Development of a multiscale modeling environment for parameterization of ReaxFF using SCC-DFTB and its application to ZnO structures

Ali Rafieefar

Thesis Submitted in Fulfillment of the Requirement for the Award of the Degree of Master of Science in Chemistry for Renewable Energy.

Department of Chemistry - Ångström Laboratory
Uppsala University

APRIL 2014
Declaration

I hereby declare that this master thesis entitled “Development of a multiscale modeling environment for parameterization of ReaxFF using SCC-DFTB and its application to ZnO structures” is the result of my own research except as cited in the references. This thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : 

Student : Ali Rafieefar

Date : April 2014

Supervisor : Peter Broqvist

Subject Specialist : Kersti Hermansson
To my beloved mother and father
Acknowledgments

First and foremost, I would like to express my profound gratitude to my supervisor Dr. Peter Broqvist and my adviser Prof. Kersti Hermansson for their expert guidance, vast knowledge and continuous support during my master thesis. It was only with their great patience and encouragement that I managed to simultaneously pass my master courses and conclude my thesis. I have the utmost respect for them both as distinguished scientists and as the individuals who helped me enter the field of computational chemistry. Besides, I would like to show my deepest appreciation to Dr. Pavlin Mitev and Dr. Daniel Spångberg for their generous help and insightful comments whenever I had a question during my thesis work. My sincere thanks also go to PhD students Matti Hellström, Stefan Huber and everyone in the Teoroo group who benevolently shared with me their immense knowledge and with whom I have had many useful discussions. Furthermore, I am genuinely grateful to Dr. Christer Elvingson who has, since my first day in Uppsala, helped and guided me through my studies. I am also thankful to the natural e-science program, eSSENCE, for granting me the opportunity to attend very useful conferences on e-science that were essential in my understanding of the vast array of IT applications in chemistry. Last but not the least, my heartfelt gratitude goes to my parents Mehran and Ghodsi who have been most selfless in their support and care throughout my studies.

Ali Rafieefar, Uppsala
Abstract

Finding new approaches to scavenge material properties of ZnO as a function of structure, size and shape is important to further improve technological applications based on ZnO nanostructures. In this perspective, a multiscale modeling approach was adopted to theoretically study ZnO using the SCC-DFTB and ReaxFF methods. I developed a Multiscale Simulation Environment to control data flow between the calculation methods and automated several of the computational processes involved. In this environment, I used calculated results for ZnO polymorphs and stable surfaces obtained at the SCC-DFTB level of theory to parameterize ReaxFF and to generate a reactive force-field. The force-field is tested on the ZnO nanostructures and the results of the force-field are taken one step back to validate their consistency with SCC-DFTB. The parameterization is thus made in a cyclic fashion, i.e. if the energetic and structural properties calculated by the ReaxFF are different from those of the SCC-DFTB, the training set is extended with these data and the parameterization process is repeated until convergence is achieved. I then chose the optimal force-field and applied it to study ZnO solid and tubular nanowires of different shapes and sizes. The approach developed in this thesis work shows the validity and efficiency of the coarse-graining scheme and can be applied to other novel materials in the field of solid-state chemistry.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iii</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>iv</td>
</tr>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>List of Figures</td>
<td>4</td>
</tr>
<tr>
<td>List of Tables</td>
<td>5</td>
</tr>
<tr>
<td>List of Appendices</td>
<td>6</td>
</tr>
</tbody>
</table>

## 1 Introduction

1.1 Zinc oxide | 10
1.2 Multiscale modeling

1.2.1 Concepts of multiscale modeling | 15
1.2.2 QM methods | 16
1.2.3 Force-field methods | 18
1.3 Constructing a Multiscale Simulation Environment | 20
1.4 Objective and outline | 21

## 2 Theory and methods

2.1 Methods

2.1.1 SCC-DFTB | 22
2.1.2 ReaxFF | 24
2.1.3 ReaxFF force-field parameterization | 26

2.2 Properties

2.2.1 Murnaghan equation of state | 28
2.2.2 Surface cleavage energies | 29

2.3 Program Modules

2.3.1 Python programming language | 29
2.3.2 Atomistic Simulation Environment (ASE) | 30
3 Development of the Multiscale Simulation Environment

3.1 Project workflow
3.2 MSE overview
3.3 MSE workflow
  3.3.1 Main window
  3.3.2 Import window
  3.3.3 "Generate Training-set" button
  3.3.4 InitialBulks class
  3.3.5 CalculateMain class
  3.3.6 “Calculate Bulk Modulus” button

4 ReaxFF parameterization and validation

4.1 Parameterization
  4.1.1 Step 1: Generation of the training set
  4.1.2 Step 2: Parameterization of ReaxFF
  4.1.3 Step 3: Validation and extension of the training set
4.2 Choosing the optimal force-field
  4.2.1 Bulk structures
  4.2.2 Surface structures
  4.2.3 Nanostructures
  4.2.4 Optimal force-field

