Ref. Ares(2013)222447 - 20/02/2013

Gant Agreement N° 257111

SQWIRE

Silicon Quantum Wire Transistors

Start date of project: 1 September 2010
Duration: 36 Months

Coordination Action
Information and Communication Technologies

Deliverable D1.4

User Guide for Quantum Simulator
Due date of deliverable: Feb. 28, 2013
Actual submission date: Feb 28. 2013
Organisation name of lead contractor for this deliverable: MAGWEL

Approval		Revision [1.0]
WP Leader	SP Leader	Coordinator

Partner	**Please, give a short description (1-3 sentences) of partners contribution to this deliverable**
MAGWEL	Construction of the GUI and solver kernel for the NEGF
Tyndall	Constriction models for quantum wires
IMEC	Semi-analytic models for quantum wires

Dissimination Level		
PU	Public	X
PP	Restricted to other programme paticipants	
RE	Restricted to a group specified bu the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

1. Aim of this study

The goal of this activity is to provide a user's guide for quantum simulator. The Non-Equilibrium Green Function formalism is implemented in the Magwel commercial simulation package, targeting development of a software tool with a state-of-the art Graphical User Interface (GUI) that can handle quantum effects. The implementation includes: • Definition of the design flow • Integration of the software components into a modular library • Optimization of kernels to enable ultra-efficient dimensioning of the device architectures.

1.1. Abstract

This deliverable describes the work that was done to construct a software tool suite with state-of-the-art GUI technology for the simulation of quasi-1D quantum structures, such as quantum wires or modulated quantum wires.

1.2. Introduction

Within the project SQWIRE there has been a considerable amount of work performed dealing with the simulation of quantum structures. Various approaches have been followed, e.g. whereas some methods [1] are based on a in-depth analytic manipulation of the Schrödinger equation by selected specific coordinate systems (in practice: cylindrical coordinates for quantum wires) with a corresponding set of base functions, alternative methods have also been advocated. The latter are from the start based on numerical techniques and depend much less on some specific underlying symmetry. The non-equilibrium Green function formalism has gained a lot of interest over the last decade, last but not least because much of the software was distributed as freely obtainable as Matlab scripts. However, such toolboxes require quite some initiative from the novice user and it is not straightforward to upgrade these toolboxes to one's specific needs, since it often requires to modify parts of the underlying compute kernels. For this reason it has been decided in the Sqwire project, to allocates a specific task to put the quantum-transport compute kernels in a software environment such that novice users can quickly apply these kernels to the quantum-wire structures of their interest, i.e. specific layouts. Within the Sqwire project, the starting point for this task has been the device simulation software of MAGWEL. In this deliverable we present a detailed description of the activities to achieve this goal. In particular, the deliverable provides a chapter that presents a detailed user guide description. However (as always) the main purpose of a user guide is that it should not be read and only be consulted in 'emergency cases'. The GUI should be 'self-evident' meaning that once the flow of desired computation is known, the user should be able by selecting appropriate menus the various details for launching the simulation. This document is organized as follows: in chapter 2 we review the NEGF formalism. It is basically a summary of the paper of Polizzi, Wang and Lundstrom[1] with emphasis on those aspects dealing with an actual numerical implementation. This work itself is based on earlier version of the Green function approach by Datta [2]. In chapter 3, we present a number of detailed steps that were performed to get this formalism operational in the MAGWEL software. Once having the Green function implementation in place, we continue in chapter 4, with the design decisions that underlay the graphical user interface.

2. NEGF formalism (summary)

The starting point for the NEGF formalism is the Schrödinger equation corresponding to the three dimensional Hamiltonian for electrons in the effective mass approximation. The effective masses are determined by the curvature of the energy-momentum surfaces as three-dimensional ellipsoids.

$$\begin{aligned} -\hbar^2/2\ \ \partial/\partial x\left(1/m_x(x,y)\partial/\partial x\right)-\hbar^2/2\ \ \partial/\partial y\left(1/m_y(x,y)\partial/\partial y\right) \\ -\hbar^2/2\, m_z(x,y)\partial^2/\partial z^2+U(x,y,z) \end{aligned} \qquad (1)$$

The masses $m_x(x,y)$, $m_y(x,y)$, $m_z(x,y)$ depend on the material at the position (x,y). The conversion to m_t and m_l requires a reference to the wafer and transport orientations.

The quantum wire aspects are highlighted by selecting a specific formulation of the generic three-dimensional wave function such that a clear splitting is achieved of the transverse and longitudinal modes.

$$\psi(x,y,z)=\sum_n \varphi_n(z)\xi^n(x,y;z) \qquad (2)$$

This expression accounts for wave functions that are modulated along the wire (channel) such that the transverse confinement is made explicit, whereas the longitudinal variation is accounted for by the z dependence. The latter can be due to variations in the the wire properties along the longitudinal direction as well as the boundary conditions that are applied at the beginning and end of the channel. For this reason the energy eigenvalues are still position (z) dependent.

$$\begin{aligned} \left[-\hbar^2/2\ \ \partial/\partial x\left(1/m_x(x,y)\partial/\partial x\right)-\hbar^2/2\ \ \partial/\partial y\left(1/m_y(x,y)\partial/\partial y\right)+U(x,y;z)\right]\xi^n(x,y;z)= \\ E^n(z)\xi^n(x,y;z) \end{aligned} \qquad (3)$$

In order to compute the Green functions we need to compute the following slice integrals:

$$\begin{aligned} a_{mn} &= \oint dxdy \frac{1}{m_z(x,y)} \xi^m(x,y;z)\xi^n(x,y;z) \\ b_{mn} &= \oint dxdy \frac{1}{m_z(x,y)} \xi^m(x,y;z)\frac{\partial}{\partial z}\xi^n(x,y;z) \\ c_{mn} &= \oint dxdy \frac{1}{m_z(x,y)} \xi^m(x,y;z)\frac{\partial^2}{\partial z^2}\xi^n(x,y;z) \end{aligned} \qquad (4)$$

These integrals arise because of substitution of the (1) into the 3D Schrödinger equation.

3. Implementation details of the Green function methods (MAGWEL)

The solution procedures of the MAGWEL solver are based on the so-called mimetic methods [3]. The underlying idea is that numerical discretization schemes should respect conservation laws and physical symmetry principles (Noether's theorem) when converting continuous equations into discretized equations. We 'translate' these requirements by noting the two-dimensional Schrödinger equation can be reformulated using a two-dimensional flux law. Defining a 2D flux :

$$\vec{F}^n(x,y;z)=\begin{bmatrix}-\hbar^2/2\,m_x(x,y)\;\;\partial\xi^n(x,y;z)/\partial x\\ -\hbar^2/2\,m_y(x,y)\;\;\partial\xi^n(x,y;z)/\partial y\end{bmatrix} \tag{5}$$

and a defining a two-dimensional source:

$$\sum{}^n(x,y;z)=\left(-E^n(z)+U(x,y;z)\right)\xi^n(x,y;z) \tag{6}$$

the 2D Schroedinger equation becomes:

$$\nabla_{2D}.\vec{F}^n+\sum{}^n(x,y;z)=0 \tag{7}$$

Next apply the finite-integration method for each node

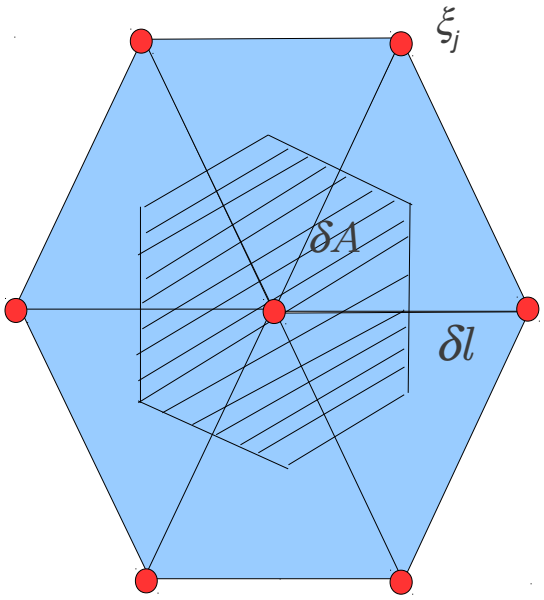


Fig 1. Assembling the 2D Schrödinger equation according to the mimetic method

$$\int_{\delta A} dS\left(\nabla_{2D}.\vec{F}^n+\sum{}^n(x,y;z)\right)=0 \tag{8}$$

Elaborating this flux-conservation rule for every node in the 2D planes results into the following discredited approximations

$$\int_{\delta A} dS\left(\nabla_{2D}.\vec{F}^{n}\right)=\oint_{\partial(\delta A)}\vec{dl}.\vec{F}^{n} \simeq\left(-\hbar^{2}/2\,m\left(\xi_{k}^{n}-\xi_{i}^{n}\right)\right)\ \delta l \tag{9}$$

$$\int_{\delta A} dS\sum{}^{n}(x,y;z)\simeq\delta A\ \left(-E^{n}(z)+U(x_{i},y_{i};z)\right)\xi_{i}^{n} \tag{10}$$

Collecting all nodal equations for each plane results into a matrix system for the values of the wave function in each node

$$\begin{bmatrix} a_{11} & a_{12} & . & . & a_{1n} \\ a_{12} & a_{22} & . & . & a_{2n} \\ . & . & . & . & . \\ a_{n1} & a_{n2} & . & . & a_{nn} \end{bmatrix} . \begin{bmatrix} \xi_1 \\ \xi_2 \\ . \\ \xi_n \end{bmatrix} = E^{n} \begin{bmatrix} \xi_1 \\ \xi_2 \\ . \\ \xi_n \end{bmatrix} \tag{11}$$

In here, the nodal wave function values are x_i . In the implementation that is constructed in the MAGWEL solver, the oxide leakage currents are suppressed: i.e. x_i s real , and $x = 0$ at the edge of the slice. There for the degrees of freedom are attached to inner slice nodes and a discrete eigenvalue problem is constructed of size of the order of number of nodes in a slice (~400).

It should be emphasized that the matrix is sparse. Furthermore, we are only interested in the lowest eigenvalues. These facts correspond to an ideal situation for applying the Jacobi-Davidson algorithm (JD) algorithm for computing eigenvalues [4].

The implementation of the quantum transport module in the MAGWEL solver is submitted to a series of constraints. First of all, the existing code should not be corrupted by the additional parts. Nor should the new parts jeopardize the run times of the existing parts. Therefore a very modular approach is recommended. Here we have applied the guidelines that the eigenvalue and wave function information is accessible through a pointer for each node. The existing data structure has already a class (“nodedata”) that contains information of the nodal voltages, lattice temperatures and carrier densities. Now a pointer is added the refers to a data structure that contains all information over the 2D Schrödinger eigenvalue problem.

```
class EigenSystem {

public:
      /** Construct a EigenSystem **/
      EigenSystem(const int size=0);
      ~EigenSystem();
      void clear();
      void initializeEigenSystem(int seed);
      double getWaveFunctionValue(int kk) const;
      void setWaveFunctionValue(int kk, double value);
      int getSize();
      std::vector<double>& getEigenValues();
private:
      int m_size;
      std::vector<double> waveFunctionValues;
      std::vector<double> eigenValues;
};
```

Header file for the eigenvalue information. It is a member of the class that contains the node data.

