Nuclear Instruments and Methods in Physics Research B 269 (2011) 2133–2138

Contents lists available at ScienceDirect

Nuclear Instruments and Methods in Physics Research B

journal homepage: www.elsevier.com/locate/nimb

Ion beam irradiation of nanostructures – A 3D Monte Carlo simulation code

C. Borschel *, C. Ronning

Institute for Solid State Physics, Jena University, Max-Wien-Platz 1, 07743 Jena, Germany

ARTICLE INFO

Article history:
Received 22 March 2011
Received in revised form 9 June 2011
Available online 19 July 2011

Keywords:
Monte Carlo
Ion beam irradiation
Implantation
Nanostructures
Nanowire

ABSTRACT

A computer code for the simulation of ion beam irradiation of nanostructures has been developed. The code simulates the transport of energetic ions through matter by means of a Monte Carlo algorithm, similar to the often-used TRIM code (Ziegler et al. (1985) [1]). New effects occur compared to bulk, when irradiating nanostructures, which are of the same size as the ion range or the damage cascade. To account for these effects, the target in our code does not consist of layers like in TRIM but can be defined as an arbitrary 3-dimensional structure. This allows to obtain more accurate 3D distributions of implanted ions and implantation damage for nanostructures, which cannot be described by a stack of layers. We demonstrate the functionality of the code by comparing simulations with ion beam implantation into nanowires.

1. Introduction

Ion beam irradiation of bulk or layered targets has been studied extensively in theory and experiment for many years and is widely applied. Today, the ion beam irradiation of nanostructures has come into focus [2], some example being the controlled structuring, shaping, or realigning of nanowires [3–6] or for doping of nanostructures [7,8]. New effects may occur during irradiation of nanostructures, because the ion range or dimension of the collision cascades becomes comparable to the size of the nanoobject itself [9–11].

In order to understand ion beam induced processes in the target material and in order to determine the distribution of the implanted ions and the distribution of ion beam induced damage or sputtering yields, it is necessary in most cases to complement irradiation experiments with computer simulations. Frequently, this is done by means of Monte Carlo (MC) codes, a variety of these has been presented in the past decades [1,12–18]. However, the irradiated targets in these codes usually consist of a number of layers, which are often assumed to have infinite lateral extension and each layer consisting of one homogeneous material. Even though some codes allow the extraction of the lateral ion and damage distribution (for example TRIM [1] from `collision.txt` or `RANGE_3D.txt`), the target still consists of layers and effects caused by lateral structures or lateral limitations of the target cannot be simulated accurately. For a few cases, TRIM-like Monte Carlo codes have been used that take into account the full 3D geometry, but these codes are very specialized for their respective applications and are not widely available [6,19]. Monte Carlo codes specialized to studying the formation of ripples under ion beam irradiation have been reported [20].

Several advanced codes have been presented, which are intended and optimized for the simulation of ion implantation into 3-dimensional semiconductor devices, for example MCIMPL [21] or TOMCAT [22]. These codes take into account effects like channeling and employ a number of special techniques to save computation time; for example trajectory replication, spatial Octree division [23,24] or convolution of Monte Carlo calculated point response functions [25]. A different type of programs that simulate ion beam irradiation can be found in the area of FIB-milling: there exist various codes that can simulate sputtering effects for non-flat targets and successfully take into account effects like redeposition and reduced sputtering in wells and related effects [26–28]. In these codes, the target is represented as an evolving surface, with a different height at each lateral position. Furthermore, these codes focus on high fluencies, where sputtering is the dominant process. They are not suitable to obtain distribution of implanted ions and implantation damage in nanostructures.

In order to bridge the gap between the "non-3D" MC codes, the comparatively complex semiconductor device codes and the high fluence "FIB-sputter-codes", we present an open source Monte Carlo code in this study, named *iradina* (**i**on **r**ange **a**nd **d**amage **i**n **na**nostructures), which allows a full 3D definition of almost arbitrary targets and can thus account for effects occurring in irradiation of nanosized objects with sizes comparable to the ion range. In this code, the simulation of ion transport through the target material proceeds similar to TRIM-like codes, using the random phase approximation (RPA), the binary collision approximation (BCA)

* Corresponding author.
E-mail address: borschel@iradina.de (C. Borschel).

0168-583X/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.nimb.2011.07.004

and the central potential approximation (CPA). The main difference lies in the target definition.

