February 2013. Vol. 2, No.2 ISSN 2305-8269

International Journal of Engineering and Applied Sciences

www.eaas-journal.org

# SIMULPY: AN OPEN-SOURCE AND FREE COMPUTATIONAL TOOL FOR DESIGN-ORIENTED MODELLING AND SIMULATION OF MOS TRANSISTORS.

**Lester de Abreu Faria[1]; Marcio Scarpim de Souza[2]; Roberto d'Amore[1]**

[1]Technological Institute of Aeronautics - ITA – Electronic Department - Brazil
[2]Brazilian Army Technological Center – CTEx – Optronic and Sensors Laboratory –Brazil

Abstract:

An open-source and free computational tool for Design-Oriented Modeling and Simulation of MOS transistors is presented. Focusing on a friendly user's interface, the software is implemented in PYTHON language and uses EKV2.6 compact transistor model, providing direct access to small sets of equations and parameters. Such features make this software ideal for transistor-level design purposes, enabling the designer to manipulate open-source equations and models, while achieving a good conformance and a continuous behavior of the transistor in all regions.

Several options of output and transfer characteristics are implemented, allowing the designer not only to examine the parametrical behavior of the transistor but also the charge behavior of each one of its nodes.

Besides providing simulations of the respective characteristics under various conditions, different "Design-Oriented" functionalities are implemented, such as "import and comparison" with a pre-established measurement file (*.txt), calculus of the relative difference between them and graphical analysis and comparison. This last feature provides high user's interaction, facilitating the perception of the transistor behavior. The simulated results are automatically saved in an *.xlsx file (Excel).

Validation against commercial simulation software LTSPICE, for AMS 0.5 µm CMOS technology, is performed and errors less than 1.2% are observed.

Key-words: Open-Source softwares; PYTHON Language; EKV Compact Model; Design-Oriented Modeling and Simulation

## I. INTRODUCTION

The availability of accurate, flexible and efficient simulation tools is critical to the successful design of any circuit, showing to be the most important phase during the design of integrated circuits [1]. Especially for analog or mixed-signal circuits, the circuit analysis must be highly accurate while simulator device models must satisfy several criteria, becoming an even more important issue for the efficient design and simulation [1] [2].

In this context, many of the necessary calculations and simulations are of a repetitive and general nature, suitable for implementation of a generic computational tool. The availability of a well-designed computational tool facilitates and increases productivity during subsequent tool and devices development and, consequently, their upgrade, where new modules can be added to an ever-increasing set of dedicated tools. The concept of an extendible toolkit leads naturally to the free and open-source philosophy, where the tool is developed cooperatively, for users' mutual benefit [3]. In broader terms of the open-source philosophy, some desirable features must be pursued, as some of the previously stated by Eric Raymond [4]: (1) Rule of Modularity: The program must be written in independent and simple modules, which must be linked through understandable interfaces. (2) Rule of Clarity: It is better to develop a program that is simple and understandable for the reader than one that is more advanced but complex and confuse. (3) Rule of Composition: The program must be designed considering the possibility of connecting it to other existing programs. (4) Rule of Simplicity: The program must be as simple as possible to meet your goals. (5) Rule of Parsimony: Programs must be as short as possible. Big programs must be written only when it is really unavoidable. (6) Rule of Transparency: The program must be designed in order to provide enough clearness to facilitate eventual inspection and debugging. (7) Rule of Robustness: The simplicity and transparency of the program must result in its ability to keep operating despite of abnormalities. (7) Rule of Optimization: "The best is the enemy of the good" (Voltaire). Before optimizing the program you must be sure that it works

International Journal of Engineering and Applied Sciences

www.eaas-journal.org

consistently. and (8) Rule of Extensibility: Keep in mind that your program must be written focusing on future upgrades.

Concerning to these requirements, the availability, combined with a high speed computational capability of software, is essential to the simulation and optimization processes. In this context, PYTHON is an ideal candidate, once besides being free, it provides open-source code, providing direct access to model equations and parameters, and a high capability of calculations and plotting [5] [6] [7] [8]. Hence, parameters can be easily observed and compared; and the transistor characteristics optimized for the circuit.