For each slice we make a finite element representation of the eigen function:

$$\xi^n(x,y;z)=\sum_{e\in T_e}\sum_{p=1}^{3}\xi^n(x_p,y_p;z)T_e^p(x,y) \qquad (12)$$

The base functions T(x,y) are the usual finite-element functions that are zero or one in the corners of the triangle [5].

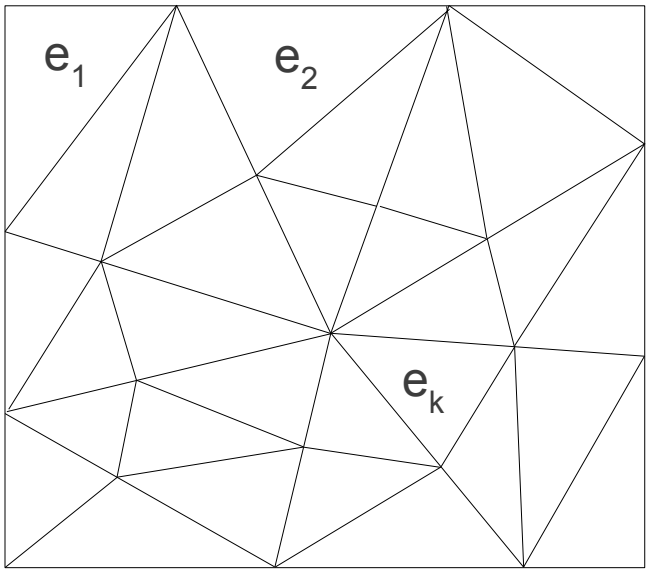


Fig. 2 Triangulation of the 2D slice.

The base functions is $T^p_e(x,y)$ assigned to triangular elements: e_k and $T^p_e = 1$ if $(x,y) = (x_p, y_p)$.

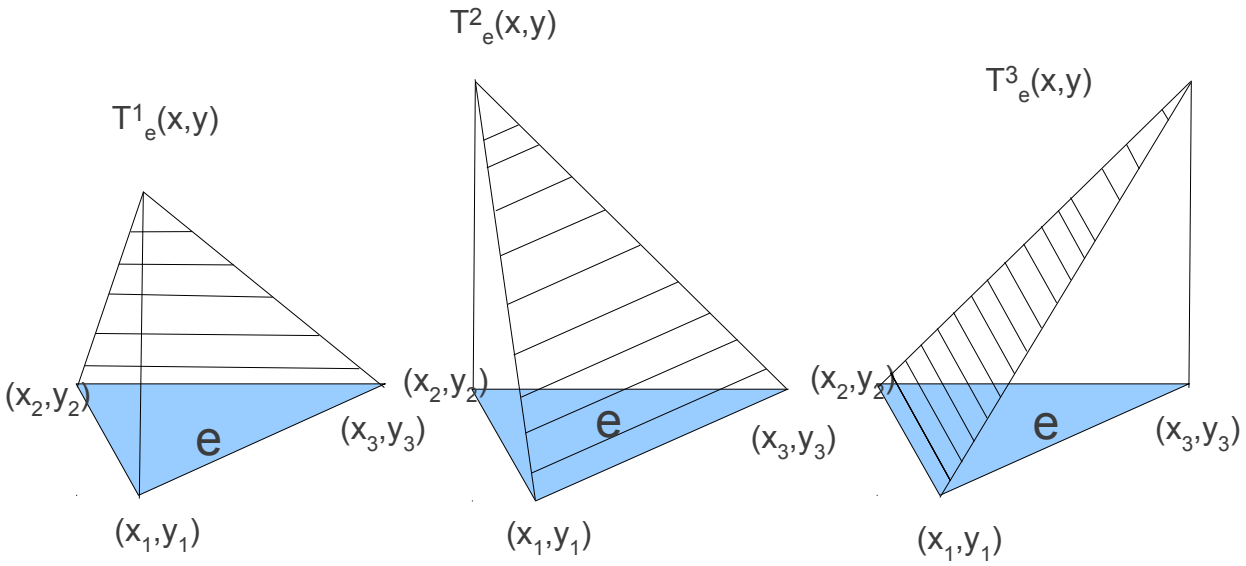


Fig. 3 Base elements for the 2D FEM description of the wave functions.

The factors a_{mn} become:

$$a_{mn}=\sum_{e\in T_e}\sum_{p=1}^{3}\sum_{q=1}^{3}\xi^m(x_p,y_p;z)\xi^n(x_q,y_q;z)\frac{1}{m_z(e)}\frac{A(e)}{12}(1+\delta_{pq})$$
$$A(e)=\frac{1}{2}\begin{vmatrix}1 & x_1 & y_1\\ 1 & x_2 & y_2\\ 1 & x_3 & y_3\end{vmatrix} \qquad (13)$$

A(e) is the area of element **e** and (x_1,y_1), (x_2,y_2), (x_3,y_3) are the corners in anti-clock wise order oeach element. The factors b_{mn} require differentiation. We use a central-finite difference formulae:

$$\frac{d\xi(z)}{dz}=(d_F+d_B)^{-1}\frac{d_B}{d_F}\xi(z+d_F)-(d_F+d_B)^{-1}\frac{d_F}{d_B}\xi(z-d_B)+\frac{d_F-d_B}{d_Bd_F}\xi(z) \qquad (14)$$

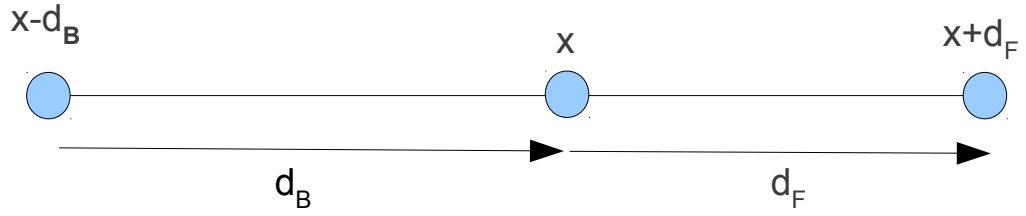


Fig. 4 Meaning of the parameters for central-finite difference.

Applying this not-necessarily equal distance central finite difference requires that the computation of b_{mn} is done according to (15).

$$b_{mn}=\sum_{e\in T_e}\sum_{p}\sum_{q}\frac{1}{m_e(z)}\xi_p^m(z)\xi_q^n(z+dz_F)\frac{1}{dz_F+dz_B}\frac{dz_B}{dz_F}\frac{A(e)}{12}(1+\delta_{pq})$$
$$-\sum_{e\in T_e}\sum_{p}\sum_{q}\frac{1}{m_e(z)}\xi_p^m(z)\xi_q^n(z-dz_B)\frac{1}{dz_F+dz_B}\frac{dz_F}{dz_B}\frac{A(e)}{12}(1+\delta_{pq}) \qquad (15)$$
$$+\sum_{e\in T_e}\sum_{p}\sum_{q}\frac{1}{m_e(z)}\xi_p^m(z)\xi_q^n(z)\frac{dz_F-dz_B}{dz_B*dz_F}\frac{A(e)}{12}(1+\delta_{pq})$$

The computing of c_{mn} requires a second derivative in central finite difference.

$$\frac{d^2\xi}{dz^2}=\frac{1}{dz_F^2}\left(1-\frac{dz_B-dz_F}{dz_B+dz_F}\right)\quad \xi(z+dz_F)$$
$$+\frac{1}{dz_B^2}\left(1+\frac{dz_B-dz_F}{dz_B+dz_F}\right)\quad \xi(z-dz_B) \qquad (16)$$
$$-\frac{1}{dz_B^2\,dz_F^2}\left[dz_B^2+dz_F^2+\left(dz_B-dz_F\right)^2\right]\quad \xi(z)$$

Substitution in c_{mn} gives the finite difference result. By looping over all cells we are able to compute all elements $a_{m\,n}(z_k)$, $b_{m\,n}(z_k)$, $c_{m\,n}(z_k)$. The 1D part of the Schrödinger equation is:

$$-\frac{\hbar^2}{2}\sum_n a_{mn}(z)\frac{\partial^2}{\partial z^2}\varphi^n(z)-\frac{\hbar^2}{2}\sum_n c_{mn}(z)\varphi^n(z)$$
$$-\frac{\hbar^2}{2}\sum_n b_{mn}(z)\frac{\partial}{\partial z}\varphi^n(z)+E^m_{subband}(z)\varphi^m(z)=E\,\varphi^m(z) \qquad (17)$$

We can organize the 1D wave function data as:

$$\phi=\begin{matrix}\phi^1(x_1)\\ \phi^2(x_1)\\ \phi^3(x_1)\\ \cdots\\ \phi^M(x_1)\\ \phi^1(x_2)\\ \phi^2(x_2)\\ \cdots\\ \cdots\\ \phi^{M-1}(x_N)\\ \phi^M(x_N)\end{matrix}\quad \text{or as}\quad \varphi=\left|\begin{matrix}\varphi^1(x_1)\\ \varphi^1(x_2)\\ \varphi^1(x_3)\\ \cdots\\ \varphi^1(x_N)\\ \varphi^2(x_1)\\ \varphi^2(x_2)\\ \cdots\\ \cdots\\ \varphi^M(x_{N-1})\\ \varphi^M(x_N)\end{matrix}\right| \qquad (18)$$

We have chosen the data organized according to the second version but there is no fundamental reason for doing so.

The 1D Hamiltonian requires the d^2/dz^2 and d/dz operators acting on ϕ. Therefore the continuous operators are replaced by the discretized operators as given in (19) and (20).

$$\frac{d^2\varphi}{d\varphi^2} \leftarrow \frac{1}{dz_F^2}\left(1-\frac{dz_B-dz_F}{dz_B+dz_F}\right)\ \varphi(z+dz_F)$$
$$+\frac{1}{dz_B^2}\left(1+\frac{dz_B-dz_F}{dz_B+dz_F}\right)\ \varphi(z-dz_B)-\frac{1}{dz_B^2\,dz_F^2}\left[dz_B^2+dz_F^2+(dz_B-dz_F)^2\right]\ \varphi(z) \qquad (19)$$

$$\frac{d\varphi}{dz} \leftarrow \frac{1}{dz_B+dz_F}\frac{dz_B}{dz_F}\varphi(z+dz_F)$$
$$-\frac{1}{dz_B+dz_F}\frac{dz_F}{dz_B}\varphi(z-dz_B)+\frac{dz_F-dz_B}{dz_B\,dz_F}\varphi(z) \qquad (20)$$

3.1. The Green function

The Green function is inverse of (E-H). This is the short version of saying what a Green function is. It is worthwhile to spend a few more words on that. Suppose we have some differential equation, say the Schrödinger equation.

$$i\hbar\frac{\partial\psi}{\partial t}=H\psi \qquad (21)$$

This equation attempt to inform us how the wave function will change in an small time interval δt under the effect of the Hamilton operator H. In other words: if we know the wave function at some time t, we can compute the wave function at a later time as

$$\psi(x,t)=\int d^3x'\,G(x,x';t,t')\psi(x',t') \qquad (22)$$

Equation (21) and (22) contain the same information. Causality requires that we only information at later time instances is computable from some solution at some instance t. Therefore it is convenient to define

G(x,x';t,t') = 0 for t' < t. This is an essential ingredient of Green functions. They correlate information in temporal order. The Green function methods all have in common the At the end point (slices) we close the set up of G with *outgoing* plane waves.