5 Results and discussion

5.1 Nanorods with a hexagonal cross-section
5.2 Parallelogram and elongated ZnO nanostructures

6 Concluding remarks and future work

References
List of Figures

1.1 ZnO bulk polymorphs 11
1.2 Top and side view of ZnO non-polar surfaces 12
1.3 Nanotube 3d model with polar and non-polar facets 13
1.4 Schematic illustration of different phenomena at size-time scales 16
1.5 Illustration of the multiscale modeling scheme 20

2.1 Multiscale simulation cloud 31

3.1 Overall project workflow 33
3.2 GUI workflow for training set preparation 35
3.3 GUI Main window screenshot 37
3.4 GUI Import window screenshot 38
3.5 'Calculate Bulk Modulus' button workflow 40

4.1 Total error in the parameterization of ReaxFF 43
4.2 Validation for bulk and surface structures 44
4.3 Validation for nanorod structures 45
4.4 Volume-energy curves for bulk polymorphs 46
4.5 SCC-DFTB SP vs ReaxFF formation energies for nanorods 50

5.1 Illustration of ZnO nanowire shapes studied 52
5.2 $\Delta E_{f}^{FF1}$ vs $1/n$(ZnO) for hexagonal nanorods 53
5.3 Nanotube wall thickness. 53
5.4 $\Delta E_{f}^{FF1}$ vs wall-thickness vs $d$ for nanotubes 54
5.5 Jagged vs smooth $\Delta E_{f}^{FF1}$ solid nanowires 55
5.6 Surface morphology effect for nanorods with the same $d$ 56
5.7 FF1 overstabilization of jagged surfaces 57
5.8 Elongated and Parallelogram shaped nanowires 58
5.9 Bandgap Energies for Nanowires 59
List of Tables

4.1 k-point sampling for ZnO bulks ........................................ 42
4.2 Calculated properties of ZnO bulk polymorphs ..................... 48
4.3 Surface energies at different levels of theory ........................ 49
List of Appendices

A  The MSE Software code documentation  68
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASE</td>
<td>Atomistic Simulation Environment</td>
</tr>
<tr>
<td>Atoms object</td>
<td>An ASE object representing a collection of atoms</td>
</tr>
<tr>
<td>CC</td>
<td>Coupled-Cluster</td>
</tr>
<tr>
<td>CI</td>
<td>Configuration Interaction</td>
</tr>
<tr>
<td>DFT</td>
<td>Density Functional Theory</td>
</tr>
<tr>
<td>DFTB+</td>
<td>The implementation of SCC-DFTB used in this project</td>
</tr>
<tr>
<td>EEM</td>
<td>Electronegativity Equalization Method</td>
</tr>
<tr>
<td>FF0</td>
<td>Initial ReaxFF model used in the parameterization scheme.</td>
</tr>
<tr>
<td>FF1</td>
<td>ReaxFF model optimized for ZnO bulk phases and surfaces</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>HF</td>
<td>Hartree-Fock</td>
</tr>
<tr>
<td>MSE</td>
<td>Multiscale Simulation Environment developed in this project</td>
</tr>
<tr>
<td>OOP</td>
<td>Object oriented programming</td>
</tr>
<tr>
<td>QM</td>
<td>Quantum mechanical calculations</td>
</tr>
<tr>
<td>QM/MM</td>
<td>Quantum mechanics/molecular mechanics</td>
</tr>
<tr>
<td>ReaxFF</td>
<td>The particular type of Reactive force-field used in this thesis</td>
</tr>
<tr>
<td>SCC-DFTB</td>
<td>Self-Consistent-Charge Density Functional Tight-Binding method</td>
</tr>
<tr>
<td>SCC-DFTB-Opt</td>
<td>SCC-DFTB calculation with optimization of lattice and/or atomic positions</td>
</tr>
<tr>
<td>SCC-DFTB-SP</td>
<td>SCC-DFTB single point calculation of ReaxFF optimized structure</td>
</tr>
<tr>
<td>SK-files</td>
<td>Files containing the Slater-Koster tables and corresponding repulsion potential for DFTB+ calculations</td>
</tr>
<tr>
<td>UHF</td>
<td>Unrestricted Hartree-Fock</td>
</tr>
<tr>
<td>znopt</td>
<td>The particular SCC-DFTB parameter set used in this thesis</td>
</tr>
</tbody>
</table>
Zinc oxide (ZnO) nanomaterials have applications in a wide array of technologies such as catalysis, electronics and solar cells. Understanding the behavior of these materials at the microscopic level is essential to improving existing and finding new applications. Here, to establish structure-property relationships, computational methods can be useful. In computational chemistry, however, there is always a trade-off between size of the system and the accuracy of the method used. Highly accurate atomistic simulations elucidate chemical phenomena to a greater detail and at a higher accuracy, but with the power of current computers, such accurate methods can rarely be used to calculate large systems such as nanostructures with thousands of atoms. One solution to this problem is to perform detailed calculations on smaller ZnO systems and via parameterization, transfer the obtained information to a less detailed (a.k.a. coarser) method to describe larger nanosystems. Parameterization is the pathway to transfer and retain as many of the features offered by the more accurate method to the method higher in the hierarchy of size and time. In this way, the nanosystem properties are calculated with the accuracy of the more detailed method but at a lower computational cost. This strategy is called multiscale modeling, which aims to bridge the gap between computational methods at different size and time scales.