On the other hand, EKV Model is gaining recognition as a public-domain model and a growing number of designers are using it in the academic research. The EKV model is built on physical properties of the MOS structure, showing to be scalable and compact [9] [10] [11] [12]. It is mainly dedicated to the design and simulation of low-voltage, low-current analog, and mixed analog-digital circuits using submicron CMOS technologies. Besides that, the model is formulated as a "single expression", which preserves continuity of first- and higher-order derivatives with respect to any terminal voltage, in the entire range of validity of the model.

It is clear that the combined circuit and device optimized simulation is capable of leading to real benefits in both device design and circuit use. The SimulPy open-source and free computational tool is being developed following these previous shown features. It is being written based on basic functions of the PYTHON language, highlighting the rules of Modularity and Extensibility, once it is desirable that it serves as a functional block to subsequent programs. Documentation, tests, graphs and user interfaces are being developed and provided as well, focusing on calculations for the semiconductor simulation domain. The main objective is to simulate, based on a small set of semiconductor parameters (EKV2.6 libraries), the transistors behavior in any situation, with high reliability and accuracy.

It can be found some other kind of Simulating Softwares in the market, as PSPICE, ELDO, LTSPICE, [13] [14] [15], etc., some of them, although being free, are not open-source. Therefore, a computational tool that is able to be changed (open-source) according to the necessities of the user allows the development and improvement of the existing models and allows, as well, the correct understanding of the methodology followed in the device characterization and simulation.

Beside all of these advantages, it will run with a low computational effort and completely free of financial costs, being available for all the research community. It shows also high graphical interactivity with the user. While several toolkits for Matlab and Python exist in other scientific domains, there was not found a toolkit readily available in this domain, with the same functionalities.

The purpose of this paper is to present the first prototype of SimulPy and some promising results in specific simulations. This paper covers some general software requirements, the language selection considerations, some users' interface screens, a practical example of a simulation and its validation to other commercial simulation software LTSPICE, in AMS 0.5um technology. Finally, some concluding remarks and future works proposal are presented.

## II. GENERAL SOFTWARE REQUIREMENTS

Currently it can be found a great variety of simulation tools and transistor libraries in the market. Each one of them shows strengths and weaknesses, according to the proposed usage and the target application. Even though, in spite accurate modeling and software structure, these softwares and simulation models may still be user's non-friendly or be quite far from the real silicon performance.

A typical Simulation tool requirement (in a simplified vision) is the modeling of the behavior of the component in all regions of operation. SimulPy is a computational tool that uses the EKV2.6 model as a standard, including a series of other specific functions focusing on the "Design-oriented facilities", such as "importing and comparison" a pre-established measurement file (*.txt), calculus of the relative difference between them and graphical analysis and comparison. This last feature provides high user's interaction, facilitating the perception of the transistor behavior. The simulated results are then automatically saved in an *.xlsx file (Excel). Besides all these functions, it was implemented a possibility of tracing graphs with axis that are independent of the primary and secondary variation parameters. Therefore, it is possible, for example, trace a graph of Cgs (Gate-Source total intrinsic capacitance) x IC (Inversion coefficient), varying the VG (gate voltage), as primary parameter, and VB, as secondary parameter. Besides complex graphs like this, which other similar softwares are not able to trace, SimulPy can trace all the other parametric ones, generally used for component characterization.

The used EKV2.6 MOSFET model is a scalable and charge-based compact simulation model built on fundamental physical properties of the MOS structure. This model is dedicated to the design and simulation of low-voltage and low-current analog circuits. [9] [10] [11] [12]. It is formulated as a "single expression", which preserves continuity of first and higher-order derivatives with respect to any terminal voltage, in the entire range of validity of the model. More information about the model, including parameters and equations, as well as its main characteristics, are out of the scope of this paper, and can be found in [9] . It is able to describe most of the effects on charges, trans-capacitances, drain current and tranconductances in weak, moderate and strong inversion regions of the MOSFET operation, as well as the drain current from conduction to saturation. These effects are modeled in a hierarchical structure that allows the implemented computer

International Journal of Engineering and Applied Sciences

www.eaas-journal.org

simulations, through the use of a concise parameter set (18 intrinsic parameters): 15 physical parameters (COX, VTO, GAMMA, PHI, KP, E0, UCRIT, XJ, DW, DL, Q0, LK, IBA, IBB, IBN) and 3 second-order fitting coefficients (LAMBDA, WETA, LETA).

