCL, for one N_A
xvector	position x, μm; from $n((x)$ and $p(x)$ simulations; for last N_A value	**x**value	right edge x of SCL (μm)
yvector	W from analytic equation (N_A)	**y**value	here = 0.5, a fraction of N_A and N_D to determine edge of SCL (somewhat arbitrary)
zvector	$\frac{\lvert W_{\text{anal. Eq.}} - W_{\text{from } SCL}\rvert}{W_{\text{from } SCL}}(N_A)$	**z**value	factor to go from 1/C (cm^2/nF) to W (μm)