Implementing multiscale modeling requires performing calculations and proper transferring of data among calculation methods at different size and time dimensions. This includes performing tasks such as preparing input for the more detailed method, running the calculation and collecting and filtering the results, creating input for the coarser method in an iterative manner. These tasks are generally time-consuming and error-prone when done without the help of automated tools and a software environment.

In this work, systematic and automatic tools for the parameterization process to establish a theoretical framework that increases the throughput of computational studies in materials science have been developed. I have developed software for a Multiscale Simulation Environment (MSE) which provides an interactive interface through which the settings of different calculations can be defined. Two computational methods are integrated in the developed MSE. One method explicitly treats electrons (more detailed and more expensive method) whereas, the other method treats the atoms directly (less detailed method and less expensive).

The MSE handles the calculations for the more detailed method and uses results for generating the information necessary (the training set) for the parameterization of the coarser method. The result of the parameterization is the chemical information formulated into sets of mathematical expressions (a force-field). In the MSE we seek to generate a force-field capable of performing theoretical calculations at the same level
of accuracy as the more detailed method (or reference method) used to parameterize it. To test this, we validate the force-field by applying it on the structures which were not present in the training set used in the force-field parameterization. If the force-field cannot reproduce the results for such structures, the training set is expanded to include these structures and the parameterization process is repeated until a one-to-one correspondence in energy and structure between the two methods is achieved.

In the scope of this project, we used the MSE (Multiscale Simulation Environment) to generate a force-field to study ZnO nanostructures in terms of how the shape of their surfaces affects their stability. By utilizing the MSE, we have, to a great extent, managed to maintain the accuracy associated with a high-level computational method to the coarser force-field model for ZnO nanostructures. This strategy allowed for an order of magnitude larger systems and sizes to be treated. Thus, the fast and seamless automation environment can be deemed to have reached its goal within the scope of the project.
Chapter 1

Introduction

1.1 Zinc oxide

ZnO is a semiconductor material used in many technologically important applications such as heterogeneous catalysis, micro-electronics, gas sensors, electrochemistry, solar cells and blue/UV lasers. One reason for this wide usage is its versatile electronic properties which are strongly intertwined with microscopic structural characteristics. At ambient conditions, ZnO crystallizes in the wurtzite structure with a wide direct band gap of ~3.37 eV. However, several other polymorphs also exist and at the nano scale, ZnO structures can be synthesized in various sizes and shapes, e.g. as nanoparticles, nanowires and nanotubes (hollow nanowires), which allows for tuning of desired properties for use in specific applications. An illustrative example where tailored ZnO nanosystems have been used is for dye-sensitized solar cells (DSCs). The DSCs are inexpensive and effective solar cells based on thin-film technology. The efficiency of DSCs is highly dependent on the physical and chemical properties of the semiconductor used as an anode. Law et al. (2005) replaced the typically used nanoparticles in the DSC with an array of ZnO nanowires. Compared with the corresponding nanoparticle-based solar cells, the use of ZnO nanowires lowered the electron/hole recombination rate and simultaneously allowed the material to absorb light in a broader spectrum. To further improve such applications and also find new ones, a thorough understanding of structure-property relationships for ZnO nanostructures needs to be established.

To study structure-property relationships, experiments alone are often not sufficient to capture the very details needed. Here, computational chemistry and physics will play an important role. Computer calculations have for long been instrumental in the prediction of new bulk crystalline compounds and in the modeling of phase transitions by studying global energy landscapes. In addition to thermodynamics, theoretical studies of different phases can assess mechanical stability, e.g. via their
elastic properties, and dynamical stability, e.g. via investigating phonon band structures.\textsuperscript{7} In this way, structural properties can be linked with energetic and electronic properties and guide experimental scientists in their search for novel material solutions for emerging technologies.

