Solis: a modular, portable, and high-performance 1D semiconductor device simulator

Sidi Ould Saad Hamady[1]

Abstract

This article presents Solis, a new modular, fast, and portable one-dimensional (1D) semiconductor device simulator designed and developed particularly for photovoltaic solar cells. Solis is coded in standard C++, runs natively on Windows and Linux, and is freely available to download. All the physical models and excitation parameters can be fully controlled using the fast embedded Lua scripting engine. Solis includes a useful set of tools coded in C, such as a code editor, graphical device editor, and data plotter. A material parameters database including well-known semiconductors and their alloys is also included. The Solis calculation engine implements the drift–diffusion transport model and takes into account indirect recombination processes (with user-defined deep or shallow levels) as well as radiative and Auger recombination. The anode and cathode parameters, including the refractive index, extinction coefficient, recombination speed, and barrier height for the Schottky contact, are fully included. It natively handles spontaneous and piezoelectric polarization, which is key for the development of devices based on wurtzite materials. The values of all the material parameters can be graded arbitrarily in position, offering flexibility for the simulation of heterostructure solar cells. In addition, Solis offers functionality needed by researchers to analyze the simulation results, including graphing (of the band diagram, carrier concentration, ionized dopant and trap concentrations, current–voltage and capacitance–voltage characteristics, quantum efficiency, etc.) with complete control of the graph and a useful mathematical console. Solis simulation results for a solar cell using Earth-abundant elements are presented herein. One of the main issues in the development of such solar cells, viz. the inhomogeneity due to phase separation in the absorber, is highlighted.

Keywords Solis · Simulation · Semiconductor devices · Scripting · Design · Portable · Solar cell · Heterostructure

1 Introduction

Numerical simulation is an important step in the design and elaboration of novel photovoltaic solar cells. Over the years, several noncommercial one-dimensional device simulators have been developed [1–3], contributing to the development of the photovoltaic industry. These tools are not portable and can hardly be upgraded or extended; For example, PC1D [1], the main one-dimensional simulator, was developed in the 1990s for the Windows operating system and uses the nonportable Microsoft Foundation Class (MFC) library with a code basis mixing calculation algorithms and user interface routines. It has been widely used in the photovoltaic community, with

the last official version released in June 2003 and updated in 2008 with the release of its source code.[1] SCAPS [3], another popular one-dimensional simulator with a large number of features and regular updates, uses the proprietary National Instruments LabWindows CVI development framework used in virtual instrumentation and running only on Windows. The last official version of AMPS-1D [2], one of the three main noncommercial device simulators, dates from 1997, and runs natively only on Windows 95 up to XP (requiring a Windows 95 up to XP virtual machine to run on Windows 7/8/10).[2]

Secondly, the existing tools, mainly the three most widely used ones presented above, do not use some paradigms in computer science related, for instance, to modularity,

✉ Sidi Ould Saad Hamady
    sidi.hamady@univ-lorraine.fr

1   LMOPS, Université de Lorraine, CentraleSupélec,
    F-57070 Metz, France

---

[1] An unofficial release of PC1D named PC1D-mod was developed with minor changes but unfortunately removing the ability to handle heterostructures.

[2] As for PC1D, there is an unofficial release of AMPS-1D named wxAMPS, with changes mainly related to the user interface.

portability, interoperability, and versatility. None of these free simulators includes a scripting language like in commercial simulators.[3] SCAPS uses a mechanism tightly linked to the user interface to automate user actions, without being a scripting language. Including a scripting language is nowadays necessary to extend the possibilities of the simulator for the development of, for example, specific physical models, graded devices, complex nonhomogeneous structures, or optimization of material parameters. On the other hand, these tools do not separate the user interface from the simulation engine; they are mixed by design. The nonmodularity, lack of separation between the user interface and backend simulation engine, and the wide use of nonportable proprietary formats and frameworks automatically imply the features underlined here and highlight the limitations now faced by these existing tools.

Thirdly, Solis includes features that are not included in the mentioned 1D simulators. It can handle spontaneous and piezoelectric polarizations, an important feature for wurtzite semiconductor materials such as gallium nitride and related alloys. In addition, it handles Schottky contacts and heterostructures in a more comprehensive way and has the capability to simulate graded semiconductor devices easily.

These facts justify the need for a new simulator that takes into account, at each step, the criteria of portability, modularity, versatility, and performance, which are now becoming crucial considering the development of heterogeneous computing platforms and the central role of simulation in the development of novel devices. I therefore developed Solis, a new 1D device simulator particularly adapted for solar cell simulation. Solis runs natively on Windows (from XP to 10) or Linux (Debian, CentOS, Ubuntu, etc.) and is easily portable to other platforms including, but not limited to, Android and macOS. It is freely available to download from the author's website[4] [4]. Its standard C++-coded simulation engine, optimized at each step of its development process, ensures high performance while retaining complete portability. The embedded fast just-in-time Lua scripting engine gives the user control over every aspect of the simulation.