To gain more familiarity with the causality requirement, imagine a Hamiltonian that is constan in time. Then the wave function can be expanded as

$$\psi(x,t)=\sum_n c_n\ \ u(x)e^{-\frac{i}{\hbar}E_n t} \qquad (23)$$

The Green function then is for t > t' :

$$G(x,t;x',t')=\sum_n u_n(x)u_n(x')e^{-\frac{i}{\hbar}E_n(t-t')} \qquad (24)$$

For t < t' equation (24) is not valid, but it can be readily adapted as

$$G(x,t;x',t')=\theta(t-t')\sum_n u_n(x)u_n(x')e^{-\frac{i}{\hbar}E_n(t-t')} \qquad (26)$$

The Green function is the solution of

$$(i\hbar\frac{\partial}{\partial t}-H)G=i\delta(x-x')\delta(t-t') \qquad (27)$$

The Green function is the amplitude for arrival at (x,t) provided the departure took place at (x',t'). As such it contains the information to calculate transport fron source to drain as a special case.

In order to illustrate computation of the Green function we consider a simple design as given in Fig. 5.

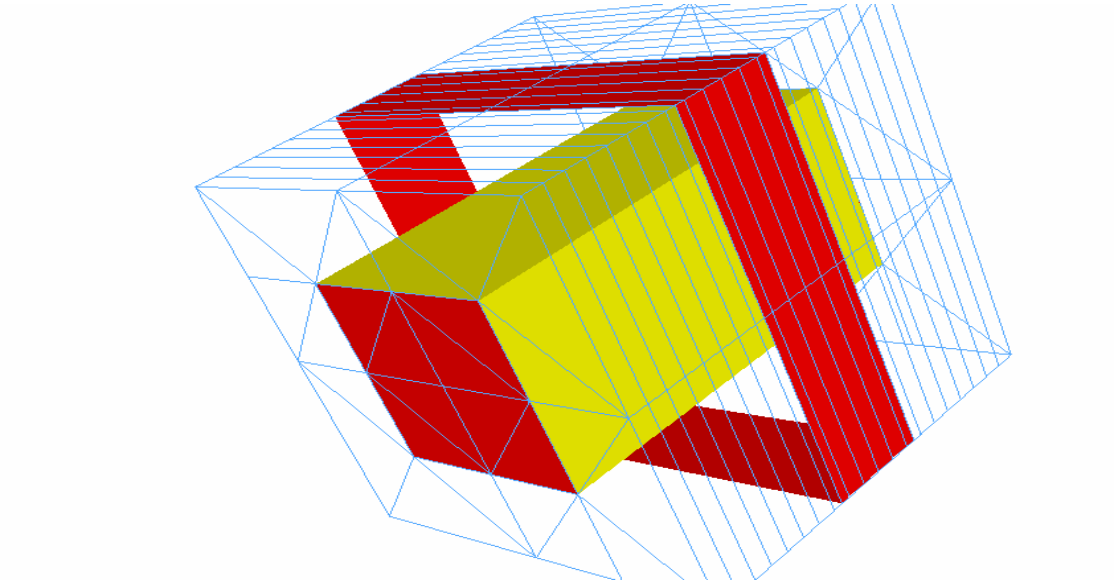
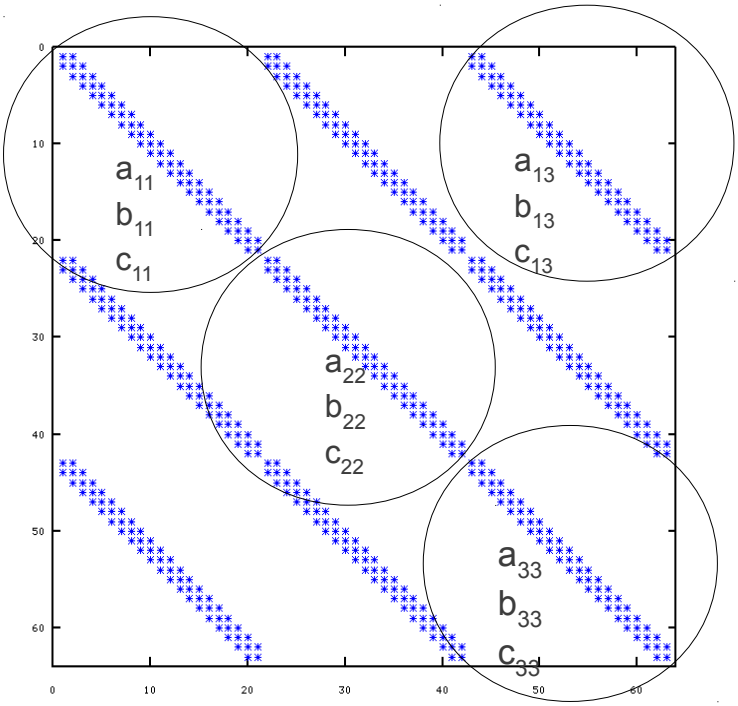


Fig. 5 Example of a quantum block wire.

The quantum wire has 21 slices (edge slices do not counted), in each slice there are 9 nodes in each slice that are not edge nodes and finally we assume that 3 eigenstates participate in the calculation.



33.9868, 38.8565

Fig 6. Sparseness pattern of the inverse of the Green function (E-H).

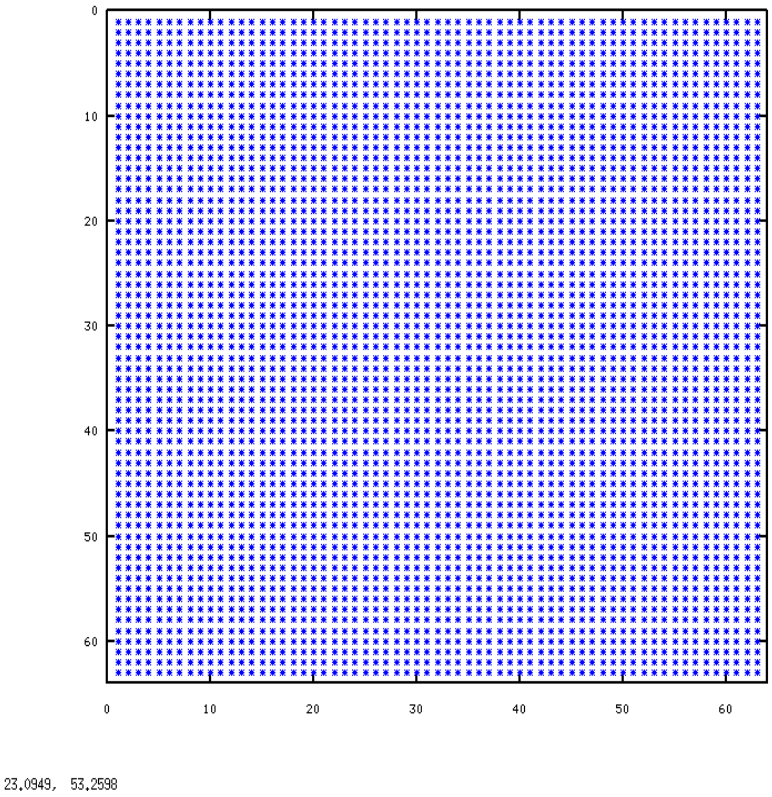


Fig. 7 The NEGF matrix is completely filled.

The source and drain are attached to the structure via self energies [1,2].

$$G=\left(E-H-\Sigma_{source}-\Sigma_{drain}\right)^{-1} \tag{28}$$

The spectral density functions are computed from the Green function and the self energies.

$$A_{s,d}(E)=G(E)\Gamma_{s,d}(E)G^{+}(E) \qquad \Gamma_{s,d}(E)=j\left(\Sigma_{s,d}(E)-\Sigma_{s,d}^{+}\right)(E) \tag{29}$$

Remember the coordinate dependence

$$A_{s,d}(E,z_i,z_j)=G(E,z_i,z_j)\Gamma_{s,d}(E)G^{+}(E,z_i,z_j)$$
$$\Gamma_{s,d}(E)=j\left(\Sigma_{s,d}(E)-\Sigma^{+}_{s,d}\right)(E) \qquad (30)$$

Finally we need the local density of states which correspond to the diagonal elements.

$$D_{s,d}(E,z_i)=\frac{1}{\pi a}\left[A_{s,d}(E,z_i,z_i)\right] \qquad (31)$$

Note that the formulas to which the Green function method refers to, still have the energy variable as one of the arguments. However, this is an integration variable and an integral over E has to be done to get the 1D electron concentration.

$$n^{m}_{1D}(z_i)=\int_{-\infty}^{\infty} dE\left(D_s(E,z_i)+D_d(E,z_i)\right) \qquad (32)$$

As a consequence, the Green function (*inverse*) has to be evaluated for every "E" again. The 3D electron concentration is

$$n^{m}_{3D}(x_i,y_i,z_i)=n^{m}_{3D}(z_i)\left|\xi^{m}(x_I,y_i;z_i)^2\right| \qquad (33)$$

This enters into Poisson equation.

After self-consistent solving the current is computed

$$I_{SD}=\int_{-\infty}^{\infty} dE\,\Gamma(E)\left(f_s(\mu_s,E)-f(\mu_D,E)\right) \qquad (34)$$

In order to perform the integral we have applied the NAG routine D01AJF which is a general-purpose integrator which calculates an approximation to the integral of a function f(x) over a finite interval [a,b].

The routine is based on the the QUADPACK routine QAGS developed by Piessens et al. [6]. It is an adaptive routine using the Gauss 10-point and Kronrod 21-point rules. In order to select the interval end-points we apply the method of [7].

3.2. 'Solve without solving'

One of the purposes of getting fast turn-around times in computation is to perform the inversion fast. A possible approach is to avoid direct solving and to use iterative method. The latter are known to be fast for sparse systems. Here we propose an iterative scheme for solving the Green function. To be more precise: we do not solve for the Green function explicitly, but determine its impact in the expressions of real interest which are actually convolutions of the Green functions. The the goal is to find

$$A = G\Gamma G^{+} \tag{35}$$

Consider: $M K^{+} = \Gamma$ where $K^{+}=(G^{+})^{-1}$ is sparse and G is the source or drain relatedby only few entries. Find M by multiple rhs iterative solving (fast!). We have $M=\Gamma G^{+}$. Next note that $(G^{+})^{-1}$ $A=M$ or $KA = M$ which is again a sparse problem with multiple rhs (fast!). Note G is not explicitly needed.