In nature, four different crystal phases of ZnO have been successfully synthesized. These are the hexagonal wurtzite structure, the closely-related zincblende\textsuperscript{*} structure,\textsuperscript{9} the cubic rocksalt structure and a graphitic structure.\textsuperscript{10, 11} Using theoretical studies, three additional structures have been studied in the literature, namely the 8-fold coordinated CsCl,\textsuperscript{10, 12} body-centered tetragonal (BCT)\textsuperscript{6, 13} and the cubane\textsuperscript{7} (Figure 1.1 shows unit cells for these bulk polymorphs). Apart from rocksalt and the CsCl-type, all other ZnO crystal types mentioned are direct band gap materials.\textsuperscript{14}

The wurtzite crystal structure is, as mentioned above, the most thermodynamically stable ZnO bulk structure at ambient conditions and is consequently also the one mostly discussed in the literature.\textsuperscript{8} This is also the crystal structure mostly discussed in the literature when it comes to surfaces and nanosystems of ZnO. The wurtzite crystal structure belongs to a hexagonal space group where Zn and O ions form corrugated hexagonal honeycomb sheets with three equivalent Zn-O bonds at each atomic site (see Figure 1.2). In these sheets, all Zn ions occupy one side while all O ions the other side. Therefore, there is a dipole moment normal to each of these sheets (see Figure 1.1). The crystal structure is built by stacking alternating layers of complementary Zn and O sided sheets, resulting in 4-coordinated Zn and O atoms in the bulk, as shown in Figure 1.1.

\textsuperscript{*}The name zincblende is originally used for hexagonal ZnS compounds\textsuperscript{8} and the correct term in the context of ZnO should be sphalerite. However, in order to be consistent with other literature, the term zincblende is used in this text.
Cleaving the wurtzite bulk crystal in different directions, results in the formation of various surfaces with different characteristics and structures. If we cleave the bulk along the direction of the above-mentioned stacking (c axis), polar ±(0001) surfaces with high surface energies are formed. The polar surfaces have a dipole moment perpendicular to the surface (in analogy to what was mentioned above) and hence are either Zn-terminated (0001) or O-terminated (0001), each having distinct physical and chemical properties. In the process of ZnO nucleation under natural and experimental conditions, precursor molecules are adsorbed on polar surfaces in order to minimize the surface energy. This results in the formation of another polar surface with an inverted polarity compared to the original one, i.e. if the original surface was Zn terminated it becomes O terminated and vice versa. Hence, the destabilizing dipole-moment in these surfaces, becomes larger as the number of layers increases. Many experimental and theoretical studies have been undertaken to understand the reconstruction mechanisms that stabilize these surfaces. Figure 1.3(a) shows an STM image under ultra high vacuum in which the formation of triangularly shaped
pits is seen to have stabilized the zinc-terminated (0001) surface.\textsuperscript{20,22}

\textbf{Figure 1.3} – (a) An STM image of a polar (0001) ZnO surface with triangular islands and holes. Image taken from Parker et al. (1998).\textsuperscript{22} (b) Nanotube wurtzite structure model of zinc oxide with stable non-polar outer and inner facets and and growth direction of polar [0001].\textsuperscript{23} (c) A SEM image of a hollow nanotube experimentally grown on an Au substrate. Image taken from Taghavi et al. (2013).\textsuperscript{23}

ZnO \textit{non-polar} surfaces, such as (1010) and (11\(\bar{2}\)0), are formed experimentally when the wurtzite crystal is cleaved in the direction perpendicular to the \textit{c} axis.\textsuperscript{23} These surfaces have the same number of zinc and oxygen ions on each layer along the growth direction. From an ionic point of view, Zn-O dimers in the non-polar surfaces, are charge neutral.\textsuperscript{16} Hence, ZnO non-polar surfaces, are energetically more stable. Indeed, ZnO growth along the \textit{c} axis is the basic mechanism in which one-dimensional ZnO nanowires and nanotubes are fabricated in solution.\textsuperscript{16} The walls of these nanosystems thus constitute non-polar surfaces [see Figure 1.3(b)].\textsuperscript{23} Hence, ZnO nanostructures, including nanoparticles and one-dimensional nanowires, have higher surface to volume ratio than extended ZnO surfaces. This is an important feature in many of the ZnO technological applications, e.g. heterogeneous catalysis.\textsuperscript{18} Thus, understanding the microscopic properties that govern the physics and chemistry of ZnO surfaces is of great significance not only for the fabrication of ZnO nanostructures but also for the study of catalytic reactions of extended ZnO surfaces.

ZnO \textit{nanowires} were first synthesized by Meulenkamp et al. in 1998.\textsuperscript{24} Among many applications, they are also used as electrodes in photoelectrochemical water splitting to generate hydrogen where they both improve charge carrier transport and lower the electron-hole recombination rate compared to their bulk and nanoparticle counter-
The size of nanowires, however, affects properties such as light absorption and band-gap energy. These size-dependent properties combined with the large surface area of the nanorods are important in such applications as water-splitting, gas sensing and light-emitting diodes (LEDs). These examples illustrate the significance of studying nanowires of different sizes and shapes.