Solis implements the drift–diffusion model by solving the Poisson–continuity equations, using both the coupled Newton method and the iterative Gummel decoupled method. It can also calculate the band diagram of any heterostructure (using the given database parameters or adding new material parameters), the potential, field, and carrier concentration distributions, the current–voltage characteristic in the dark and under illumination (with a monochromatic, AM1.5, or any user-defined spectrum), the capacitance-voltage characteristic, the external quantum efficiency, etc. It takes into account various recombination processes and can handle both nonideal ohmic and Schottky contacts, and can also include spontaneous and piezoelectric polarizations in wurtzite III–N materials. Solis is complementary to some very new tools such as Simudo [5] and Solcore [6]. Since Solis is fully implemented in C++ and offers high performance while being fully extensible through the integrated scripting language, it is more suitable for 1D simulation and numerical optimization, while these Python tools are suitable for specific simulations including, e.g., quantum effects or intermediate-band materials.

2 Modeling, numerical methods, and implementation

The Solis simulation engine uses the drift–diffusion model, which is the de facto standard physical model used to describe and calculate the electrical properties of semiconductor devices [7]. Solis implements both the coupled Newton method and the decoupled Gummel method to solve the model equations.

2.1 Drift–diffusion model

The drift–diffusion model is composed of the Poisson equation and the two current continuity equations, as shown in one-dimensional (z) and *normalized* form in Eq. 1. The normalization factor in the Poisson equation is taken to be equal to unity, using the Debye length as the position normalization parameter.

---

[3] For example, the commercial reference simulator Atlas® from Silvaco® includes a C interpreter that can be used to extend it and develop custom physical models.

[4] The latest Solis release (extracted total size less than 10 MB) is available to download:

http://www.hamady.org/download/solis_windows_64bit.zip (for Windows XP/Vista/7/8/10 64 bit).

http://www.hamady.org/download/solis_linux_64bit.tgz [for Linux (Debian, CentOS, Ubuntu, etc.) 64 bit].

The Solis user manual is also available to download here: http://www.hamady.org/download/solis_simulator.pdf.

Table 1 The normalization parameters used in the drift–diffusion model

Parameter	Normalization factor
Permittivity	ε_0
Potential	Thermal voltage, kT/q
Energy	Thermal energy, kT
Carrier concentration	$N = 10^{22}$ m^{-3}
Space	Debye length, $L = \sqrt{\varepsilon_0 kT/(q^2 N)}$
Mobility	$\mu_0 = 0.1$ m^2/V-s
Recombination	$\beta_0 = 10^{-20}$ m^3/s
Continuity factor	$\gamma^2 = \mu_0 kT/(\beta_0 q N L^2)$

$$\frac{\mathrm{d}}{\mathrm{d}z}\left(\varepsilon(z)\frac{\mathrm{d}V(z)}{\mathrm{d}z} + P(z)\right) = n(z) - p(z)$$
$$+ N_A^-(z) - N_D^+(z)$$
$$+ \sum_t N_{tA}^-(z) - \sum_t N_{tD}^+(z)$$
$$\frac{\mathrm{d}J_n}{\mathrm{d}z} = -\,\mathrm{GR}(z)/\gamma^2 \tag{1}$$
$$\frac{\mathrm{d}J_p}{\mathrm{d}z} = +\,\mathrm{GR}(z)/\gamma^2$$
$$J_n(z) = -n(z)\mu_n\frac{\mathrm{d}\phi_n(z)}{\mathrm{d}z}$$
$$J_p(z) = +p(z)\mu_p\frac{\mathrm{d}\phi_p(z)}{\mathrm{d}z},$$

with the electrostatic potential V, the spontaneous and piezo-electric polarization P (if any), the permittivity ε, the free electron concentration n, the free hole concentration p, the ionized donor N_D^+ and ionized acceptor concentration N_A^-, and the ionized donor- N_{tD}^+ and acceptor-like trap concentration N_{tA}^-. The concentrations can vary with the position coordinate z. J_n and J_p are the electron and hole currents, and GR is the net generation term. γ^2 is a normalization parameter (the continuity factor shown in Table 1). μ_n and μ_p are the electron and hole mobilities, while ϕ_n and ϕ_p are the normalized quasi-Fermi levels.