4. GUI design and implementation

The set up of (any) simulation with the MAGWEL solver, start from a main window, that in the widget view has four blocks (header, physics, solver, analysis).

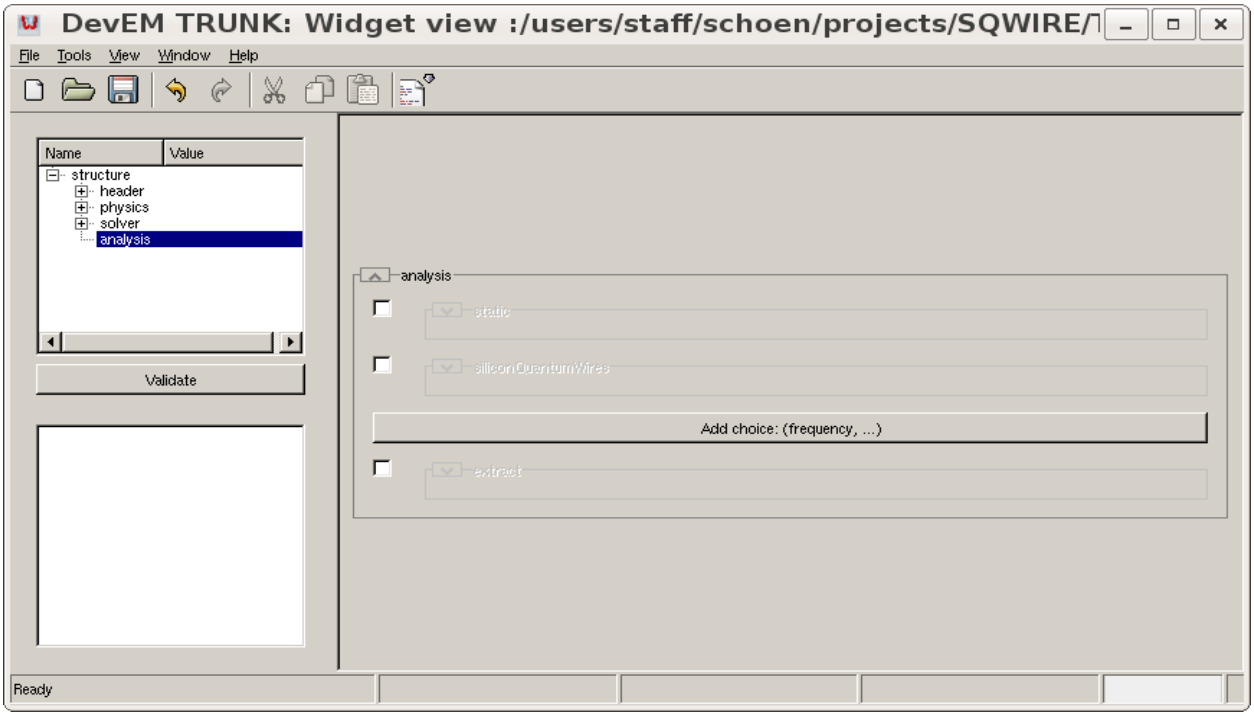


Fig. 8 Main window of the magwel solver

Fig. 8 shows the main window of the widget view. The widget view is one of the four presentations that are available for simulation set up. Besides the widget view, we have the 2D structure lay out interface, the 3D structure viewer (needed to check the structure connectivity and contact positioning) and the the XML editor, that allows for ascii text based input file generation.
Within the "analysis" part of the widget view, a specific entry is designed for launching quantum-transport simulations. The entry is shown in Fig. 9. A drop down menu can be activated for selecting wafer and transport orientations as is shown in Fig. 10. Moreover, one must select from the analytic 2d Schrödinger solution methods (that were developed at IMEC) or the NEGF approach with modifications

as designed by Tyndall. This is illustrated in Fig. 11. Assuming that we select the NEGF method, we must provide a few parameters that control the calculations. For example, the number of modes must be given. The entry box in shown in Fig. 12. In a similar way, if one selects the analytic approach, a corresponding menu box is opened that needs to be filled with parameters needed in the analytic calculations. This is illustrated in Fig. 13.

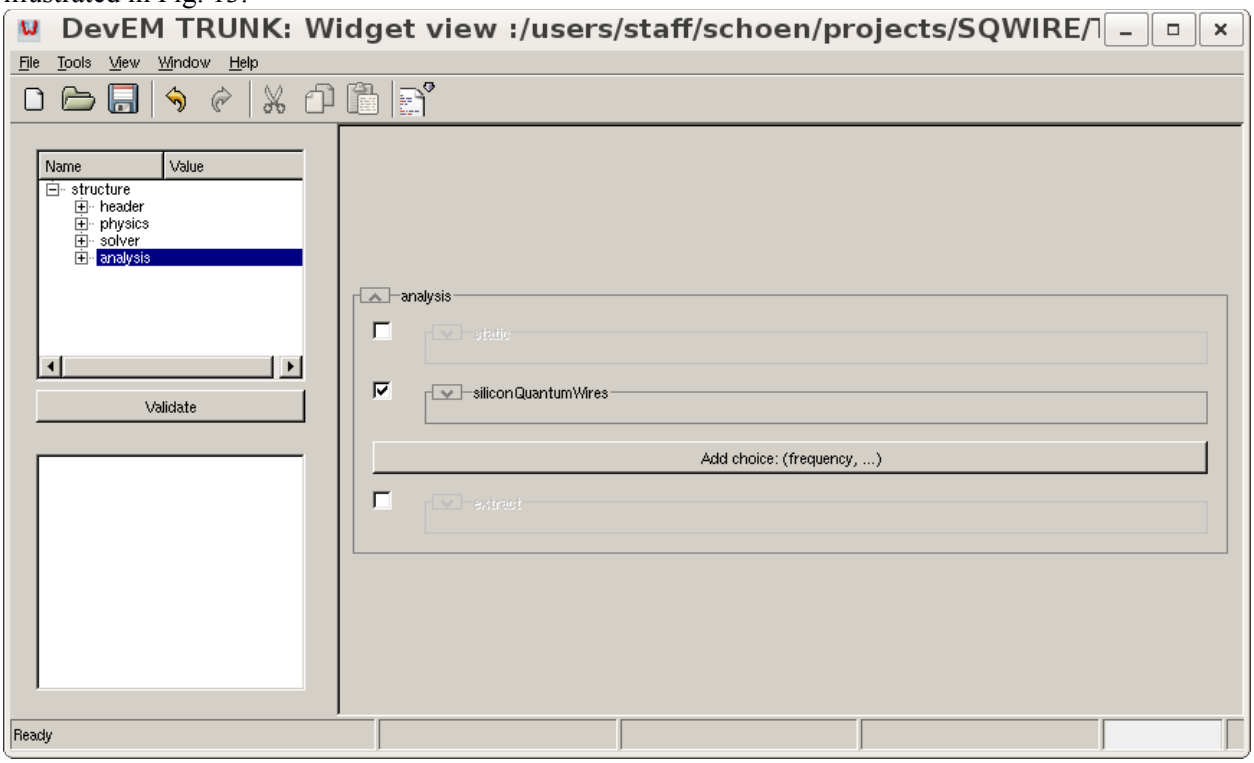


Fig 9. The analysis block allow the user to activate the analysis of quantum wires