Vayssieres et al. (2001) wrote one of the first papers on the formation of ZnO with tubular nanotube geometry. Such structures, due to their inner walls formed by non-polar surfaces, have higher surface area in comparison to the nanowires. Compared to their solid, non-hollow counterparts, hollow nanotubes show higher porosity, better charge transport and lower mass density, properties that give enhanced functionality to these structures in applications such as photocatalytic reactions and optoelectronic devices. Hence, similar to nanowires, it is important to investigate the stability dependence of nanotubes on their size. Li et al. (2005) studied size dependence of nanorod structures and observed that nanorods with a diameter of 40 nm or more tend to form hollow nanotubes and those with a smaller diameter display solid nanowire formation growth in the polar direction [see Figure 1.3(c)]. Elias et al. proposed an experimental method to convert nanowires to nanotubes with control over their lengths and wall thicknesses. Experimentally, ZnO nanorods are usually in the order of hundreds of nanometers in thickness and length [see Figure 1.3(c)].

1.2 Multiscale modeling

As stated in the previous section, the hexagonal wurtzite nanorods (used here to collectively refer to tubular and solid nanowires) can be constructed by cleaving the surfaces from the bulk crystal in a particular direction. Hence, in order to calculate properties of these 1D-nanorods, an accurate description of the surface structures as well as of the bulk wurtzite is needed. In computational chemistry, however, there is always a trade-off between size of the system and the accuracy of the method used. Thus, despite their accuracy and transferability, high-level methods (such as first-principles methods) are too expensive for nanorods of sizes compatible with experiments. One solution to this problem is to perform detailed calculations on the smaller ZnO bulk and surface systems and then transfer the obtained information to a coarser, less detailed method to describe larger nanorods. In this fashion, one could say that the nanosystems properties would be calculated with the accuracy of the more detailed method but at a lower cost. We call this approach multiscale modeling (a concept which generally has many definitions) and it is discussed in the following sections. In Section 1.2.1 we give an overview of different ways to approach the problem, and in Section 1.2.2 a more detailed account of how we approached the problem in this project is provided.
1.2.1 Concepts of multiscale modeling

Only a hundred years ago, scientific explanations of physical phenomena were mainly based on classical Newtonian physics. This could perfectly explain macroscopic events but failed to provide sufficient explanation for processes at the sub-atomic level. Quantum physics led to a different understanding of particle energies as non-continuous quantized values and the subsequent advent of quantum chemistry focused on the development of theoretical tools to treat and describe chemical characteristics of systems at microscopic levels, such as chemical bonding and reactions. Thus, while in classical mechanics, properties of large systems are the matter of interest, electrons (and to some extent nuclei) are the particles modeled in quantum mechanical theory.

Theoretical calculations for a chemical system based on classical mechanics need less computational resources than quantum-mechanical ones. Hence, such calculations can be applied to large systems of particles. First-principles methods based on quantum chemistry, are more complex and have comparatively more degrees of freedom. Therefore, considering that computational resources will be limited in a foreseeable time, such models can hardly be used for large systems composed of thousands of atoms/molecules.

Many of the scientific questions that have attracted the computational chemists in the past years, require an understanding of chemical phenomena at various size and time scales. **Multiscale modeling** aims to bridge the gap between the scientific levels that address these phenomena and thereby endeavors to provide tools and methods to model large systems in long time ranges with the accuracy of for example, the first-principles quantum mechanical (QM) level. Quantum mechanics/molecular mechanics (QM/MM)\textsuperscript{33} is one such approach in which a key region of the system is modeled using very detailed but computationally costly first-principles methods while the rest of the system is described with less detailed methods (e.g. semi-empirical methods or force-fields). Bromley et al.\textsuperscript{34} used QM/MM methodology to study Cu/ZnO surface catalytic reactions. However, the QM/MM method has so far been predominantly used in the modeling of large biological systems.

Another approach to multiscale modeling used in solid-state chemistry is **coarse-graining** meaning that more detailed but computationally expensive QM calculations are used to parameterize cheaper atomistic models or even continuum mechanics models. Figure 1.4 shows how different complex chemical phenomena in materials science can be placed in the size-time ladder and categorized into micro, meso and macro scale systems. In the micro region, microscopic phenomena such as the electronic structure of the chemical system, elementary steps in a catalytic reaction or the role of pressure on phase transitions of the system are studied. These phenomena are usually performed using high-level methods. In the mesoscopic region, phenomena such as surface reconstruction or nanorod dynamics are studied using parameterized
semi-empirical or force-field methods. Macroscopic phenomena, such as heat and mass transfer need be simulated with even coarser methods at larger sizes and longer times. It should be mentioned that although, for example, reactive force-field methods describe structures and phenomena in the lower meso-scale region, they are usually used for the nano-sized systems but can in principle be used for very small systems as well. Following the discussion in Section 1.1, in the case of ZnO nanowire simulations, we are interested in the upper micro and lower meso regions as depicted in Figure 1.4. In the following, an overview of our computational strategy for ZnO calculations is provided.