It should be noted that for very small nanostructures, ion beam irradiation can also be simulated by using molecular dynamics (MD) [2]. The MD simulations yield very accurate results compared to Monte Carlo (MC) simulations as they avoid a number of simplifying approximations made in the MC approaches. However, the necessary computation times are orders of magnitude larger, yielding MD simulations unpracticable to obtain distribution of ions and implantation damage or sputter yields for nanostructures larger than a few tens of nm [11].

2. Program description

Iradina simulates the transport of ions through a target consisting of a number of cells by means of a Monte Carlo (MC) code.

Several very fast routines are adapted from the open source computer code "corteo", which has been presented by Schiettekatte [29]. These fast routines account for significant savings in computation time compared to TRIM. Due to this increased speed and modern cpu capabilities, techniques like trajectory replication are not necessary and large numbers of ions can be simulated directly.

First, the target structure and composition needs to be represented in the program. A number of various materials, which are present in the target, can be defined, and each of these materials may contain any number of different elements in arbitrary concentrations. For each element in each material a lattice binding energy E_b, a displacement energy E_d and a surface binding energy E_s must be defined.

The structure of the target is described by a rectangular box of arbitrary size, which is subdivided into equal-sized rectangular cells. In each direction, the size and number of the cells can be defined independently. Each cell can contain one of the defined materials or alternatively vacuum. Furthermore, each cell has a number of counters, to record how many ions are implanted into this cell and how much damage is produced (for example vacancies and interstitials of each type). These counters are used in the end to obtain the spatial distributions of implanted ions and damage.

To simulate the ion beam irradiation, a large number of ions are consecutively assumed to impinge on the target under a predefined angle and the transport of the ions through the target is simulated by a Monte Carlo method similar to other MC ion transport codes. Detailed descriptions and discussions of the physical background of these simulations can be found in literature [1,30]; thus, the process will only be described briefly here.

An ion is described at any time by its momentary position in the target, its energy and its moving direction (the velocity vector normalized to unit length). Each ion is followed through the target until it leaves the target or until it is stopped within the target (meaning it has lost all its energy). The path of the ions is followed step by step in a loop, where each step corresponds to one collision with one target nucleus. The following actions take place in each step:

Depending on the ion's momentary position, the material is determined in which the ion currently moves. A flight length is selected randomly according to a Poisson distribution, such that the mean flight length corresponds to the average atomic distance within this material. Over the complete flight length, the ion's energy loss to target electrons is assumed to be constant. This electronic stopping, which mainly depends on the ion energy and the target composition, is obtained from precalculated tables, which are loaded into memory on program startup. The logarithmically scaled stopping tables from corteo are used, because they offer the advantage of extremely fast access, by utilizing a neat indexing mechanism, which avoids the lengthy calculations of logarithms (details in [29,30]). The tables were calculated using SRIM [1] (SRModule.exe). The Bragg rule is applied to calculate electronic stopping in compounds. The energy loss is varied randomly to account for the effect of energy loss straggling. This is calculated using the formula and fit data provided by Yang et al. [31], the fast routine from corteo is used here as well [29].

The electronic energy loss is subtracted from the ion energy and the ion position is updated by adding the product of unit velocity vector and flight length. If the new position is outside the target, the program either disregards this ion, or, in case periodic boundary conditions are defined in the relevant direction, the ion's position is set to the opposite side of the target.

Next, the material at the new position of the ion is determined. If the material is not vacuum, a collision of the ion with a target atom will take place. In case the material contains more than one element, the collision partner is randomly selected according to the relative atomic fractions of the elements in the material. Once the collision partner is determined, an impact parameter is randomly selected between zero and a maximum impact parameter that depends on the flight length (this is done like in corteo, for details on how to select an appropriate impact parameter, see [30]). Knowing the impact parameter and the energy of the ion, the scattering angle can be calculated by solving the scattering integral. The scattering angle depends on the interatomic potential used to describe the interaction. *Iradina* uses the universal potential [1]. Instead of solving the scattering integral numerically or using the MAGIC algorithm [12], we make use of the much faster corteo routine, which determines the scattering angle from precalculated tables with a fast indexing mechanism, following the idea of Yuan et al. [32]. From the scattering angle and a randomly selected azimutal angle, the new flying direction of the ion is calculated. From the kinematics of the collision, the energy transferred to the target nucleus is calculated and subtracted from the ion energy (this is called nuclear or elastic energy loss).