All the parameters in these equations are *dimensionless*, i.e., normalized, using the normalization parameters defined in Table 1. The currents are expressed in terms of quasi-Fermi levels for simplicity, and include both drift and diffusion terms. The net generation term GR(z) includes light-induced carrier generation (photogeneration) and the various recombination phenomena, including nonradiative Shockley–Read–Hall (SRH), radiative, and Auger mechanisms.

The electron and hole concentrations are given by

$$n(z) = n_i e^{(+V(z)+V_n(z)-\phi_n(z))}$$
$$p(z) = n_i e^{(-V(z)+V_p(z)+\phi_p(z))}, \tag{2}$$

where n_i is the intrinsic carrier concentration. V_n and V_p are the band parameters for heterostructures and nonuniform graded materials (i.e., for homostructures $V_n = V_p = 0$).

The net generation term GR(z) is expressed as

$$\mathrm{GR}(z) = G(z) - R_D(z) - R_I(z) - R_A(z)$$
$$G(z) = \int_\lambda \alpha(\lambda,z)\phi(\lambda)\exp(-\alpha(\lambda,z)z)\,\mathrm{d}\lambda$$
$$R_D = \beta(np - n_i^2)$$
$$R_I =$$
$$\sum_t \frac{(np - n_i^2)}{\tau_p(n + n_i\exp(E_t - E_i)) + \tau_n(p + n_i\exp(-E_t + E_i))}$$
$$R_A = (np - n_i^2)(C_n n + C_p p) \tag{3}$$

where $G(z)$ is the photogeneration rate, $R_D(z)$ is the direct (radiative) recombination rate, $R_{I(z)}$ is the SRH recombination rate, and $R_A(z)$ is the Auger recombination rate. $R_I(z)$ account for dopants and traps (where "t" stands for trap), each described by an activation energy E_t and electron and hole lifetimes τ_n and τ_p. E_i is the intrinsic Fermi level. $\alpha(\lambda)$ is the absorption coefficient, varying with the light wavelength λ, and ϕ is the incident light flux. β is the direct recombination coefficient. C_n and C_p are the Auger coefficients for electrons and holes, respectively.

2.2 Solving methods and Solis implementation

The drift–diffusion model as defined in Eq. 1 is solved to determine the potential $V(z)$ and the quasi-Fermi levels $\phi_n(z)$ and $\phi_p(z)$. The other quantities are then calculated using these fundamental parameters. The Scharfetter–Gummel discretization scheme [8] is used to ensure the numerical stability needed in this partial differential equation (PDE) system. The discretized PDE system is then solved using two methods implemented in Solis, giving complete flexibility:

- The Gummel iterative decoupled method. In this method, the three equations are solved iteratively until convergence: Starting with initial guess vectors $V(z)$, $\phi_n(z)$, and $\phi_p(z)$, Solis solves the discretized Poisson system using the Newton–Raphson method and determines the new value of the potential vector $V(z)$, which is then used to solve the electron continuity equation and determine the $\phi_n(z)$ vector. $V(z)$ and $\phi_n(z)$ are then used to solve the hole continuity equation to obtain the $\phi_p(z)$ vector, and the process iterates until convergence. A damping strategy is implemented in Solis to ensure convergence in most cases, since this method can converge slowly for problems where the three drift–diffusion equations are highly coupled.
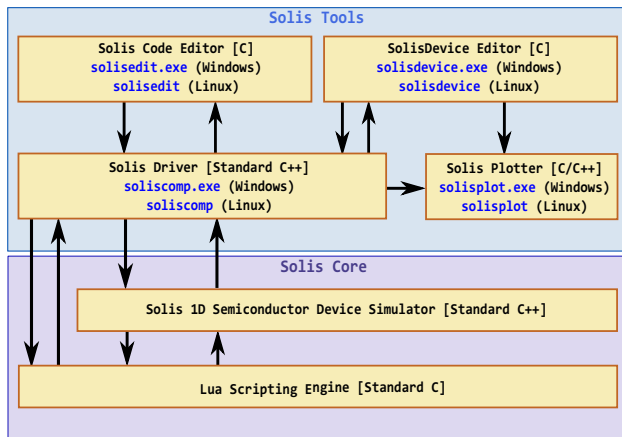
Fig. 1 A schematic of the Solis architecture. The core part of Solis is entirely coded in standard C++ (the Lua engine being coded in standard C) and is easily portable to almost all operating systems, in addition to the natively supported Windows and Linux systems. The Solis code editor, device editor, and data plotter are developed in C and use the IUP GUI toolkit coded in C [9] and the Scintilla component coded in C++ [10]

- The Newton method solves the full discretized nonlinear drift–diffusion PDE system simultaneously using the Newton–Raphson method. This method is suitable for highly coupled cases and can converge fast, providing that a good starting solution is given. Solis provides a starting point for the solver to speed up the convergence, and can also use the Gummel method to refine the initial guess.