![Figure 1.4](image)

**Figure 1.4** – Schematic illustration showing examples of some chemical phenomena encountered in ZnO studies that can be calculated at different size and temporal scales. In multiscale modeling, structural and energetic information is transferred from more detailed atomistic levels (QM, semi-empirical etc.) to more coarse-grained methods (force-field simulations, statistical mechanical simulations, etc.) in the size-time ladder so that the latter methods can describe chemical systems at an accuracy compatible with atomistic method calculations. Adapted from Broqvist (2005)\(^{35}\) and Schlög et al. (2001).\(^{36}\)

### 1.2.2 QM methods

In Figure 1.4, lowest in time and size scale are the first-principles quantum mechanical calculations. These methods use laws of quantum mechanics to solve the Schrödinger equation and give properties of a chemical system. In the Hartree-Fock (HF) method, for example, a linear combination of a finite set of atomic (spin-)orbitals (basis functions
in HF) is used to find a solution to the Schrödinger equation described within the Born-Oppenheimer approximation. By using Slater determinants, antisymmetry of the wave function is warranted. Here, the true energy of the system and the energy eigenvalue is gained by putting the wave function into the energy expression and minimizing the energy with respect to changes in the (spin-)orbitals. HF lies in the micro region in the size-time ladder illustrated in Figure 1.4 and has been used, e.g. to study phase changes in ZnO at high pressure.

In the HF method however, each electron is in practice exposed to an average field of all other electrons and thus electrons come too close to each other, i.e. the electron-electron correlation interaction is not accounted for. Electron correlation contribution to the energy might be less than 1% of the total energy, but still it has a great impact on the calculations. For instance, due to neglect of electron correlation, HF tends to overestimate bandgaps compared to experiment.

Post-HF ab initio methods are able to include electron correlation in the calculation of the total energy of a system. Configuration-Interaction (CI) is one such method in which electron motion is correlated by inclusion of small amounts of excited-state electronic configurations into the ground state HF reference. This gives electrons more mobility and lowers the energy by inclusion of electron correlation. Möller–Plesset perturbation (MPn with n being the order of perturbation expansion) theory is another post-HF method which includes perturbation contributions to the HF energy and thus includes electron-correlation in the total energy calculation. In Coupled-Cluster (CC) methods, in contrast to MPn which include corrections to the reference HF wave function to a given order, all corrections of a given type to infinite order are included. CC is usually accurate, e.g. CCSD(T) is the “gold standard” in calculations with high quality results that can be used as a yardstick to see how accurate new calculation methods are.

The above mentioned QM methods are, however, based on wave functions that are described by atomic orbitals. In density functional theory (DFT) it has been shown that the energy of any system of atoms can be derived from its electron density. The electron density depends on the three spatial coordinates whereas, an N-electron wave function depends on 3N spatial coordinates and N spin coordinates. This makes DFT a computationally cheaper ab initio method in which electron correlation is implicitly accounted for through the electron density. The biggest challenge (still unsolved) in DFT, however, is to find a universal functional for the exchange correlation energy.

Ab-initio electron structure calculations afford a thorough framework to un-

\[ ^{†} \] In Unrestricted-HF(UHF), electron spin is included. Therefore, electron-electron interaction is accounted for but only to a small extent.

\[ ^{‡} \] CCSD(T) is truncated coupled cluster which includes all S (singles) and D (doubles) type corrections and an estimate to the connected T (triple excitations) contribution is derived from a less expensive (non-iterative) MP4 calculation.
derstand chemistry at the electronic level. They offer a solid base to derive exact
energies, predict reaction paths and describe ground-state structures of molecules and
materials. However, this chemical versatility comes at a high computational cost and
even for small molecules, millions of integral calculations might be necessary. There-
fore, these methods cannot be used for systems larger than a few hundreds of atoms
or (for the cheaper QM methods) for times longer than tens of picoseconds. One way
to circumvent this limitation is to use semi-empirical methods that can describe in-
teractions between particles with less complexity through making various additional
approximations. Such parameterized methods rely on higher-level QM calculations or
experimental data as their reference. Parameterization is aimed to transfer and retain
as many of the features offered by the more accurate method (here an ab-initio method)
to the method higher up in the size and time hierarchy (semi-empirical methods, see
Figure 1.4). Semi-empirical calculations therefore, commonly result in a significant re-
duction in calculation costs but still with a direct treatment of electrons.43

The Self-Consistent-Charge Density Functional Tight-Binding method (SCC-
DFTB)44 is one such method. SCC-DFTB is an approximation to DFT methods and
has increasingly gained prevalence in computational chemistry to study biological sys-
tems45 and inorganic materials.46 SCC-DFTB does not have to be fitted to empirical
data and is typically parameterized solely based on ab-initio DFT calculations. SCC-
DFTB, treats electrons explicitly, e.g. models band structure. However, since it is an
approximate method and thus needs less computer resources than DFT, it can be used
for larger systems and at longer time scales.44 In our strategy to formulate a multiscale
modeling solution to the ZnO problem discussed in Section 1.1, we started the cal-
culations of ZnO bulk polymorphs and surfaces with SCC-DFTB. However, when the
system size is increased even further (as in the case of ZnO nanorods), a direct treat-
ment of the electronic structure will be computationally burdensome. One approach to
resolve this issue is an implicit description of electrons, thereby considering the atoms,
and not the electrons, as the smallest units in the system under investigation (atomistic
model). Interatomic interactions can then be formulated into sets of mathematical ex-
pressions or force-fields. This approach results in calculations substantially faster than
QM and semi-empirical methods.