With the new direction and energy, the ion proceeds to the next collision step, unless its energy has dropped below a defined minimum energy (for example 5 eV or the lowest surface binding energy in the target). In the latter case, the ion is assumed to be stopped completely, and the current cell's ion counter is increased by one.

In order to obtain data about the ion beam induced damage, the energy transfer to the collision partner of the ion (the "recoil") needs to be considered in each step. If a recoil obtains more energy than its displacement energy E_d, it is released from its site, leaves behind a vacancy, and becomes itself a moving projectile in the target, that can cause further displacements ("collision cascade"). The transport of the recoils through the target is simulated in the same way as the ion (using recursive function calls). The moving recoil is followed until it leaves the simulation volume or until it has lost its energy and becomes an interstitial (in which case the interstitial counter for the respective element and cell is increased by one).

A special case of collisions is considered in the code: it may happen that the projectile loses so much energy to the recoil that its remaining energy is below the displacement energy. Now, if the two collision partners are of the same type, no vacancy is created, because the projectile is assumed to replace the recoil. Depending on whether the displacement of an atom leads to a vacancy or a replacement, the respective vacancy- or replacement-counter for the simulation cell is increased by one.

Another special case concerns surface atoms: in order to remove a surface atom and eject it into vacuum, less energy is necessary than for removing a bulk atom from its site (the energy barrier for removing a surface atom is E_s instead of E_d for the bulk atom). A recoil atom is considered to be a surface atom in the code, if its distance to a neighboring vacuum cell is less than the average atomic spacing. In case the recoil atom is a surface atom and its velocity

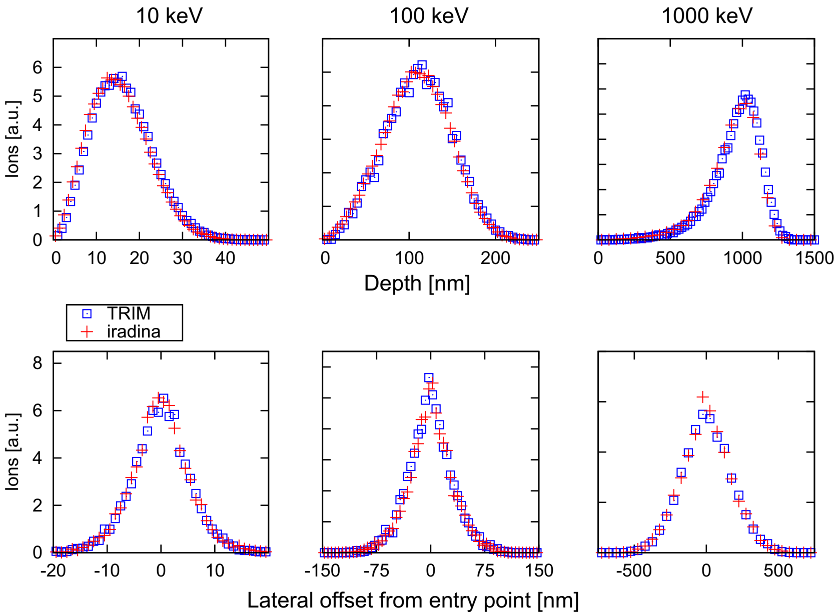

Fig. 1. Comparison of *iradina* and TRIM (SRIM 2008.03): simulations of Ar implantation into bulk Si for energies of 10, 100, and 1000 keV at perpendicular incidence. Top row: depth distribution of implanted ions, bottom row: lateral distribution of implanted ions (integrated over depth).

after the collision points into a vacuum cell, the code uses the surface binding energy instead of the displacement energy in order to determine whether the atom has to be displaced. Ions or recoils ejected from the surface are not discarded, their way through vacuum is also followed from cell to cell: this is necessary, because after traveling a short distance through vacuum they may again get in contact with material and cause further damage – an effect which cannot be accounted for by codes using flat targets as in TRIM.

After a predefined number of ions has been simulated, the code stores the cell counters as 3-dimensional histograms of ion and damage events, from which the distribution of implanted ions and the distributions of the various types of damage can be extracted.

Two issues caused by the cellular geometry should be mentioned: (1) Surface planes that are not perpendicular to one of the cartesian axes become stepped. This is of little concern to the distribution of implanted ions as long as the cells are small compared to the complete target. However, this distorts the sputter yields, because the stepped plane in the simulation has a larger surface area than the real plane. Additionally, sputtered atoms may re-enter the target and cause further sputtering. (2) When a projectile crosses a cell boundary the material might change, which could require a correction of the electronic stopping. However, this is neglected in the current version for *iradina*, because flight paths are usually much shorter than cell dimensions, and for the small energies used when implanting nanostructures, nuclear stopping dominates in many cases anyway.