Solis can handle both Dirichlet and Neumann boundary conditions. The device contacts can be nonideal ohmic or rectifying (Schottky). The numerical parameters, such as the starting initial guess, the damping parameter and tolerance, and the boundary conditions, can be set by the user to handle specific structures, even if Solis automatically adapts these parameters for convergence.

As underlined above, Solis has been developed with modularity, versatility, and performance as the main criteria and guidelines. Its general architecture is shown in Fig. 1, and its main components are:

- The simulator driver, *soliscomp.exe* (or *soliscomp* under Linux), controlling the simulator engine. The driver and the simulation engine are developed in standard C++, implementing the numerical methods described above and handling the whole simulation process: input file format and device definition, memory management, data input and output, and solving procedures. The simulator driver also controls the embedded Lua scripting engine that can be used to implement user-defined physical models and parameters, permitting advanced control of the

simulation process. The simulator driver is a command-line executable with a device input in a simple and clean text format. It can be easily controlled and automated by any language (and thus the embedded Lua interpreter), giving the user unprecedented flexibility and control over the simulation.
- A graphical device editor, *solisdevice.exe* (or *solisdevice* under Linux), implemented in C. This tool provides an easy-to-use graphical frontend to Solis.
- An advanced code editor, *solisedit.exe* (or *solisedit* under Linux), implemented in C, to edit the device structure using the straightforward Solis syntax. This editor offers all the functionality found in modern editors and can be used as a general code editor.
- A data plotter, *solisplot.exe* (under Windows) or *solisplot* (under Linux), implemented in C and C++. This tool is used by Solis to plot the simulation results but could also be used as a standalone data plotter.
- An advanced scientific calculator, *soliscalc*, implemented in C.

The size of the whole Solis distribution, including documentation and examples, is less that 10 MB.

These tools are independent and can be used in a complementary way; For example, *solisdevice* can be used to perform simulations in a complete graphical environment. To develop a complete simulation framework, with custom physical models and optimization scripts, one can use *solisedit* to achieve full control over the code. These tools are, *by design*, compatible; e.g., an input file created with the code editor can be open/modified by the graphical device editor, and vice versa.

The Solis input format is an easy-to-use plain-text format with a simple syntax and some useful features such as variable definition and a mathematical parser.

A user manual, describing the underlying physics, the input syntax, the models used, the implementation, and the software tools, is provided with the Solis distribution.

3 An application to the study of CZTS solar cells

Next-generation thin-film solar cells face, among other challenges, two critical issues: their overall cost and environmental impact. The overall cost is affected by the lack of abundance of the chemical elements contained in the active layers and the elaboration techniques used to fabricate such solar cells. On the other hand, the environmental impact must take into account, for example, the toxicity of the chemical elements used in the elaboration process and the recycling procedure. One of the main compound semiconductors used in thin-film solar cell technology is copper indium gallium
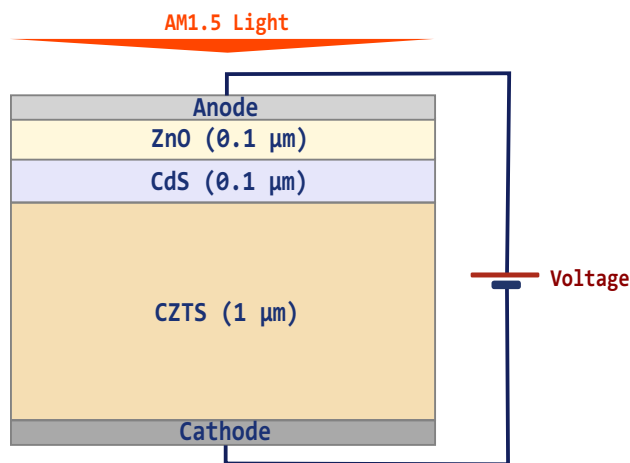
Fig. 2 The ZnO/CdS/CZTS solar cell structure. The ZnO and CdS layers are *n*-doped with a donor concentration of 10^{18} cm^{-3}. The CZTS *p*-doping concentration is equal to 10^{16} cm^{-3}

Fig. 3 The ZnO/CdS/CZTS band diagram at thermodynamic equilibrium

selenide CuIn$_{1-x}$Ga$_x$Se$_2$ (CIGS) with the record efficiency recently reaching 22% [11]. While mature, CIGS technology is affected by the use of indium, a costly and rare element that is already widely used today in screen manufacturing [12]. Other technologies that are far less mature than CIGS are also affected by their use of indium, e.g., those based on the potentially promising indium gallium nitride (InGaN) alloy [13, 14]. To address this issue, several materials using Earth-abundant and nontoxic elements have been developed. Among these materials, copper zinc tin sulfide Cu$_2$ZnSnS$_4$ (CZTS) alloys [and, to a lesser extent, copper zinc tin selenide Cu$_2$ZnSnSe$_4$ (CZTSe)], are become promising materials for use in thin-film solar cells [15, 16]. The record photovoltaic efficiency obtained using these new materials, 12.6% [16], while below the CIGS efficiency, is higher than that of the well-established amorphous silicon solar cells. CZTS has the potential to reach the CIGS level if some issues can be addressed. Some of these issues are related to phase separation in these complex alloys.