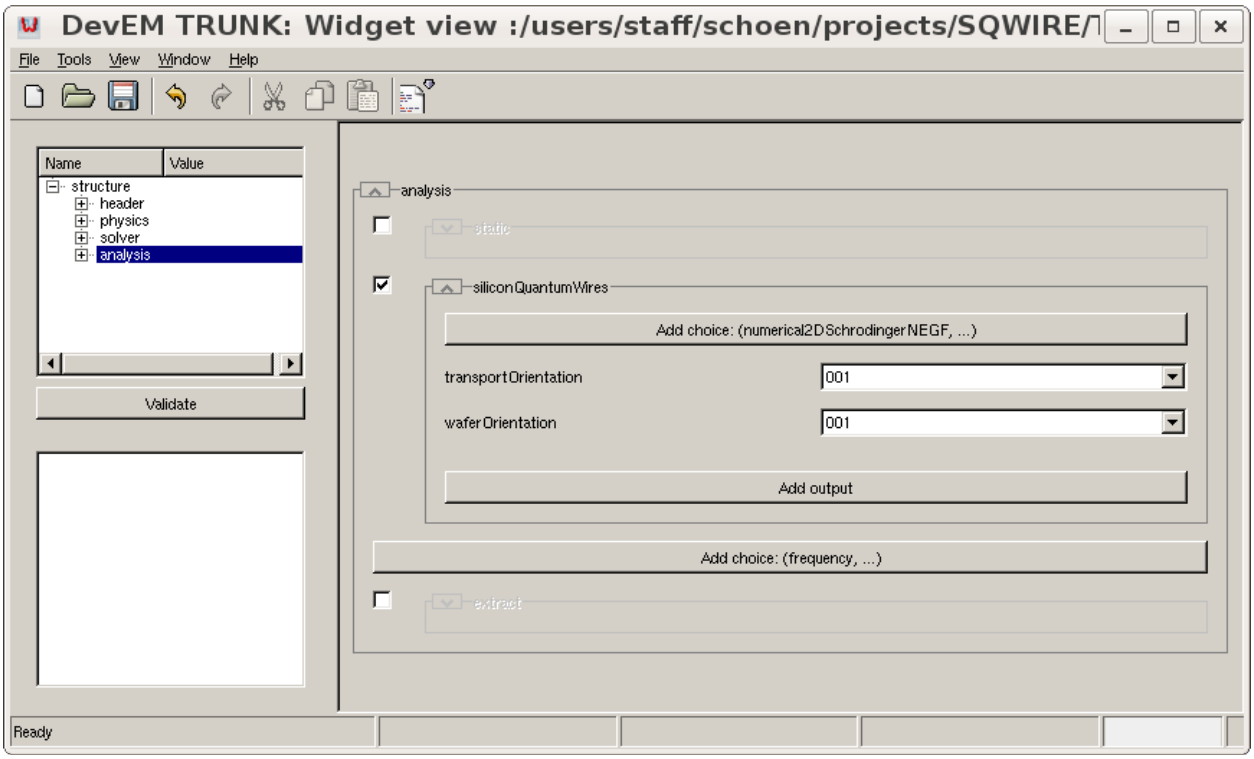


Fig. 10 Clicking the quantum wire analysis GUI part

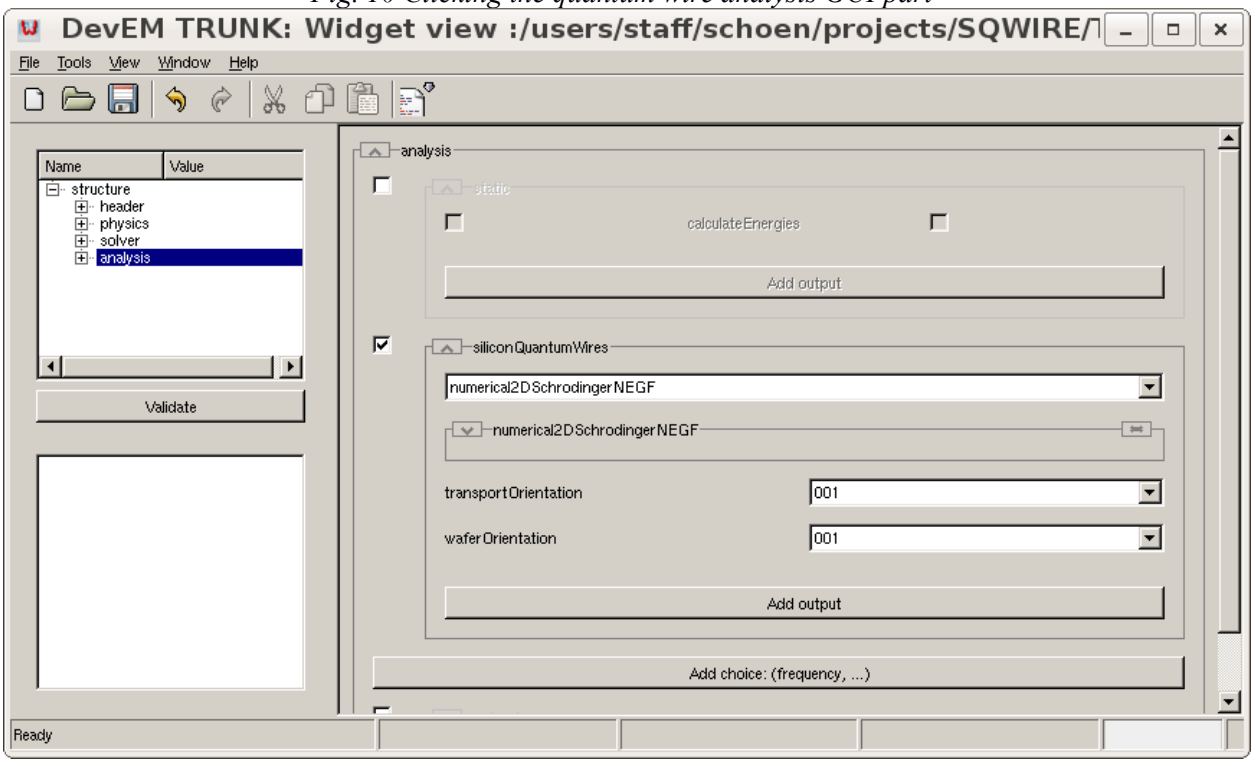


Fig. 11 Choosing between the NEGF or the analytic methods

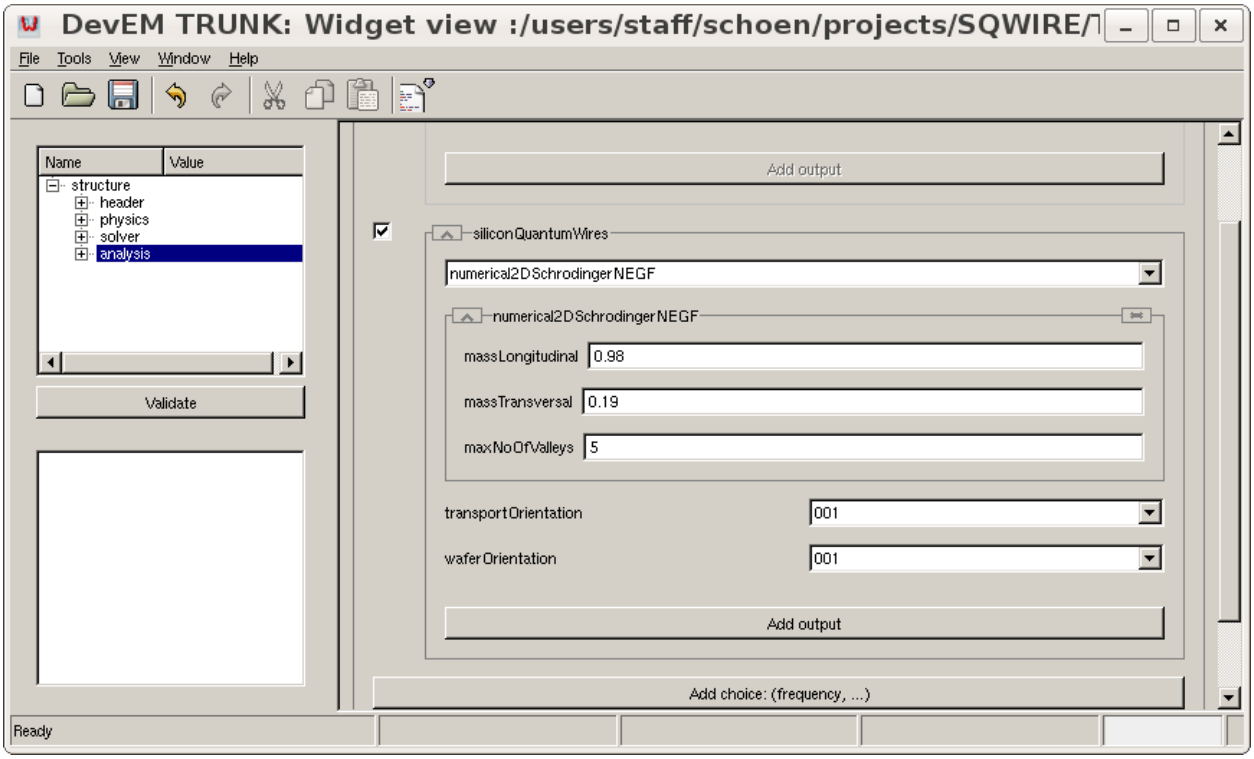


Fig. 12 Input parameters needed for the NEGF method

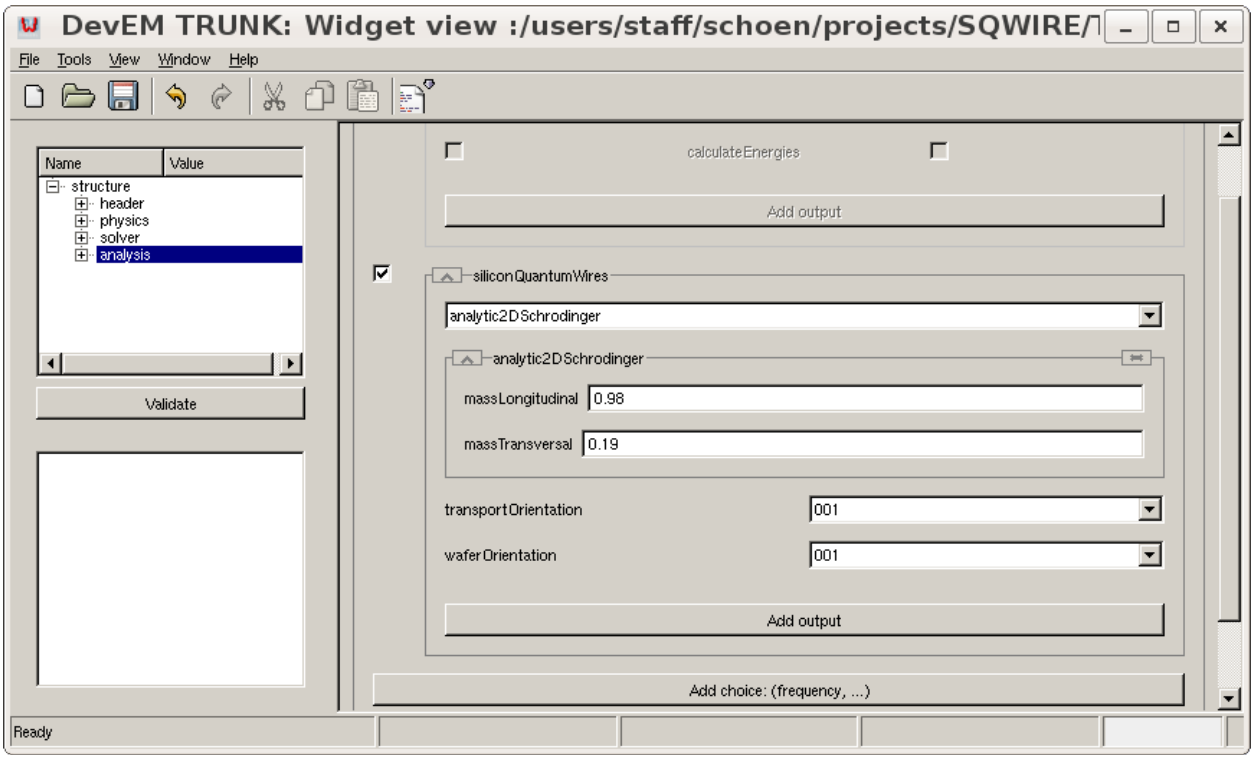


Fig. 13 Input parameters for the analytic quantum solver.

4.1. Solver menu options

Once that the analysis options have been determined, the user must provide some information about how the solution should be obtained. Then one must enter the "solver" section of the widget view. The main entry is shown in Fig. 14. It consists of two main blocks. The first block deals with meshing and in particular which kind of meshing strategy should be followed. The other main block, named 'magwel' , provides the interface to the details of the linear algebra. In general the solution can only be found if a set of criteria for convergence are respected. Each equation is written as "LHS=0" i.e. left-hand side is zero. In practice the "zero" is a small number determined by the machine accuracy. A specific segment is added to the semiconductor transport equations, where we can now select for a quantum-transport treatment. This is illustrated in Fig. 15. The different methods such the coupled-mode space or uncoupled mode space approach can be activated (Fig. 16). Also for the analytic methods (IMEC) one may select for specific details (Fig. 17).