3. Simulation results and examples of applications

3.1. Comparison to TRIM

Since the computation is similar to TRIM [1], *iradina* should generate comparable results when using layered or bulk targets. We use such comparisons as a test in order to verify correct basic functionality of our code: the implantation of Ar ions into bulk silicon with a flat surface and with different Ar energies was simulated with both, TRIM and *iradina*. The resulting depth distributions and lateral distributions of implanted ions are compared in Fig. 1. The distributions are in good agreement for all three ion energies. Comparisons have also been made for lighter and heavier ions (H, N, Ag, and Pb in the range of 10–1000 keV). Mostly, the results are in good agreement as for argon; only for hydrogen at 10 keV *iradina* underestimates the straggling a few percent compared to TRIM.

The production of damage within the collision cascades is compared in Fig. 2, which shows the depth distribution of displacements occurring during ion beam irradiation. The results are also in good agreement. One minor discrepancy occurs here. A small fraction of the displacements does not lead to vacancies but to replacements as explained in Section 2. For Ar implantation into silicon, the results from *iradina* yield a replacement fraction of about 30% (from all displacements occurring). This is in agreement with the SRIM manual,[1] however, the calculations from TRIM only show a replacement fraction of about 10%.[2] However, the method to use the displacement energy as a sharp cut-off to calculate the replacement fraction is a very simple model, that cannot be expected to yield very accurate results compared with experiments. Furthermore, this does not affect the shape of the distribution of damage but only its amplitude by a factor of about 0.7. For many materials, a number of point defects anneal out during the ion beam implantation process (“dynamic annealing”), thus, the absolute simulated damage concentration is not very accurate for irradiation at $T > 0$ K anyway.

As *iradina* makes use of the fast routines from corteo [29] for the calculation of stopping power and scattering integrals, it is faster than TRIM by about one to two orders of magnitude, depending on the simulation parameters. It should be noted though, that this increase in computation speed is bought by extensive access to large tables in memory, so this increase depends not only on the simulation parameters, but also on the architecture of the computer in use, the available memory, the operating system and on the size of the cpu cache.[3] It should be noted that the division into

[1] See www.srim.org

[2] Since the authors do not have access to the source code of the current version of TRIM, they cannot determine the exact reason for this discrepancy.

[3] Comparisons were done on a standard personal computer: Intel Core 2 Duo E8200, 2.66GHz, 6 MByte L2 cache memory, 2 GB memory, Linux 2.6 and Windows.

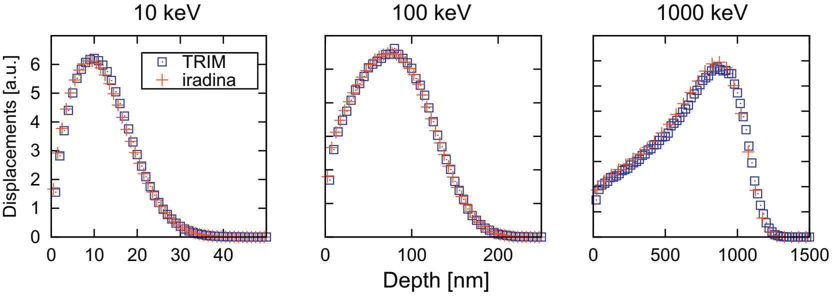

Fig. 2. Comparison of *iradina* and TRIM (SRIM 2008.03): simulations of Ar implantation into bulk Si for energies of 10, 100, and 1000 keV at perpendicular incidence. The absolute depth distributions of displacements occurring in the collision cascades are shown.

many cells has little effect on the computation time. Cells are stored in memory in a linear fashion and the calculation of the cellindex from the momentary ion position is simple and fast.

3.2. Doping of semiconductor nanowires

Doping of semiconductor nanowires by ion implantation is discussed in this section as an example for the application of *iradina*. Fig. 3a shows a typical implantation box profile for Zn implantation into bulk GaAs. Implantation profiles for different ion energies are superimposed with appropriate factors in order to create an almost homogeneous doping profile over a depth range of about 250 nm.