The next sections present results for a ZnO/CdS/CZTS solar cell, firstly without phase separation and, secondly, with a phase separation simulated using the capability of Solis to handle arbitrary position-varying physical properties.

3.1 The single-phase CZTS solar cell

Figure 2 shows the simulated structure with the ZnO window, the CdS buffer layer, and the stoichiometric single-phase CZTS absorber, while Fig. 3 plots the band diagram simulated by Solis at thermodynamic equilibrium. Table 2 presents the basic CZTS, CdS, and ZnO parameters used in the simulations. The CZTS extinction coefficient varies with
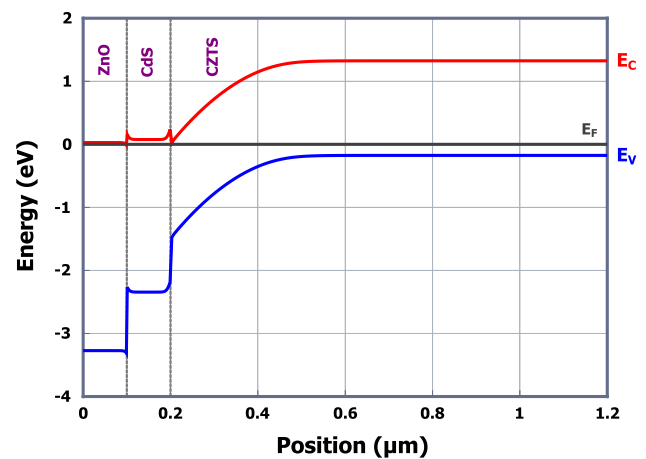
Table 2 The basic parameters of CZTS, CdS, and ZnO used in the simulation

	E_G (eV)	χ (eV)	N_C (cm^{-3})	N_V (cm^{-3})	ϵ_r
ZnO	3.3	4.4	2.2×10^{18}	1.8×10^{19}	9
CdS	2.42	4.2	1.8×10^{19}	2.4×10^{18}	9
CZTS	1.5	4.5	2.1×10^{18}	8.9×10^{18}	6.5

E_G is the bandgap, χ is the electron affinity, N_C and N_V are the density of states in the conduction and valence band, respectively, and ϵ_r is the relative permittivity. The direct and Auger recombination parameters are set to $\beta = 10^{-12}$ cm^3/s and $C_n = C_p = 10^{-30}$ cm^6/s, respectively. The experimental data are taken from Refs. [18, 19]

the the photon energy according to the standard formula in Eq. 4.

$$k(E) = k_0 \left(\frac{2E_G}{E} \right) \left(\frac{E - E_G}{E_G} \right)^{\frac{1}{2}}, \tag{4}$$

where E_G is the CZTS bandgap, E is the photon energy, and k_0 is the extinction coefficient at $E = 2 \times E_G$, taken to be equal to 0.2 [17].

The used AM1.5 solar spectrum is the ASTM G173-03 reference spectrum taken from NREL [20].

Figure 4 shows the current–voltage characteristic of the homogeneous CZTS solar cell obtained using Solis and that simulated using the Atlas® tool from Silvaco®. The obtained photovoltaic efficiency of $\approx$ 19.2% is close to the highest reported value for a CIGS solar cell [11] and highlights the potential of this alloy as an alternative to CIGS. Further optimization, taking the doping and thickness values as parameters, would give the absolute optimum efficiency using, for example, the method presented in Ref. [21]. Another issue with the CZTS solar cell structure is the use of CdS, which contains cadmium. This could be replaced with In$_2$S$_3$ [22] or,
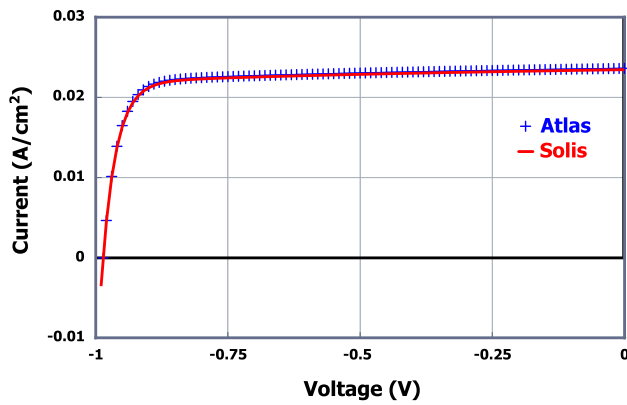
Fig. 4 The current–voltage characteristic of the single-phase CZTS solar cell under illumination by the AM1.5 spectrum. The voltage is applied to the top contact of the *n–p* structure. The symbols indicates the characteristic simulated using Atlas® from Silvaco®, while the straight line shows that simulated by Solis. The photovoltaic performance obtained by Solis and Atlas® is presented in Table 3