1.2.3 Force-field methods

Force-fields express interactions between the particles (typically atoms) with an array of
potential functions. Bonded interactions like bond-stretching and compression and non-
bonded charge interactions are also modeled. The simplicity of the analytical formulas
for the potential functions allows for calculations on systems exponentially larger than
QM methods. In a “QM-based force-field”, the parameters are fitted to a suite of
energies and geometries (or a training set) from more accurate calculation methods. A
force-field’s application and transferability is constrained by the configurations used in
its training set. Accuracy and reliability of force-fields is thus limited to the training set used in their parameterization.

The atomistic interaction energy for an N-body system based on particle co-ordinates \((r_1, \ldots, r_n)\) can be expressed as follows:

\[
\phi_{\text{tot}} = \sum_{i=1}^{N} \sum_{j>i}^{N} \phi_2(r_i, r_j) + \sum_{i=1}^{N} \sum_{j>i}^{N} \sum_{k>j}^{N} \phi_3(r_i, r_j, r_k) + \ldots \tag{1.1}
\]

Ideally there would exist a suite of analytical functions that can calculate the total interaction \((\phi_{\text{tot}})\) for any set of coordinates for any type of chemical system. However, such functions have not been found and therefore a system-specific functional form has to be chosen. The aim should thus be set to define a proper functional form that describes all the significant configurations for the system and problem at hand, e.g. condensed bulk phases and surface morphologies. In the simplest case, for a system consisting of two elements, only low order pairwise interactions are considered (neglecting \(\phi_3 + \ldots\)). For example in the case of ZnO systems, O-O, Zn-O and Zn-Zn two-body potentials comprise \(\phi_{\text{tot}}\). However, the two-body potentials are usually affected by non-pairwise contributions from other particles that are bonded to these atoms (a.k.a. many-body problem). To solve this problem, one might use more terms in Eq. (1.1) which exponentially increases the types of interactions that should be calculated, e.g. adding 3-body, 4-body terms. However, a common approach to solve the many-body problem is to instead use effective pair-potentials. This means to fit the two-body potentials to a training set that includes many-body interactions. Although in this approach, the two-body terms are no more representative of the interaction energy between the two atoms, but applying the effective pair-potentials limits the number of energy terms in Eq. (1.1)\(^\text{47,48}\).

In this thesis work the reactive force-field family ReaxFF\(^\text{49}\) has been used. ReaxFF has an adjustable set of parameters that upon fitting to experimental data, ab-initio calculations or SCC-DFTB results, are hoped to reproduce energies, forces and geometries of the reference method (training set). Atomic charges in ReaxFF are described via the Electronegativity Equalization Method (EEM), the bonded interactions of atoms are considered bond order dependent and non-bonded interactions (van der Waals and Coulomb interactions) are treated between each pair of particles. It also accounts for polarization effects by utilizing a geometry dependent charge calculation scheme.

ReaxFF can model bond breaking and formation, charge transfer between the atoms and different types of chemical bonds, including covalent, metallic and ionic ones. ReaxFF can perform calculations on large ZnO nanorods with an accuracy of its training set and at a much lower computer cost compared to the QM or semi-empirical
methods used in its parameterization.

**Figure 1.5** – (a) Illustration of an atomistic simulation environment that can embed calculations methods across temporal and dimensional scales. (b) Schematic illustration of hierarchy of computational methods in this project in their operable size and temporal scales. Multiscale simulation environment allows for interaction between the methods to occur in a semi-automatic and seamless fashion.

### 1.3 Constructing a Multiscale Simulation Environment

Implementing multiscale modeling requires calculations and proper transfer of data between the different calculation methods. This includes tasks such as preparing input for the more detailed method, running the calculation and collecting and filtering the results, creating input for the coarser method and so on. These tasks are generally time-consuming and error-prone when performed without the help of automated tools. One of the main parts of the present thesis work was to build a framework to facilitate construction and validation of ReaxFF force-fields to be used in subsequent simulations of ZnO nano-systems. Figure 1.5(a) summarizes schematically our strategy for this framework and we call it a Multiscale Simulation Environment (henceforth abbreviated as MSE). The circles represent tasks that are to be done outside this environment and the two-way arrows imply interaction of these task with the environment. Thus, in our MSE developed in this thesis, data obtained from a semi-empirical QM calculation is read into the environment, transferred to a parameterization process which occurs outside the environment using an external software. The optimal parameter set is later used in the environment to perform larger calculations using a force-field method.