Iradina was used to simulate ion beam implantation into a GaAs nanowire (diameter of 150 nm) under an oblique angle, using the same ion energies and fluencies as for bulk. For the simulation, it is not necessary to represent the complete nanowire. Only a thin disk of the nanowire is used instead, and periodic boundary conditions (PBC) are applied along the nanowire axis (z-direction), as illustrated in Fig. 3b. Thus, the simulation implies translational symmetry along the nanowire axis and thus infinite length. This is a good approximation because the length of a real nanowire is typically 100 times greater than its diameter.

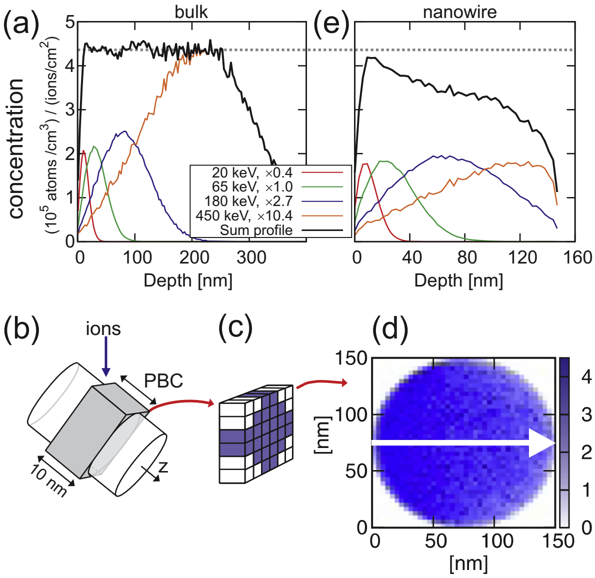

Fig. 3. (a) Simulated concentration profiles for implantation into bulk material with different energies and resulting sum profile. (b) Schematic illustration of simulation of ion implantation into nanowire. Only the grey shaded box is simulated and periodic boundary conditions (PBC) are applied along the NW axis (z-direction). (c) The simulation volume is divided into cells. (d) Distribution of implanted ions in the simulated nanowire (in arb. units). (e) Shows the simulated depth distribution of implanted ions in the nanowire for the same energies and fluencies as in (a). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The slice from the nanowire, which is used for the simulation, is divided into an array of 40 × 40 cells, resulting in cell dimension of 3.75 nm × 3.75 nm, see Fig. 3c. In the third direction, no subdivision into more than one cell is required due to the translational symmetry along the z-axis. In each of the 1600 cells, the implanted ions are counted. The resulting two dimensional implantation profile is shown in Fig. 3d. A linear cross section through this profile was made (marked by the white arrow) to compare the implantation depth profile (Fig. 3e) with the bulk profile (Fig. 3a).

First, the implanted profile is not homogeneous as compared to bulk and secondly, fewer ions are implanted into the nanowire. This is due to the lateral limitations of the nanowire and hence the significant probability of ions leaving the nanowire on the side. For the lowest energy (20 keV) the ion range is much less than the nanowire diameter, the implanted profiles in bulk and NW are almost the same. However, for the higher energy (450 keV), when the ion range becomes comparable to the nanowire diameter, the fraction of ions leaving the nanowire increases to about 50%.

When using implantation profiles obtained from TRIM simulations (bulk), as done in [33,34], it has been observed that charge carrier concentrations in ion beam doped nanowires are lower than the expected doping concentrations from ion implantation. While this can partly be explained by effects like autocompensation, migration and only fractional ionization of dopants, the lower concentration of implanted ions in nanowires must be taken into account and contributes significantly to this observation.

This effect can be measured directly for highly doped nanowires: we implanted GaAs nanowires (diameter of 40 nm) with Mn ions of 50 keV and a fluence of $1 \cdot 10^{16}$ ions/cm^2, details will be published elsewhere. TRIM simulations (bulk targets) yield an average Mn concentration of 5%. We used *iradina* to simulate the implantation into nanowires, similar to the example illustrated in Fig. 3. From the simulation results of *iradina*, an average Mn concentration of about 2.5% was predicted. The implanted nanowires were inserted into a transmission electron microscope and the concentration was measured using energy dispersive X-ray spectroscopy. Indeed, the measured concentrations on different points on the nanowire were all between 2% and 3%. Clearly, bulk simulations (TRIM) overestimate the concentration by about a factor of 2, while simulation results with the correct nanowire geometry (*iradina*) yield much more accurate concentrations.