In addition to the SimulPy specific-domain requirements, it was also implemented some additional requirements in order to enable reading data files, interpolate data and visualize the processed results. These functionalities were implemented by the use of the following modules: SYMPY, NUMPY, CSV and MATPLOT, all of them being free and open-source, as well, and accessible for download in internet.

In order to compare the simulated results to previous ones (experimental or simulated), a common need is the reading of tables generated by other instruments and simulators, such as Semiconductor Parameter Analyzer – B1500 or ELDO and PSPICE, for example. In this case, a simple and easy to manipulate *.txt file can be imported, being read by SimulPy and serving as standard for future calculations and comparations (based on them it can be done the calculus of the difference of both data and the tracing of comparative graphs). Generally, all instruments and simulators are able to generate this kind of file, or another kind of file that can be transformed in *.txt (*.csv files, for example).

Data visualization is a very important element in the understanding and validation of the simulations of the component - errors are more readily recognised in graphical visualisation than in tabular data. The data visualization is presented in different formats of two-dimensional (x, y) data graph plots. These presentations are suited for a better interpretation and analysis of all the parameters and their fittings with the experimental behaviour. Two-dimensional data graphs include combinations of linear and log scale plots.

The user interface that was designed for SimulPy is the simplest one that could be implemented, in order to allow an easy understanding and a high interactivity with the user, not being computationally "heavy" or slowing the calculation process. This interface was provided through the use of decision tasks and questions, while showing graphs that allow the user to decide, based on the component behaviour. If there is not a decision to be made, all the calculus is done automatically, not requiring the user's interference.

Finally, at the end of the program, the graphs are traced and presented to the user, who can interact with it by zooming in or out, panning axis, stretching up-down or left-right, configuring sub-plots, editing curves lines and axes parameters or saving figure in a different file. The model codes are open source, providing direct access to the used model equations and parameters. Thus, parameters can be easily observed and analyzed and component characteristics optimized for circuit design purposes.

## III. LANGUAGE CONSIDERATIONS

There can be found a number of languages that provide built-in operations on scalars, vectors and arrays/matrices with single operators. Examples of such languages are Matlab [16] (or its open source equivalents SciLab7 and Octave8) and Python9 [7] together with Scipy and Numpy [5] [8]. Any one of these languages offers many benefits, among them: (1) A behavioral variable is easily modelled as a vector or a single column of an array or matrix. All vectors are easily converted, interpolated and used as inputs of the calculus in an easy way, and presented as a graph, allowing the user to take some decisions. (2) Vector or array variables can be loaded from *.txt files. (3) Graphical visualisation tools are available and easy to use, not requiring high computational efforts. and (4) An interactive environment supports very fast development time, as well as the capability to write script files for more complex problems.

The two main candidates for the SimulPy are Matlab and Python. For this kind of application, calculations and plotting, both softwares show practically the same performances., However, Python seems to be better than MatLab considering the structure of the language, since it does not present some of the flawed design decisions and work-around that MatLab does. [3].

Python has a large number of modules providing a considerable capability as a general purpose language, while Matlab has a large number of specialised and powerful scientific and engineering toolboxes. Matplotlib, a Python Data Visualization tool, shows stronger capability than the Matlab similar one [3]. Python, as well, is supported by Numpy and SciPy, if considering scientific processing tools. Finally, it is possible to highlight that Python is free and open source, while Matlab is a proprietary source and much more expensive, especially for the toolboxes [3].

After some months working with both softwares in the SimulPy development, the decision for Python was taken based on a big picture and on trade-off between functionalities, price, availability of support material in internet and easiness of understanding and programing. In recent years Python have been well tested, and have stabilised and matured sufficiently to support mainstream tool development.

Finally, all Python files have a common structure, being all of them (or the great majority) free and easily downloaded from internet. The same file that is imported by the user also contains the summary documentation and test/demonstration code.