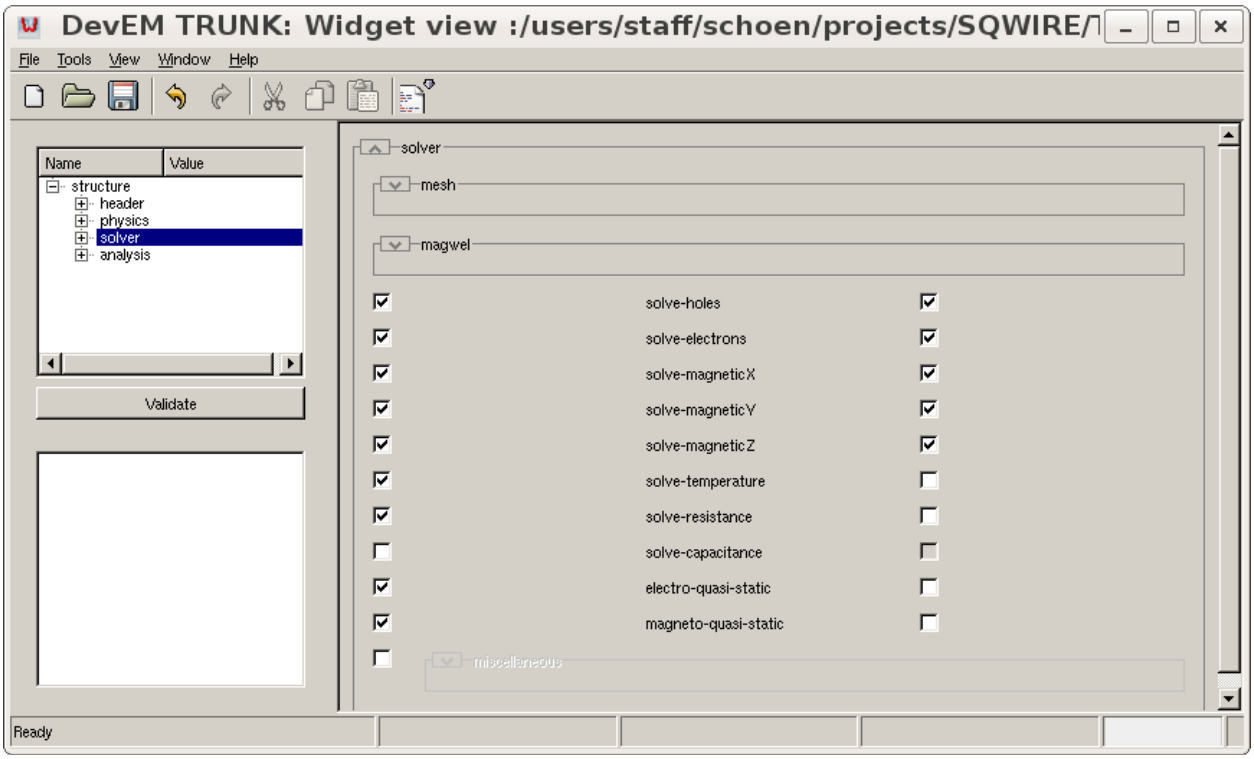


Fig . 14 Main solver setting window

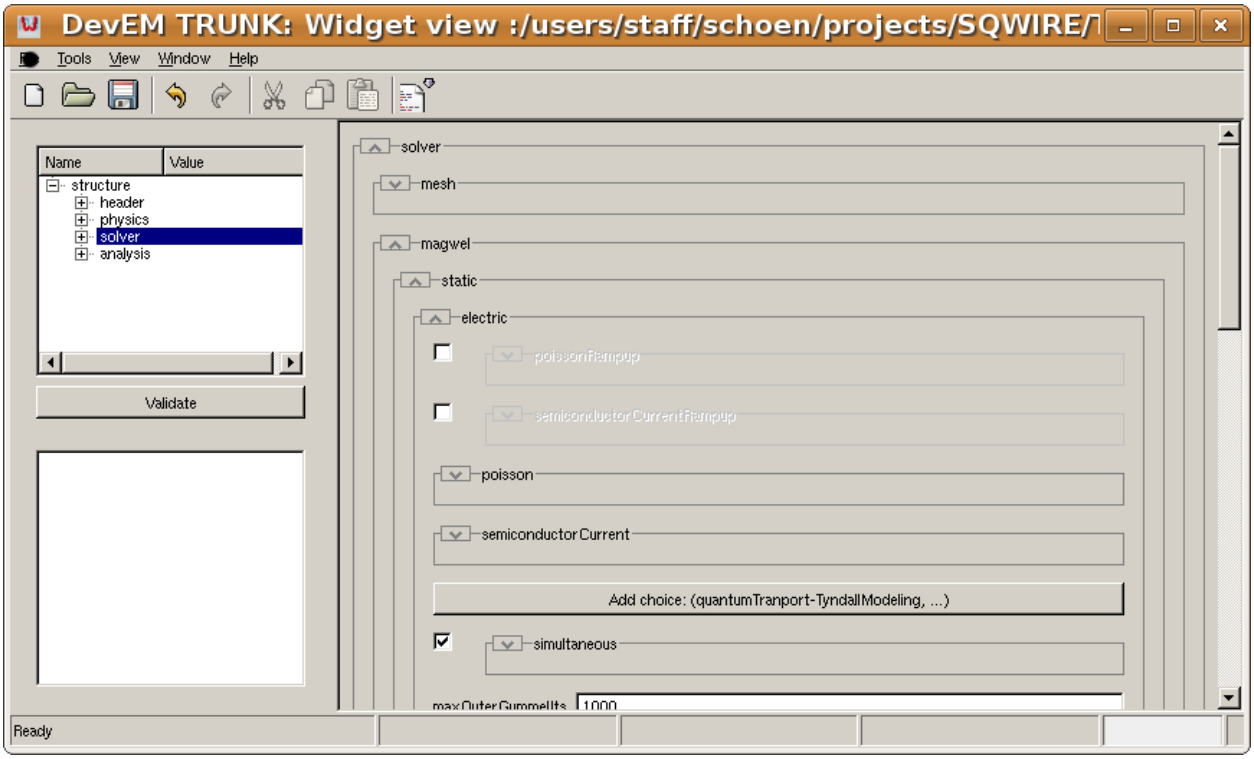


Fig . 15 Entering the solution part (i.e. not the mesh part) of the solver GUI

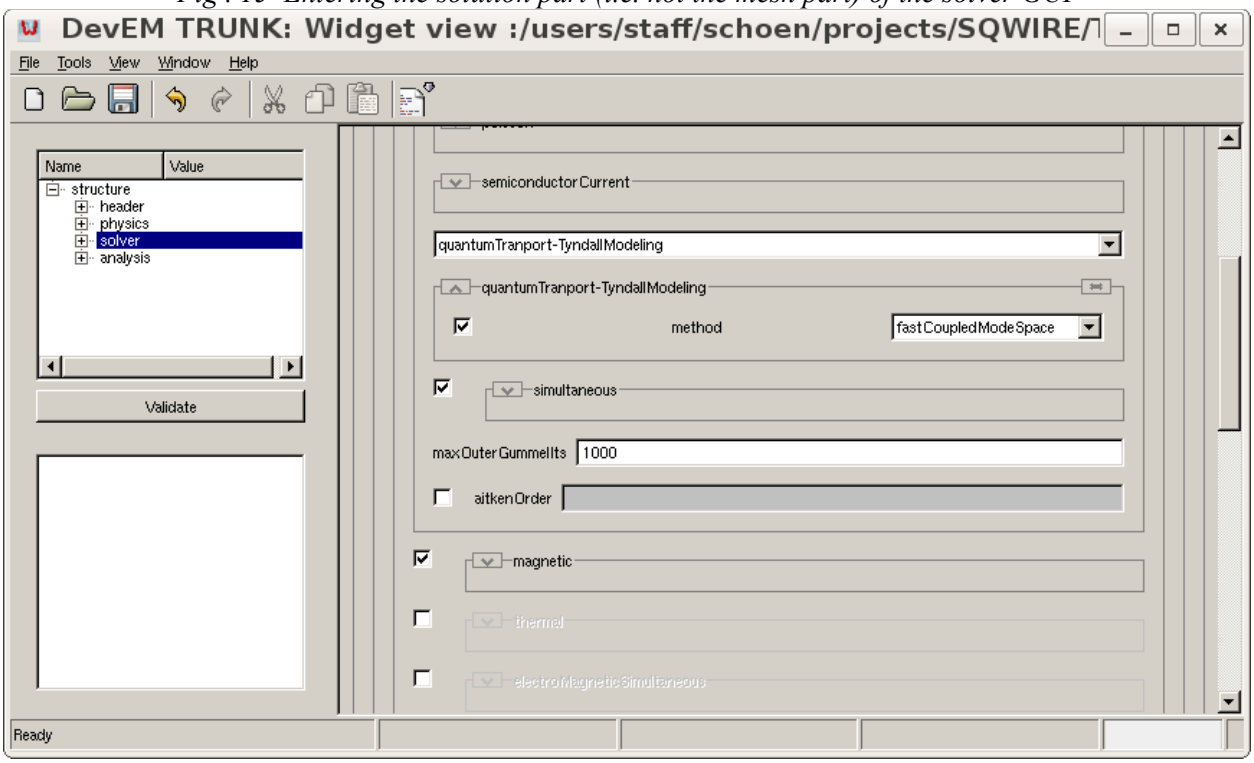
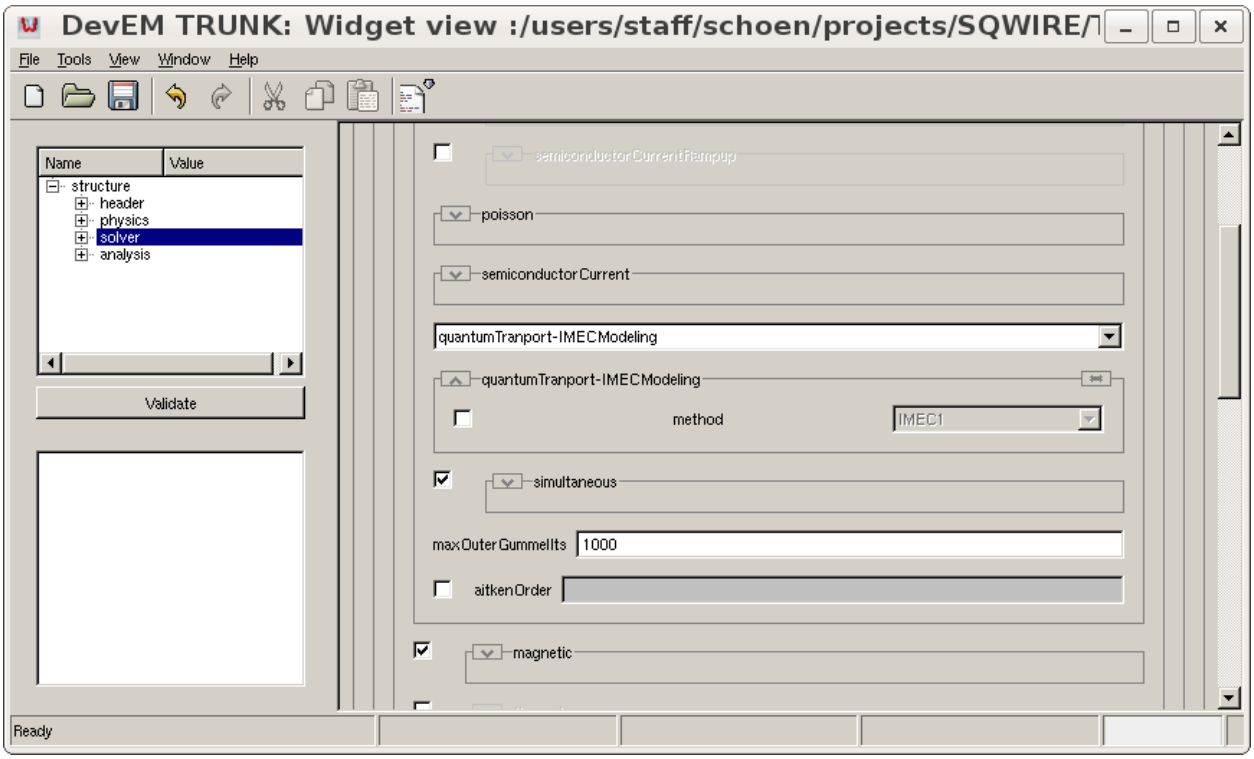


Fig. 16 In the section semiconductor current there is now a dedicated quantum transport option



Fig, 17 The analytic models of IMEC are accessible via a drop down selection menu.