3.3. Controlling nanostructure morphology by ion beams

Apart from the concentration of implanted ions, the distribution of ion beam induced damage in nanostructures can also be of interest, as illustrated in the following example: nanowires have been observed to bend under ion beam irradiation with high fluencies

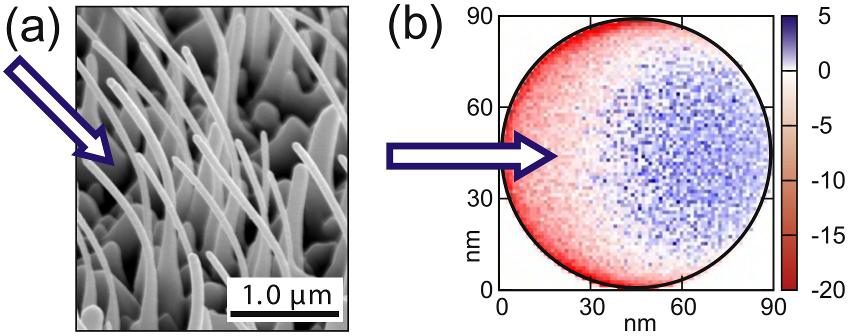

Fig. 4. (a) SEM image of bent ZnO nanowires after irradiation with 100 keV Ar ions. Arrow indicates ion beam direction. (b) Result from *iradina* simulation showing the distribution of damage within the nanowire. The difference value of interstitials minus vacancies is shown (arb. units). Blue: excess interstitials, red: excess vacancies. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

[3,4,6,35,36]. The direction of the bending can be controlled by the ion species and energy, and this effect can even be used to align nanowires. While in some cases the bending effect was explained with amorphization of the material [35], we also observed bending of ZnO nanowires, which cannot be amorphized during ion beam irradiation at RT [37]. The ZnO nanowires can be bent away from the ion beam direction by using small ion energies: the ions only penetrate a small fraction of the nanowire diameter, and only the side of the nanowire facing the ion beam is damaged. The damaged materials has a lower density. The expansion of the material on that side of the nanowire leads to a bending momentum away from the ion beam (details in Ref. [36]). However, when using larger ion energies, where the ion range is comparable to the nanowire diameter, the nanowires are observed to bend *towards* the ion beam, see Fig. 4a. The nanowires were initially straight and perpendicular but after irradiation they are bent towards the ion beam direction. In order to explain this effect, *iradina* has been used to calculate the distribution of damage within the nanowire. When implanting ZnO at room temperature, most of the ion beam induced defects anneal out immediately [37]. We calculated the remaining damage using the difference value of interstitials minus vacancies, assuming annihilation of defects in each simulation cell. This remaining damage is illustrated in Fig. 4b. Excess vacancies remain on the side facing the ion beam, excess interstitials remain at the back side. This inhomogeneous distribution of defects leads to an inhomogeneous expansion of the material. A bending momentum towards the ion beam in induced, as observed in the experiment, Fig. 4a, [36].

4. Conclusion

We have presented the new Monte Carlo code *iradina* for the simulation of ion beam irradiation of nanostructures. In contrast to TRIM-like codes, *iradina* allows arbitrary 3D target structures and can thus account for ion beam irradiation effects peculiar to nanosized objects. For example, *iradina* can explain the lower than expected doping concentrations in ion beam doped nanowires. Furthermore, *iradina* was successfully used to explain the ion beam induced bending of nanowires. Comparisons to bulk simulations using TRIM verify correct basic functionality of the code.

The authors are aware, that some issues like the replacement fraction and accurate sputter yield calculations remain. Furthermore, *iradina* is static at the moment and cannot perform dynamic simulations as TRIDYN [15], but future implementations may include dynamic composition variations. *Iradina* is written in plain C and will be made available online [38] under the terms of the GNU general public license version 3 [39].

Acknowledgements

Special thanks go to Konrad Gärtner for fruitful discussions, and thanks to Steffen Milz, Raphael Niepelt, Sebastian Geburt, Susann Spindler, Martin Gnauck and Andreas Johannes for testing the program. We thank François Schiettekatte for publishing the corteo source code.