## IV. USER'S INTERFACE CONSIDERATIONS

The simulation methodology established for EKV v2.6 model is well described in [9], but goes beyond the scope of this paper and, for convenience, will not be discussed in detail.

International Journal of Engineering and Applied Sciences

www.eaas-journal.org

For the purpose of this paper, it is sufficient to remember that simulation of components is based on sequential tasks and on a pre-defined set of parameters (EKV model) that are used as inputs in the form of *.txt files. After these data are read, all the necessary calculus is performed automatically.

First of all, it is important to define some software's areas (users' interface) that are presented to the user, as shown in Figure 1.

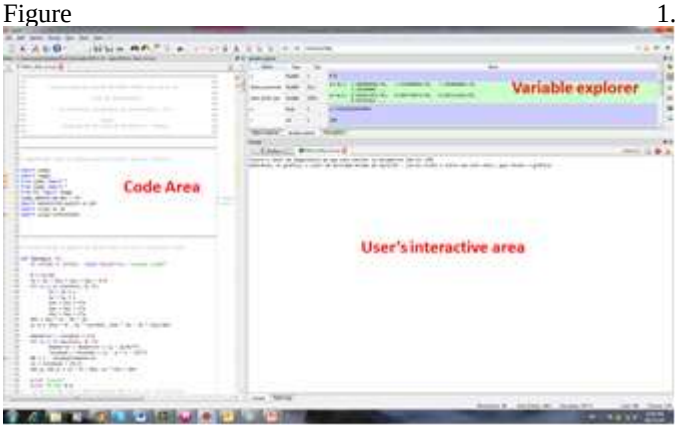



*Figure 1. PYTHON typical users' interface*

Figure 1 shows the three main zones of the PYTHON environment. As it can be seen, it looks like Matlab environment, where on the left there is a "Code Area", which embeds the code source; a "Variable explorer", where the variables of the program are presented with their values; and the "User's interactive Area", where the questions will be presented to the user, focusing on the input data and simulation parameters.

From now on, it is presented several users' interface screens in order to comprehend the functionalities of the software. Figure 2 shows the possibilities and definition of primary and secondary variation on the transistor terminals.

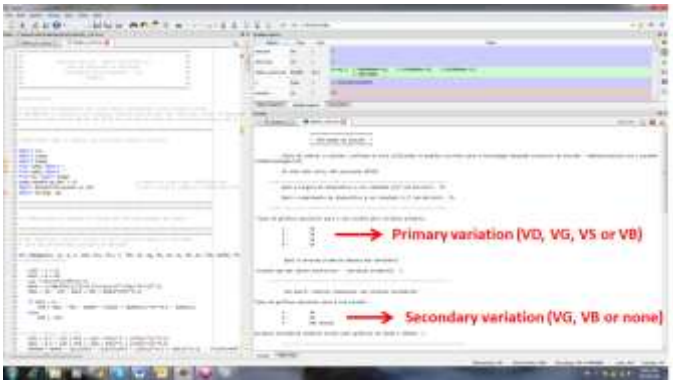



*Figure 2. SimulPy interface screen, considering the variations that can be implemented to the transistor's terminals*

After defining the variations that will be implemented to the transistors' terminals during simulations, it must be defined the axis of the graph that will be presented to the user. As can be seen in Figure 3, 17 options can be defined, for both ordinate and abscissa.

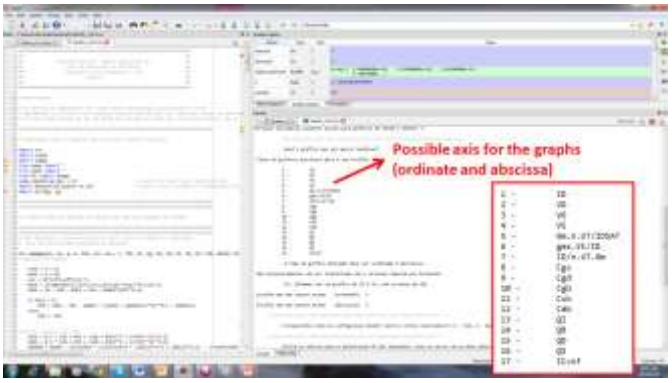



*Figure 3. SimulPy interface screen, considering the different possibilities for the graph axis*

The next step of the software is to define the bias of the terminals, as well as the range of the primary and secondary variations, pre-established. It can be decided, as well, at this point, if there is a file that must be imported and compared to the proposed simulation, what can be seen in Figure 4.