Table 3 The photovoltaic performance obtained using Solis and Atlas® for the single-phase CZTS solar cell illuminated with an integrated AM1.5 power of 0.099271 W/cm²

	η (%)	J_{SC} (A/cm²)	V_{OC} (V)	FF (%)
Solis	19.263	0.0234	0.985	82.73
Atlas®	19.260	0.0236	0.986	82.77

η is the efficiency, J_{SC} is the short-circuit current, V_{OC} is the open-circuit voltage, and FF is the fill factor

more consistently, with ZnS [23], containing Earth-abundant elements.

3.2 The multiple-phase CZTS solar cell

In the complex CZTS quaternary alloy, phase separation is usually observed, with evidence of two phases: ZnS and Cu_2SnS_3, in addition to the Cu_2ZnSnS_4 phase [24]. The presence of ZnS is also evidenced between the CZTS absorber and the bottom molybdenum contact [25]. The ability of a simulator to handle any position-varying semiconductor parameters is key to the study and evaluation of the impact of phase separation on the performance of the solar cell. Using Solis, one can set any material graded profile, using the embedded scripting language, and thereby simulate the phase separation in CZTS. To capture the real material properties precisely, the profile of the position-varying parameters (bandgap, affinity, permittivity, density of states, etc.) can be continuous and inspired by experience; For instance, an abrupt square profile could be used, and is in fact equivalent to defining a multilayer structure (or superlattice), but this is not well adapted to a material showing phase separation. A *continuous* profile with no abrupt changes in the
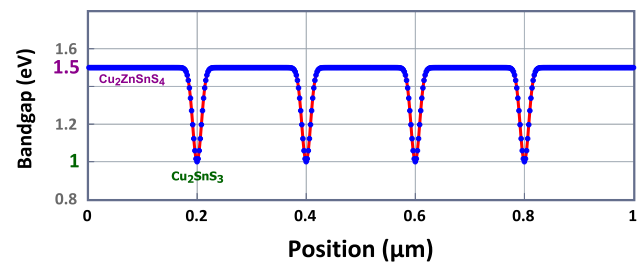

Fig. 5 The periodic inverted Gaussian profile used to model the variation of the bandgap with position in the CZTS alloy based on the Cu_2SnS_3/Cu_2ZnSnS_4 phase separation fraction. The bandgap of Cu_2ZnSnS_4 is ≈ 1.5 eV, while the bandgap of Cu_2SnS_3 is ≈ 1 eV. The fraction between the secondary Cu_2SnS_3 phase and the main Cu_2ZnSnS_4 phase is $\zeta = 0.16$. The same type of function is used for the other parameters, viz. the affinity, permittivity, and density of states
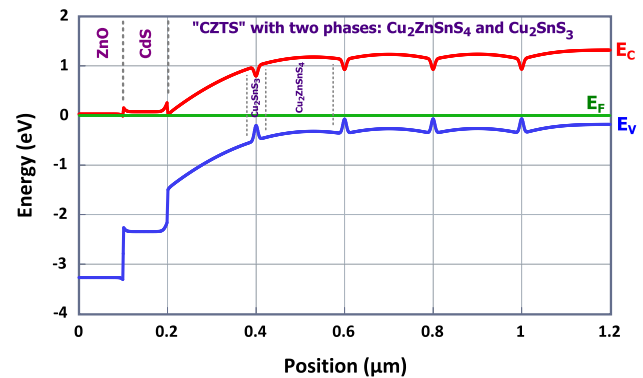

Fig. 6 The ZnO/CdS/CZTS band diagram at thermodynamic equilibrium for the multiple-phase CZTS absorber

physical parameters is more well adapted to the realistic simulation of multiphase alloys. Such a *smoothed* profile is modeled here using a periodic inverted Gaussian function, as plotted in Fig. 5.