4.2. Structure design – wire lay out

The Guiding principle is to set up of the computation in 2+1 dimensions. Therefore the wire is cut in 2D slices. Next the slices are glued together in a 1 D string.

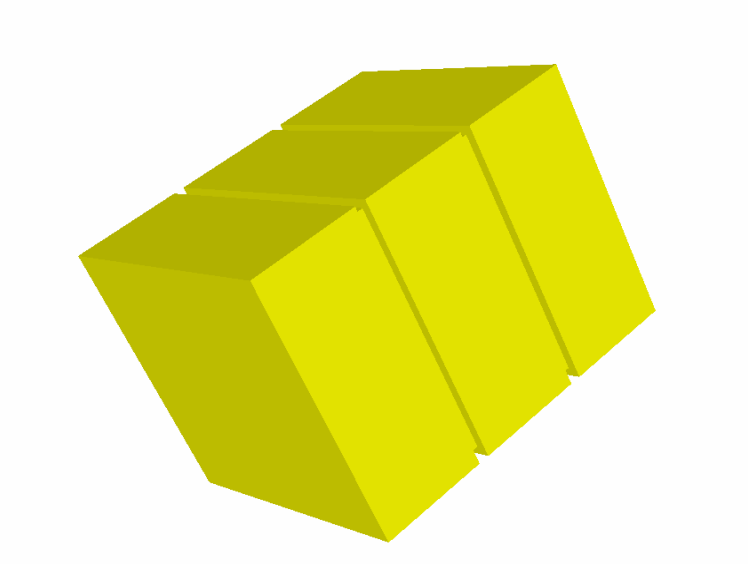


Fig. 18 layout of the tower of slices

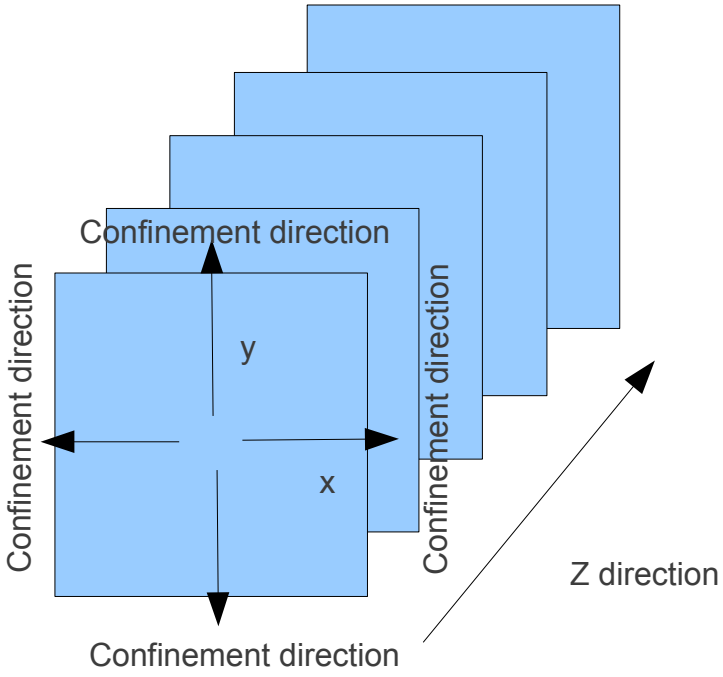


Fig. 19 Glueing together the 2D planes.

In each slice a 2D Schrödinger problem needs to be solved. In the MAGWEL solver arbitrary 2D shapes and meshing strategies can be selected.

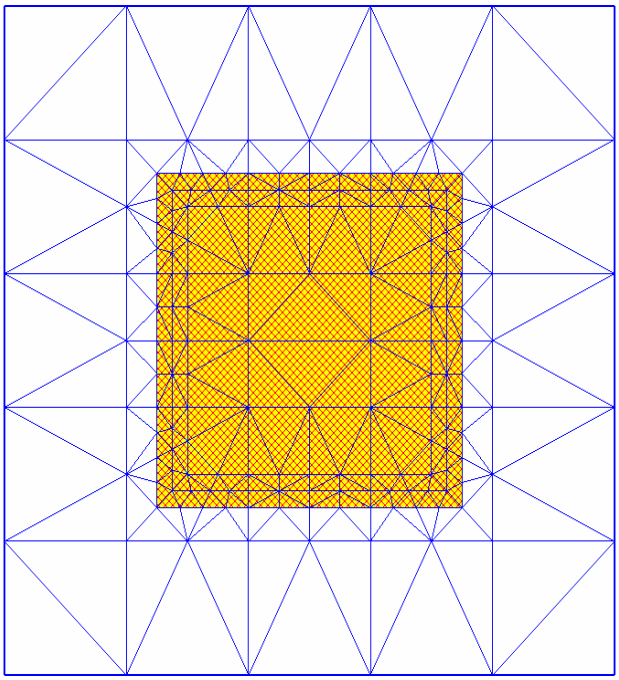


Fig. 20 Unstructured meshing odthe slices

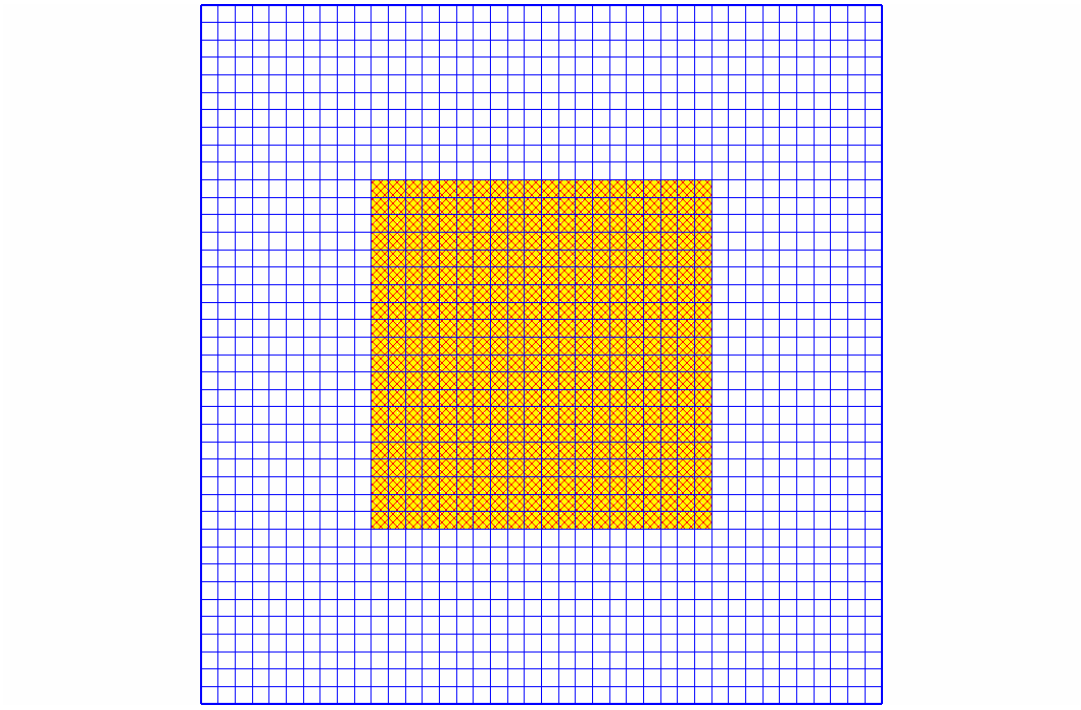


Fig. 21 Structured meshing of the slices

The 2D drawing is very 'general' as is shown in Fig. 22.

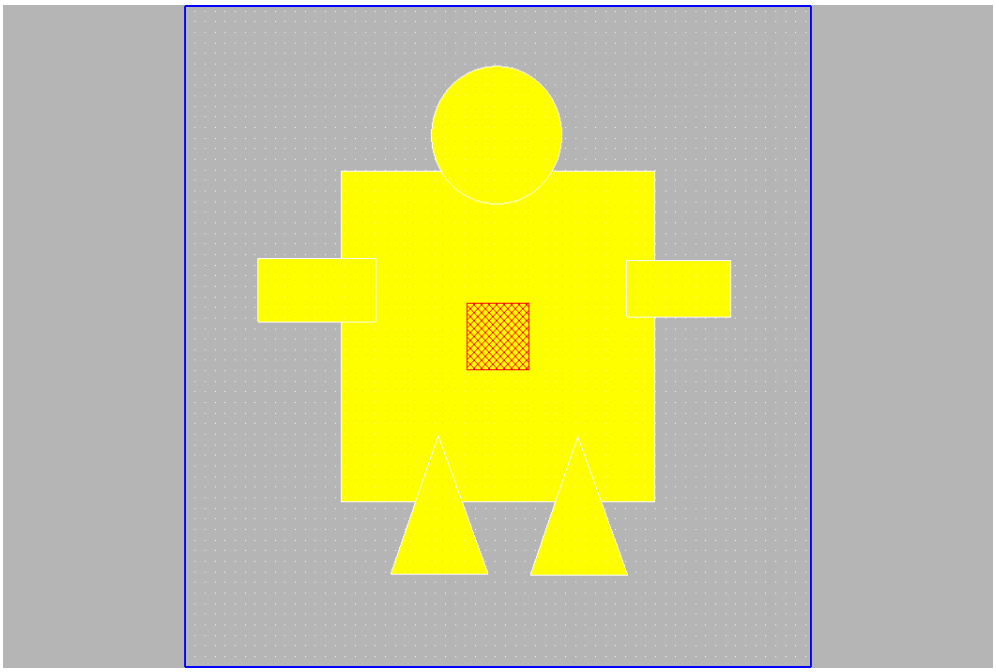


Fig. 22 Generic slice lay out drawing facility..