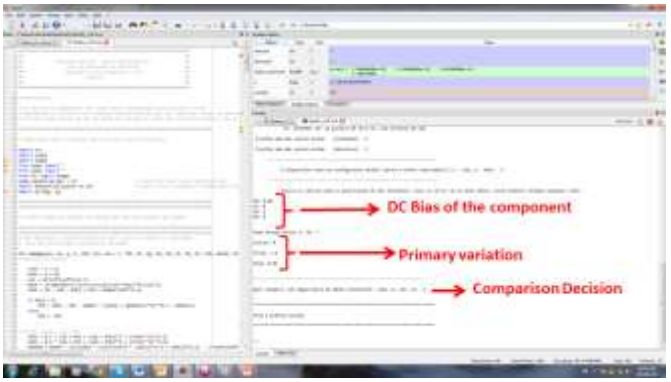



*Figure 4. SimulPy interface screen, considering biasing of the transistor and the possibility of comparison with an additional file*

And finally, the graph is presented to the user. In Figure 5, it can be seen several functionalities of the PYTHON – SimulPy graph, what makes it a graph with high interactivity possibilities.

International Journal of Engineering and Applied Sciences

www.eaas-journal.org

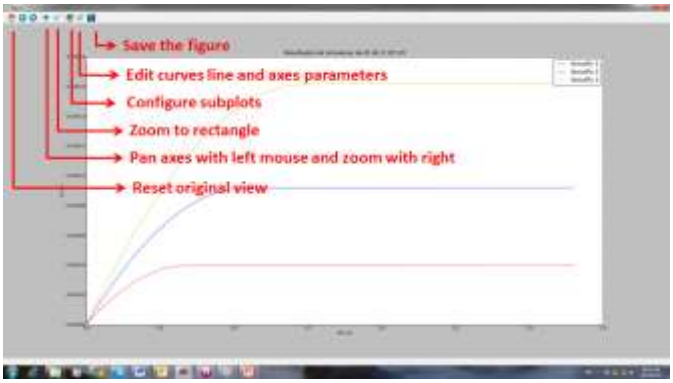
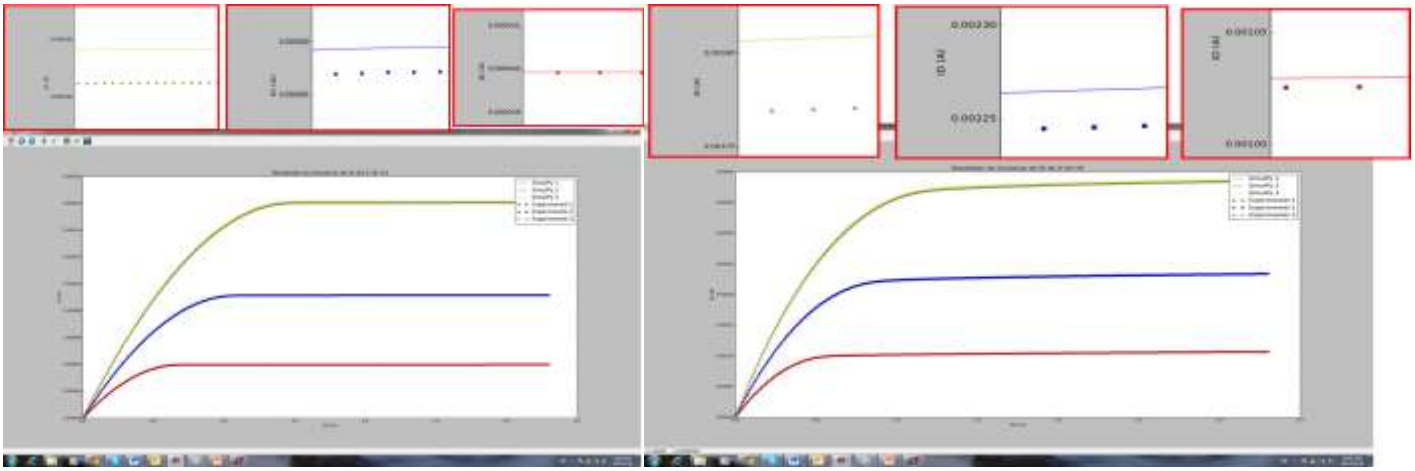