References

[1] J.F. Ziegler, J.P. Biersack, U. Littmark, Stopping and Range of Ions in Solids, Pergamon, New York, 1985. Available from: <www.srim.org>.
[2] A.V. Krasheninnikov, K. Nordlund, Ion and electron irradiation-induced effects in nanostructured materials, J. Appl. Phys. 107 (2010) 071301, doi:10.1063/1.3318261.
[3] L. Romano, N.G. Rudawski, M.R. Holzworth, K.S. Jones, S.G. Choi, S.T. Picraux, Nanoscale manipulation of Ge nanowires by ion irradiation, J. Appl. Phys. 106 (2009) 114316, doi:10.1063/1.3267154.
[4] K. Jun, J. Joo, J.M. Jacobson, Focused ion beam-assisted bending of silicon nanowires for complex three dimensional structures, J. Vac. Sci. Tech. B 27 (2009) 3043–3047, doi:10.1116/1.3259919.
[5] V. Tuboltsev, J. Raisanen, Sculpturing nanowires with ion beams, Small 5 (2009) 2687–2691, doi:10.1002/smll.200901424.
[6] C. Borschel, R. Niepelt, S. Geburt, C. Gutsche, I. Regolin, W. Prost, F.-J. Tegude, D. Stichtenoth, D. Schwen, C. Ronning, Alignment of semiconductor nanowires using ion beams, Small 5 (2009) 2576–2580, doi:10.1002/smll.200900562.
[7] C. Ronning, C. Borschel, S. Geburt, R. Niepelt, Ion beam doping of semiconductor nanowires, Mater. Sci. Eng. R 70 (2010) 30–43.
[8] C. Ronning, C. Borschel, S. Geburt, R. Niepelt, S. Müller, D. Stichtenoth, J.-P. Richters, A. Dev, T. Voss, L. Chen, W. Heimbrodt, C. Gutsche, W. Prost, Tailoring the properties of semiconductor nanowires using ion beams, Phys. Status Solidi B 247 (2010) 2329–2337, doi:10.1002/pssb.201046192.
[9] B. Koslowski, S. Strobel, P. Ziemann, Ion polishing of a diamond (100) surface artificially roughened on the nanoscale, Diam. Relat. Mater. 9 (2000) 1159–1163, doi:10.1016/S0925-9635(99)00327-1.
[10] A. Klimmer, P. Ziemann, J. Biskupek, U. Kaiser, M. Flesch, Size-dependent effect of ion bombardment on Au nanoparticles on top of various substrates: thermodynamically dominated capillary forces versus sputtering, Phys. Rev. B 79 (2009) 155427, doi:10.1103/PhysRevB.79.155427.
[11] T.T. Jarvi, J.A. Pakarinen, A. Kuronen, K. Nordlund, Enhanced sputtering from nanoparticles and thin films: size effects, EPL 82 (2). doi:10.1209/0295-5075/82/26002.
[12] J.P. Biersack, L.G. Haggmark, A Monte Carlo computer program for the transport of energetic ions in amorphous targets, Nucl. Instrum. Methods 174 (1980) 257–269.
[13] M. Hautala, Nuclear stopping in polycrystalline materials: range distributions and Doppler-shift attenuation analysis, Phys. Rev. B 30 (9) (1984) 5010–5018, doi:10.1103/PhysRevB.30.5010.
[14] B. Ber, V. Kharlamov, Y. Kudrjavtsev, A. Merkulov, Y. Trushin, E. Zhurkin, Computer simulation of ion sputtering of polyatomic multilayered targets, Nucl. Instrum. Methods B 127/128 (1997) 286.
[15] W. Möller, W. Eckstein, J.P. Biersack, Tridyn binary collision simulation of atomic collisions – and dynamic composition changes in solids, Comput. Phys. Commun. 51 (1988) 335–368.
[16] M. Posselt, A short overview on Monte Carlo simulations of ion beam penetration into amorphous solids, Phys. Status Solidi A 94 (1986) 337–341.
[17] W. Möller, W. Eckstein, Trydyn – a trim simulation code including dynamic composition changes, Nucl. Instrum. Methods B 2 (1984) 814–818.

[18] Hyun-Yong Lee, Hong-Bay Chung, Three-dimensional Monte Carlo calculation of Ga^+ ion penetration in an a-$Se_{75}Ge_{25}$ thin film, J. Appl. Phys. 78 (1995) 5975–5980.

[19] D. Schwen, M. Huang, P. Bellon, R.S. Averback, Molecular dynamics Xe bubble re-solution in UO_2, J. Nucl. Mater. 392 (2009) 35–39, doi:10.1016/j.jnucmat.2009.03.037.

[20] M. Hautala, I. Koponen, Simulations of submicron-scale erosion and ripple formation on ion bombarded solid surfaces, Nucl. Instrum. Methods B 117 (1996) 95–100.