The analysis presented here is limited to the two phases usually observed in this material: Cu_2ZnSnS_4 with a bandgap of ≈ 1.5 eV, and Cu_2SnS_3 with a bandgap of ≈ 1 eV. In the chosen profile, the ratio between the Gaussian full-width at half-maximum (FWHM) and the total period represents the fraction between the secondary Cu_2SnS_3 phase and the main Cu_2ZnSnS_4 phase. This ratio ζ is equal to zero for a homogeneous CZTS layer, as presented in Sect. 3.1, and is now increased up to 0.4.

Figure 6 shows the band diagram simulated by Solis at thermodynamic equilibrium for the multiple-phase CZTS absorber with a phase ratio $\zeta = 0.16$.

The photovoltaic performance obtained by Solis for the multiple-phase CZTS solar cell with $\zeta = 0.16$ is presented in Table 4. For comparison, the performance of the homogeneous CZTS absorber is shown in Fig. 4.

Table 4 The photovoltaic performance obtained by Solis for the multiple-phase CZTS solar cell with $\zeta = 0.16$ when illuminated with an integrated AM1.5 power of 0.099271 W/cm^2

η (%)	J_{SC} (A/cm^2)	V_{OC} (V)	FF (%)	R_S (Ω cm^2)	R_P (Ω cm^2)
12.53	0.0213	0.777	75.00	2.2	139

η is the efficiency, J_{SC} is the short-circuit current, V_{OC} is the open-circuit voltage, FF is the fill factor, R_S is the series resistance, and R_P is the shunt resistance
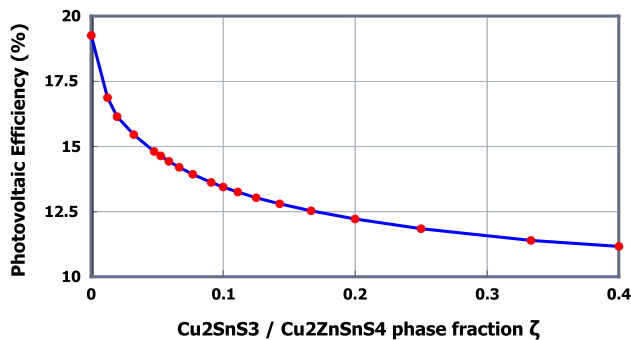
Fig. 7 The variation of the efficiency of the multiple-phase CZTS solar cell with ζ, the fraction between the secondary Cu_2SnS_3 phase and the main Cu_2ZnSnS_4 phase in the absorber

The phase separation dramatically impacts the efficiency of the solar cell, even for a relatively low ζ ratio, and the efficiency $\approx 12\%$ found here approaches the experimental record [16]. To illustrate this behavior, Fig. 7 shows the variation of the photovoltaic efficiency with respect to ζ, the fraction between the secondary Cu_2SnS_3 phase and the main Cu_2ZnSnS_4 phase. The efficiency decreases rapidly to reach 11% for a fraction of 0.4. In this complex heterostructure, with continuously varying CZTS absorber parameters, the change in the electrical field profile causes a decrease in the open-circuit voltage, and the accompanying impact on the photogenerated carrier collection decreases the short-circuit current. Other models, including e.g. a random distribution and interface-related inhomogeneities, could be used to simulate the effect of phase separation. For more complex structures, a more specific and comprehensive approach, in two dimensions (2D) or even three dimensions (3D), should be used to capture the effect of phase separation in the solar cell absorber more precisely.

4 Conclusions

A new modular, portable, and high-performance 1D semiconductor device simulator is presented. This free simulator, Solis, is developed in standard C++, runs natively on Windows and Linux, and is portable to other platforms. It includes a fast scripting engine permitting the development of custom physical models and excitation parameters with a clean and simple language and simulation input. An application to the study of a CZTS heterostructure solar cell is presented, taking into account phase separation in the absorber, modeled using the capability of Solis to handle arbitrary position-varying parameters. The Solis simulator is continuously updated and will include in the coming release a coherent optical propagation model and the possibility to simulate the admittance spectroscopy characteristic, while transient simulation functionality will be included if it turns out to be important for emerging photovoltaic technologies. Tunneling through the anode Schottky contact, for Schottky solar cells or detectors, is already implemented and will be included in the next release.