*Figure 5. SimulPy interface screen, showing the high graph interaction possibilities*

In order to validate the proposed software, it was chosen a commercial and free simulation software called LTSPICE [15], which is widely used in both academic research and in professionals' simulations. In order to maintain the consistency, the same EKV2.6 model from AMS 0.5um technology was used in both simulations (SimulPy and LTSPICE) and the comparison results were presented. Several graphs were simulated and the results can be seen below.

Figures 6 (a) and (b) show the comparison between the results from SimulPy and LTSPICE, for transistors of W/L=25um/25um and 25um/0.8um, which represent the behaviour of a long and wide transistor and a short channel transistor, respectively. The chosen parametric measurement in this characterization was the IDxVD curve, with VG=1.5 (in red), 2.0 (in blue) and 2.5 (in yellow).

## V. VALIDATION OF THE SOFTWARE



*(a)* *(b)*

*Figure 6. IDxVD results from SimulPy and LTSPICE, for transistors of (a) W/L=25um/25um and (b) 25um/0.8um*

In Figure 6(a) and (b) it is possible to see the a high level of conformance between the results of SimulPy and LTSPICE (Experimental). In these cases, errors of 0.487% and 0.725% were achieved considering the following equation

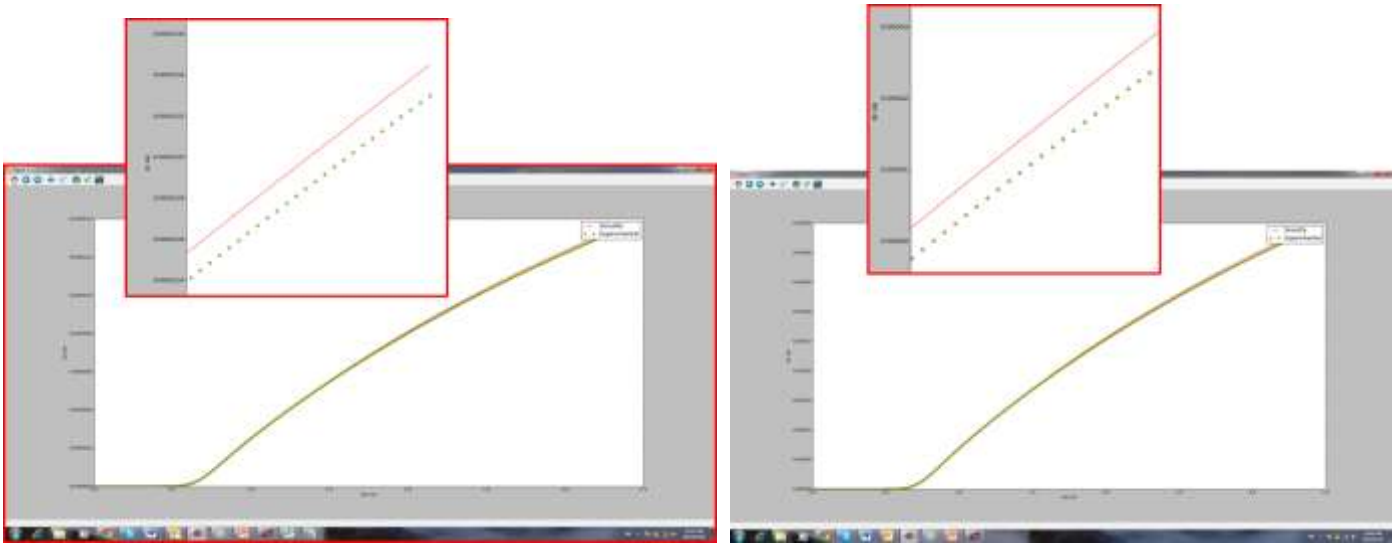
$$Error = \frac{\sum Relative\_errors}{Number\_of\_points} \quad (I)$$

It can be also seen in Figure 6(a) and (b) a zoom in each one of the lines, providing to dimension the high accuracy of the simulation software when compared to other commercial one.