[21] G. Hobler, S. Selberherr, Monte Carlo simulation of ion implantation into two- and three-dimensional structures, IEEE T. Comput. Aid. D. 8 (1989) 450–459, doi:10.1109/43.24873.

[22] D. Li, G. Wang, Y. Chen, L. Lin, G. Shrivastav, S. Oak, A. Tasch, S. Banerjee, B. Obradovic, A computationally efficient simulator for three-dimensional Monte Carlo simulation of ion implantation into complex structures, Nucl. Instrum. Methods B 184 (2001) 500–508.

[23] B.J. Obradovic, G. Balamurugan, G. Wang, Y. Chen, A.F. Tasch, Monte Carlo simulation of ion implantation into topographically complex structures, 1998, pp. 513–516. doi:10.1109/IEDM.1998.746410.

[24] H. Stippel, S. Selberherr, Three dimensional Monte Carlo simulation of ion implantation with Octree based point location, in: Proceedings of VPAD, 1993, pp. 122–123. Available from: <http://in4.iue.tuwien.ac.at/pdfs/vpad1993/pdfs/00724750.pdf>.

[25] A. Burenkov, K. Tietzel, A. Hossinger, J. Lorenz, H. Ryssel, S. Selberherr, A computationally efficient method for three-dimensional simulation of ion implantation, SISPAD 99 (1999) 55–58, doi:10.1109/SISPAD.1999.799258.

[26] H.-B. Kim, G. Hobler, A. Steiger, A. Lugstein, E. Bertagnolli, Full three-dimensional simulation of focused ion beam micro/nanofabrication, Nanotechnology 18 (2007) 245303.

[27] W. Boxleitner, G. Hobler, FIBSIM – dynamic Monte Carlo simulation of compositional and topography changes caused by focused ion beam milling, Nucl. Instrum. Methods B 180 (2001) 125.

[28] D. Kunder, E. Baer, M. Sekowski, P. Pichler, M. Rommel, Simulation of focused ion beam etching by coupling a topography simulator and a Monte-Carlo sputtering yield simulator, Microelectr. Eng. 87 (2010) 1597.

[29] F. Schiettekatte, Fast Monte Carlo for ion beam analysis simulations, Nucl. Instrum. Methods B 266 (2008) 1880–1885.

[30] F. Schiettekatte, Monte Carlo simulation of ion beam analysis spectra using corteo, 2009. Available from: <http://www.lps.umontreal.ca/schiette/uploads/Recherche/Corteo20090220.pdf>.

[31] Q. Yang, D.J. O'Connor, Z. Wang, Empirical formulae for energy loss straggling of ions in matter, Nucl. Instrum. Methods B 61 (1991) 149–155.

[32] B. Yuan, F.C. Yu, S.M. Tang, A database method for binary atomic scattering angle calculation, Nucl. Instrum. Methods B 83 (1993) 413–418.

[33] D. Stichtenoth, K. Wegener, C. Gutsche, I. Regolin, F.J. Tegude, W. Prost, M. Seibt, C. Ronning, P-type doping of GaAs nanowires, Appl. Phys. Lett. 92 (2008) 163107.

[34] A. Colli, A. Fasoli, C. Ronning, S. Pisana, S. Piscanec, A.C. Ferrari, Ion beam doping of silicon nanowires, Nano Lett. 8 (2008) 2188–2193.

[35] E. Pecora, A. Irrera, S. Boninelli, L. Romano, C. Spinella, F. Priolo, Nanoscale amorphization, bending and recrystallization in silicon nanowires, Appl. Phys. A 102 (2011) 13–19, doi:10.1007/s00339-010-6040-2.

[36] C. Borschel, S. Spindler, D. Lerose, A. Bochmann, S.H. Christiansen, S. Nietzsche, M. Oertel, C. Ronning, Permanent bending and alignment of ZnO nanowires, Nanotechnology 22 (2011) 185307, doi:10.1088/0957-4484/22/18/185307.

[37] S.O. Kucheyev, J.S. Williams, C. Jagadish, J. Zou, C. Evans, A.J. Nelson, A.V. Hamza, Ion-beam-produced structural defects in ZnO, Phys. Rev. B 67 (2003) 094115, doi:10.1103/PhysRevB.67.094115.

[38] Iradina, 2011. Available from: <http://www.iradina.de>.

[39] GNU general public license, 2007. URL: <http://www.gnu.org/licenses/gpl.html>.