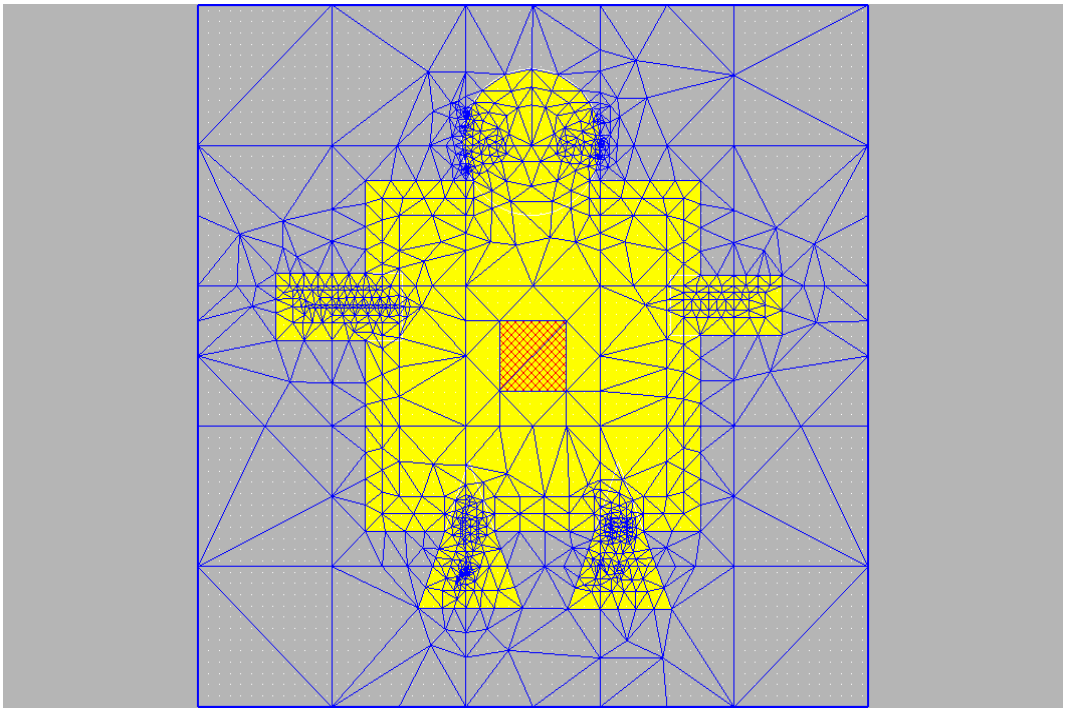


Fig. 23 The meshing of the generic slice lay out.

5. Conclusions and recommendations

In this document we described the implementation details of a NEGF solver starting from an existing 3D device simulator. We have described in detail the steps that are not extensively elaborated on in the existing literature [1,2]. In particular, we have generalized the approach to allow for non-equidistant meshing of the longitudinal direction. Moreover, in the transverse direction several meshing algorithms are available. The solving of the 2D Schrödinger equation in each slice exploits fast iterative solution techniques based on the Jacobi-Davidson algorithm. The MAGWEL solver has a dedicated sector dealing with 2D and 3D viewing of field variables. This sector has been extended to view the ground and first two excited states of the Schrödinger equations. The interface to fields (wave functions) viewing has been constructed.

6. References

[1] J. Wang, E. Polizzi, M. Lundstrom “A three-dimensional quantum simulation of silicon nanowire transistors with the effective-mass approximation” *J. App.Phys. 96, 2192, 2004*
[2] Supriyo Datta “Electronic Transport in Mesoscopic Systems” Cambridge University Press 1995 ISBN 0 521 41604-3.
see also: Supriyo Datta “Nanoscale device modeling; The Green function method” Superlattices and Microstructures Vol. 28, 2000, 253-278
[3] Mikhail Shashkov “Mimetic Finite Difference Methods for Partial Differential Equations and Discrete Vector and Tensor Analysis” T-7, Los Alamos National Laboratory shashkov@lanl.gov webpage: cnls.lanl.gov/ shashkov
[4] Gerard L.G. Sleijpen and Henk A. van der Vorst “ A Jacobi-Davidson iteration method for linear eigenvalue problems” SIAM J. Matrix Anal. Appl. (SIMAX), 17:2 (1996), pp. 401-425.
[5] David S. Burnett “Finite Element Analysis: from concepts to applications” Addison-Wesley Publishing Company” 1987 ISBN 0-201-10806-2
[6] R. Piessens, E. de Doncker-Kapenga, V. Ueberhuber, D. Kahaner “QUADPACK” A subroutine package for Automatic Integration – 1983 Springer-Verlag.
[7] M. Ali pourghadi, Wim Magnus, Bart Soree, Marc Meuris, Marc Heyns, Kristin De Meyer “Study of the Junction Depth on Ballistic Current Using the Subband Decomposition Method” SISPAD 2007 Ed. T. Grasser and S. Selberherr, Springer-Verlag.
[8]A. Afzalian, N.D. Akhavan, C-W Lee, R. Yan, I. Ferain, P. Razavi, J-P. Colinge “ A new FastCoupledModeSpace NEGF algorithm for efficient 3D Simulations of switching characteristics enhancement in constricted tunnel barrier silicon nanowire MuGFETs.” J. Comp Electronics. DOI 10, 1007/s10825-009-0283-1 (2009)

7. Appendix

The schema file part that generates the widget views of chapter 4 is presented below. It represents the design of the GUI based on the document object modeling (DOM).

```
<xs:complexType name="tyndallType">
        <xs:element name="method" default="fastUncoupledModeSpace" minOccurs="0">
                <xs:annotation>
                        <xs:documentation>what method to use, default FUMS</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                        <xs:sequence>
                           <xs:element name="method" default="fastUncoupledModeSpace"
                        minOccurs="0">
                                <xs:annotation>
                                  <xs:documentation>what method to use, default
                        FUMS</xs:documentation>
                                </xs:annotation>
                                <xs:simpleType>
```

```
                                <xs:restriction base="nonEmptyString">
                                        <xs:enumeration value="uncoupledModeSpace" />
                                        <xs:enumeration value="coupledModeSpace" />
                                        <xs:enumeration value="fastUncoupledModeSpace" />
                                        <xs:enumeration value="fastCoupledModeSpace" />
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
    </xs:complexType>


    <xs:complexType name="imecType">
        <xs:sequence>
            <xs:element name="method" default="IMEC1" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>what method to use, default IMEC1
                </xs:documentation>
                </xs:annotation>
                <xs:simpleType>
                    <xs:restriction base="nonEmptyString">
                        <xs:enumeration value="imec1" />
                        <xs:enumeration value="imec2" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>


    <xs:complexType name="TyndallAnalysisType">
        <xs:sequence>
            <xs:element name="massLongitudinal" default="0.98" minOccurs="1">
                <xs:annotation>
                    <xs:documentation>The longitudinal carrier mass</xs:documentation>
                </xs:annotation>
                <xs:simpleType>
                    <xs:restriction base="xs:double">
                        <xs:minInclusive value="0.0001" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="massTransversal" default="0.19" minOccurs="1">
                <xs:annotation>
                    <xs:documentation>The longitudinal carrier mass</xs:documentation>
                </xs:annotation>
                <xs:simpleType>
```

```
                    <xs:restriction base="xs:double">
                        <xs:minInclusive value="0.0001" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>

            <xs:element name="massOxide" default="0.55" minOccurs="1">
              <xs:annotation>
                <xs:documentation>The electron mass in Oxide</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:double">
                    <xs:minInclusive value="0.0001" />
                </xs:restriction>
              </xs:simpleType>
            </xs:element>

            <xs:element name="semiconductorInsulatorBarrier" default="3.0" minOccurs="1">
              <xs:annotation>
                <xs:documentation>Si/SiO2 potential barrier. Default 3 Volt</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:double">
                    <xs:minInclusive value="0.0001" />
                </xs:restriction>
              </xs:simpleType>
            </xs:element>

            <xs:element name="maxNoOfValleys" default="5" minOccurs="1">
                    <xs:annotation>
                        <xs:documentation>The maximum number of valleys to be
                    used</xs:documentation>
                    </xs:annotation>
                    <xs:simpleType>
                        <xs:restriction base="xs:integer">
                            <xs:minInclusive value="1" />
                            <xs:maxInclusive value="100" />
                        </xs:restriction>
                    </xs:simpleType>
            </xs:element>

            <xs:element name="maxNoOfSubbands" default="0" minOccurs="1">
              <xs:annotation>
                <xs:documentation>The maximum number of energy subbands to be used
                    (default 0: then the program calculates the number of
            subbands)</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:integer">
```

```
                    <xs:minInclusive value="1" />
                    <xs:maxInclusive value="100" />
                  </xs:restriction>
                 </xs:simpleType>
                </xs:element>

                <xs:element name="energyVariation" default="fixed" minOccurs="1">
                 <xs:annotation>
                  <xs:documentation>fixed or adaptive energy stepping strategy
                      (default fixed)</xs:documentation>
                 </xs:annotation>
                 <xs:simpleType>
                  <xs:restriction base="nonEmptyString">
                    <xs:enumeration value="fixed" />
                    <xs:enumeration value="adaptive"/>
                  </xs:restriction>
                 </xs:simpleType>
                </xs:element>

        </xs:sequence>
</xs:complexType>

<xs:complexType name="ImecAnalysisType">
        <xs:sequence>
                <xs:element name="massLongitudinal" default="0.98" minOccurs="1">
                        <xs:annotation>
                                <xs:documentation>The longitudinal carrier mass</xs:documentation>
                        </xs:annotation>
                        <xs:simpleType>
                                <xs:restriction base="xs:double">
                                        <xs:minInclusive value="0.0001" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
                <xs:element name="massTransversal" default="0.19" minOccurs="1">
                        <xs:annotation>
                                <xs:documentation>The longitudinal carrier mass</xs:documentation>
                        </xs:annotation>
                        <xs:simpleType>
                                <xs:restriction base="xs:double">
                                        <xs:minInclusive value="0.0001" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
        </xs:sequence>
</xs:complexType>

        <xs:element name="siliconQuantumWires" minOccurs="0">
         <xs:annotation>
```

```
        <xs:documentation>DC analysis using Quantum wire modeling</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>

          <xs:element name="transportOrientation" default="001" minOccurs="1">
            <xs:annotation>
              <xs:documentation>orientation of transport direction wrt silicon lattice
               default: 001</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="nonEmptyString">
                <xs:enumeration value="001" />
                <xs:enumeration value="111" />
                <xs:enumeration value="110" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>

          <xs:element name="waferOrientation" default="001" minOccurs="1">
            <xs:annotation>
              <xs:documentation>orientation of the wafer </xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="nonEmptyString">
                <xs:enumeration value="001" />
                <xs:enumeration value="111" />
                <xs:enumeration value="110" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>

          <xs:choice id="devemQTM" >
    <xs:element name="numerical2DSchrodingerNEGF" type="TyndallAnalysisType"
    minOccurs="0">
          <xs:annotation>
            <xs:documentation>using numerical solutions in transverse planes and Datta
          theory for transport direction (Tyndall) </xs:documentation>
          </xs:annotation>
    </xs:element>
    <xs:element name="analytic2DSchrodinger" type="ImecAnalysisType"  minOccurs="0">
          <xs:annotation>
            <xs:documentation> using analytic base functions in transverse planes and LDA
              for transport direction (IMEC) </xs:documentation>
          </xs:annotation>
    </xs:element>
</xs:choice>

<xs:sequence>
```

```
        <xs:element name="output" type="staticPlotTypeQTM" minOccurs="0"
        maxOccurs="unbounded" />
</xs:sequence>
</xs:sequence>
</xs:complexType>
</xs:element>
```