Compliance with ethical standards

References

1. Basore, P., Clugston, D.: PC1D version 4 for Windows: from analysis to design. In: Conference Record of the Twenty Fifth IEEE (IEEE) Photovoltaic Specialists Conference, pp. 377–381 (1996)
2. Zhu, H.,Kalkan, A., Hou, J., Fonash, S.: Applications of AMPS-1D for solar cell simulation. In: AIP Conference Proceedings, (AIP), vol. 462, pp. 309–314 (1999)
3. Burgelman, M., Nollet, P., Degrave, S.: Modelling polycrystalline semiconductor solar cells. Thin Solid Films **361**, 527 (2000)
4. Hamady, S.O.S.: Software for photovoltaics (2010). http://www.hamady.org/photovoltaics.html
5. Dumitrescu, E., Wilkins, M., Krich, J.: Simudo: a device model for intermediate band materials. J. Comput. Electron. **19**, 111 (2019)
6. Alonso-Álvarez, D., Wilson, T., Pearce, P., Führer, M., Farrell, D., Ekins-Daukes, N.: Solcore: a multi-scale, Python-based library for modelling solar cells and semiconductor materials. J. Comput. Electron. **17**(3), 1099 (2018)
7. Selberherr, S.: Analysis and Simulation of Semiconductor Devices. Springer, Berlin (2012)
8. Scharfetter, D., Gummel, H.: Large-signal analysis of a silicon read diode oscillator. IEEE Trans. Electron. Dev. **16**(1), 64 (1969)
9. The IUP GUI toolkit, (C) 1994–2017 Tecgraf/PUC-Rio (2017)
10. The Scintilla Component (C) 1998–2016 Neil Hodgson (2016)
11. Kamada, R., Yagioka, T., Adachi, S., Handa, A., Tai, K., Kato, T., Sugimoto, H.: New world record Cu(In, Ga)(Se, S)$_2$ thin film solar cell efficiency beyond 22%. In: 2017 IEEE 44th (IEEE) Photovoltaic Specialist Conference (PVSC), pp. 1–5 (2017)
12. Woodhouse, M., Goodrich, A., Margolis, R., James, T., Lokanc, M., Eggert, R.: Supply-chain dynamics of tellurium, indium and gallium within the context of PV module manufacturing costs. In: IEEE 38th (IEEE) Photovoltaic Specialists Conference (PVSC), vol. 2, pp. 1–6 (2012)

13. Jani, O., Ferguson, I., Honsberg, C., Kurtz, S.: Design and characterization of GaN/InGaN solar cells. Appl. Phys. Lett. **91**(13), 132117 (2007)

14. Hamady, S Ould Saad, Adaine, A., Fressengeas, N.: Numerical simulation of InGaN Schottky solar cell. Mater. Sci. Semicond. Process. **41**, 219 (2016)

15. Jiang, M., Yan, X.: Solar Cells-Research and Application Perspectives. InTech (2013)

16. Wang, W., Winkler, M., Gunawan, O., Gokmen, T., Todorov, T., Zhu, Y., Mitzi, D.: Device characteristics of CZTSSe thin-film solar cells with 12.6% efficiency. Adv. Energy Mater. **4**, 7 (2014)

17. Jiang, M., Li, Y., Dhakal, R., Thapaliya, P., Mastro, M., Caldwell, J., Kub, F., Yan, X.: Cu_2ZnSnS_4 polycrystalline thin films with large densely packed grains prepared by sol–gel method. J. Photonics Energy **1**(1), 019501 (2011)

18. Madelung, O.: Semiconductors: Data Handbook. Springer, Berlin (2012)

19. Simya, O., Mahaboobbatcha, A., Balachander, K.: A comparative study on the performance of Kesterite based thin film solar cells using SCAPS simulation program. Superlattices Microstruct. **82**, 248 (2015)

20. NREL. Reference Solar Spectral Irradiance: ASTM G-173 (2004)

21. Hamady, S Ould Saad, Fressengeas, N.: SLALOM: open-source, portable, and easy-to-use solar cell optimizer. Application to the design of InGaN solar cells. EPJ Photovoltaics **9**, 13 (2018)

22. Barkhouse, D., Haight, R., Sakai, N., Hiroi, H., Sugimoto, H., Mitzi, D.: Cd-free buffer layer materials on $Cu_2ZnSn(S_xSe_{1-x})_4$: Band alignments with ZnO, ZnS, and In_2S_3. Appl. Phys. Lett. **100**(19), 193904 (2012)

23. Nguyen, M., Ernits, K., Tai, K.F., Ng, C., Pramana, S., Sasangka, W., Batabyal, S., Holopainen, T., Meissner, D., Neisser, A.: ZnS buffer layer for $Cu_2ZnSn(SSe)_4$ monograin layer solar cell. Sol. Energy **111**, 344 (2015)

24. Walsh, A., Chen, S., Wei, S., Gong, X.: Kesterite thin-film solar cells: advances in materials modelling of Cu_2ZnSnS_4. Adv. Energy Mater. **2**(4), 400 (2012)

25. Shin, B., Gunawan, O., Zhu, Y., Bojarczuk, N., Chey, S.: Thin film solar cell with 8.4% power conversion efficiency using an earth-abundant Cu_2ZnSnS_4 absorber. Prog. Photovoltaics Res. Appl. **21**(1), 72 (2013)