As another example, Figure 7 (a) and (b) show the comparison between the results from SimulPy and LTSPICE, for transistors of W/L=25um/25um and 25um/0.8um, like the previous ones, but now the chosen parametric measurement in this characterization was the IDxVG curve, with VS=0V.

International Journal of Engineering and Applied Sciences

www.eaas-journal.org



*(a)* *(b)*

*Figure 7. IDxVG results from SimulPy and LTSPICE, for transistors of (a) W/L=25um/25um and (b) 25um/0.8um*

In Figure 7(a) and (b) it is possible to confirm the a high level of conformance between the results of SimulPy and LTSPICE (Experimental). In these cases errors were bigger, in the magnitude of 1.12% and 1.15%, but already been considered acceptable for simulation purposes.

It can be also seen in Figure 7(a) and (b) a zoom in each one of the lines, what was done in the final position of the VG axis (from 3.1 to 3.3V), which is the worst case, providing to note the high accuracy of the simulation software when compared to other commercial one.

Finally, in order to demonstrate the high potential of SimulPy, it was performed a more complex simulation, Cgs x IC, varying the VG (0 to 3.3V), as primary parameter, and VB (0V in red, -0.5V in blue and -1.0V in yellow), as secondary parameter. This simulation is not possible in general simulators, avoiding the scientific community to verify a number of parameters in their simulations. The results can be seen in Fig.8.

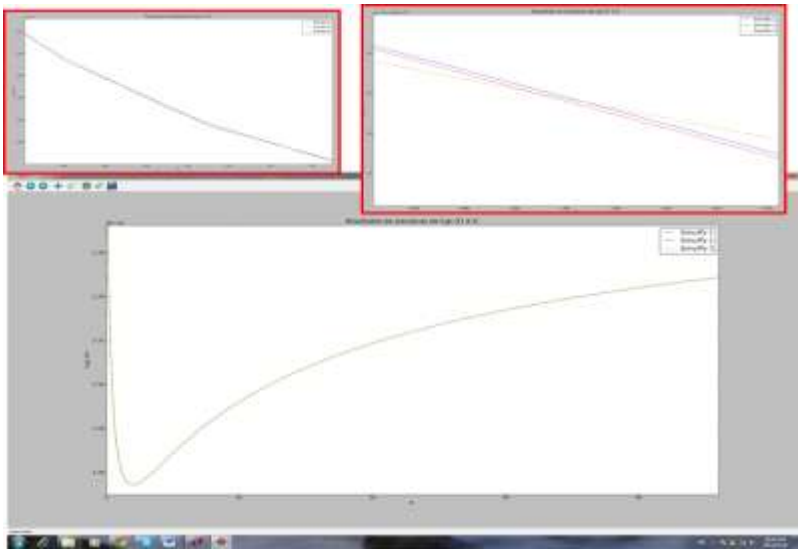


*Figure 8. Cgs x IC results from SimulPy and LTSPICE, for transistors of W/L=25um/25um*

Figure 8 show the high potential and the several possibilities that SimulPy provides to the researcher. It can be seen (bottom) the provided simulation for IC parameter from 0 to 48 and the two top details (both around IC=1) with different levels of zoom in, in order to provide the differentiation of the simulated curves.

## VI. CONCLUDING REMARKS AND FUTURE WORK

It was presented an open-source and free computational tool for Design-Oriented Modeling and Simulation of MOS transistors, called SimulPy. The provided friendly user's interface was achieved through an easy and coherent implementation in PYTHON language and in EKV2.6 compact transistor model, providing direct access to small sets of equations and parameters. Such features lead to the conclusion that this software shows to be ideal for transistor-level design purposes, enabling the designer to manipulate open-source models, while keeping the continuous behavior and a good conformance of the transistor behavior in all regions.

SimulPy is an ongoing project, and is recently supporting the activities in our research team' laboratory. Unfortunately, until the submission of this paper, SimulPy was already under further development, some upgrades being done before making it available to the scientific community. This precaution was taken in order to provide the best possible product and implement some more functionalities and documentation.

Although SimulPy was not already completely available for the research community, what we intend to do as soon as possible, in a near future users will be able to benefit freely from this original authors' work, considering that any modifications that such users make must be released under compatible terms, as provided in the open-source rules. It is desirable that after available, SimulPy must be used and stressed in order to detect any fail, what must be sent to this author in order to provide upgrades.

In the way SimulPy is implemented until now, several options of output and transfer characteristics were implemented, allowing the designer not only to examine the electrical behavior of the transistor but also the charge behavior of each one of its nodes. Besides providing simulations of the respective characteristics under various conditions, it was implemented different "Design-Oriented" functionalities, such as "import and comparison" with a pre-established measurement file (*.txt), calculus of the parametric difference between them and graphical analysis and comparison.

Finally, validation against commercial simulation software LTSPICE, for AMS 0.5 µm CMOS technology, was performed and errors less than 1.2% were observed. Considering the presented examples, it is important to highlight that the software reflects the behavior of the component and presents a high potential to be used by the scientific community. Future works lead to a wide temperature model, including a wide range of cryogenic temperatures, as well as the implementation of 3D graphs and simulations.

## REFERENCES:

[1] Asparuhova, K.; Angelov, G.; Hristov, M.; "Parameter Extraction, Optimization, and Simulation of EKV Model using MATLAB"; *Proceedings of 26th International Conference on Microelectronics (MIEL 2008)*, Serbia; 11-14 may, 2008

[2] Angelov, G.; Asparuhova, K.; "Optimization and simulation of the EKV model using MATLAB"; *ELECTRONICS* 2007; 19–21 September, Sozopol, Bulgaria

[3] Willers, C. J.; et al; "Pyradi: an open-source toolkit for infrared calculation and data processing"; *Proceedings of SPIE 8543, Technologies for Optical Countermeasures IX*, 85430J (November 8, 2012)

[4] "Unix Philosophy." *http://en.wikipedia.org/wiki/Unix_philosophy* (August 2012).

[5] SciLab. *http://www.scilab.org* (2011).

[6] Matplotlib. *http://matplotlib.sourceforge.net/* (2011).

[7] The Python computer language. *http://www.python.org/* (2011).

[8] Numpy. *http://numpy.scipy.org/* (2011)

[9] Bucher, M.; Lallement, C.; Enz, C.; Théodoloz, F.; Krummenacher, F.; "The EPFL-EKV MOSFET Model Equations for Simulation"; *Technical Report; Revision II*, July, 1998

[10] Vittoz, E.; Enz, C.; Krummenacher, F.; "An Analytical MOS transistor model valid in all regions of operation and dedicated to low voltage and low-current applications"; *Analog Integrated circuits and signal Processing*, 8, 83-114, 1995

[11] Bucher, M.; Lallement, C.; Enz C.; "An Efficient Parameter Extraction Methodology for the EKV MOST Model"; *Proceedings of the 1996 IEEE International Conference on Microelectronic Test Structures*, Vol. 9, March 1996

[12] Vittoz, E.; Enz, C.; "MOS Transistor Modeling for Low-Voltage and Low-Power Analog IC Design"; *Microelectronic Engineering;* 39 (1997) 59-76

[13] Orcad PSPICE Reference Manual - Electronics Lab; *http://www.electronics-lab.com/downloads/schematic/013/tutorial/PSPCREF.pdf*; (2013)

[14] Eldo Users Manual; *http://www.engr.uky.edu/~elias/tutorials/Eldo/eldo_ur.pdf;* (2013)

[15] Design Simulation and Device Models; *http://www.linear.com/designtools/software/#LTspice* (2013)

[16] Matlab. The MathWorks Inc, *http://www.mathwo rks.com* (2011).