In this project, we use the MSE to create a force-field to study ZnO nanostructures. The computational methods we use are schematically illustrated in Figure 1.5(b). Here, the MSE is used to automatically create a training set to fit ReaxFF parameters based on SCC-DFTB data [parameterization arrow in Figure 1.5(b)]. Thereafter, the MSE uses the so acquired force-field to calculate energies for a test suite of ZnO...
nanowires and nanotubes not included in the training set and compare with SCC-DFTB data to check the transferability of the created force-field [feedback arrow in Figure 1.5(b)]. If ReaxFF derived properties mismatch those of SCC-DFTB, the structures are automatically added to the training set and the parameterization/feedback cycle is repeated until energies and geometries of the two atomistic methods are in satisfactory accord. Using this strategy, we generate an optimal force-field which is used to calculate ZnO nanostructures of different shapes and sizes not accessible for with the underlying QM-method.

1.4 Objective and outline

The objective of the thesis work presented here is two-fold, i.e. 1) to build the MSE and 2) to find structure-stability relationships in ZnO nanostructures. Here, we use the outlined multiscale simulation strategy based on coarse-graining to reach system sizes of experimental relevance [see Figure 1.5(b)].

The outline of the project is as follows: the different methods used in the development of the MSE will be discussed in detail in Chapter 2; in Chapter 3, a detailed description of how we approach the multiscale modeling problem including explaining the workflow of the project and the implementation and development of the MSE is given; in Chapter 4 we will discuss one cycle of ReaxFF parameterization based on our multiscale modeling approach. Here we discuss how the initial, parameterized and re-optimized force-fields perform in calculations for ZnO bulk polymorphs, non-polar surfaces and ZnO nanorods; computational details, specifications and calculation results of applying the program on ZnO nanostructures are discussed in Chapter 5; and in final in Chapter 6, we conclude the thesis.
Chapter 2

Theory and methods

The theoretical methods used in this project are briefly described in Section 2.1. Properties calculated and modules utilized are discussed in Section 2.2 and in Section 2.3, respectively.

2.1 Methods

In Section 2.1.1, we briefly discuss the theory behind SCC-DFTB, the components in its total energy term followed by modules used for SCC-DFTB calculations. Section 2.1.2 deals with the theory of ReaxFF and specifics regarding the general procedure of parameterization used in this study.

2.1.1 SCC-DFTB

All SCC-DFTB calculations done in this thesis work were aimed at generation and validation of force-fields for ReaxFF. Detailed description of the SCC-DFTB method can be found elsewhere. Here only a brief description is given. Ab-initio calculations can, in principle, produce highly accurate results. However, considering the large computational expense of these calculations, only relatively small systems can be studied. For large scale MD simulations over long time, parameterized simulation models that are approximations to QM methods become necessary.

Tight-binding methods are traditionally used to calculate electronic band structures. SCC-DFTB, however, is not limited to this and can be used to calculate, e.g. total energies. Being an approximation to Kohn-Sham DFT, SCC-DFTB is computationally faster than DFT and given the right parameter set, chemically more
accurate than the force-field methods. SCC-DFTB is therefore used for larger systems and at longer time scales than DFT, but still directly treats the electrons.

In SCC-DFTB the total energy expression, $E_{tot}^{SCC-DFTB}$, is derived from second order-expansion of DFT energy:

$$E_{tot}^{SCC-DFTB} = E_{elec} + E_{rep} = E_{BS} + E_{SCC} + E_{rep}$$ (2.1)

where, $E_{elec}$ denotes the electronic and $E_{rep}$ the repulsive energy term, and where $E_{elec}$ includes the zero order ($E_{BS}$) as well as the second order ($E_{SCC}$) terms. The zero order, a.k.a. the “band structure term”, is the summation of energy over all occupied eigenstates represented by a linear combination of atomic orbitals (LCAO).

$$E_{BS} = \sum_{i} n_i \varepsilon_i = \sum_{i} n_i \langle \Psi_i | H_0 | \Psi_i \rangle$$ (2.2)

In Eq. (2.2), $n_i$ denotes occupation, $\varepsilon_i$ is the orbital energy for each of the occupied electronic eigenstates signified by $\Psi$ and the basis-set used for these calculations is minimal, i.e. for $s$, $p$ and $d$ angular momentum states, one, three and five radial functions are used, respectively. The diatomic distance-related Hamiltonian matrix elements are calculated by DFT and stored in SCC-DFTB parameter files as so-called Slater-Koster tables.

The SCC-DFTB method treats the electronic wave function of the system as a linear combination of atomic orbitals. Normal DFTB does not treat charges self-consistently (i.e. starting with an arbitrary guess and performing iterative minimization until fluctuations in the charge densities will not affect the total energy beyond a defined criterion) and thus excludes the $E_{SCC}$ term [see Eq. (2.1)]. In SCC-DFTB which is used throughout this project, the $E_{SCC}$ energy term derived from charge density fluctuations is included in the total energy expression.

The $E_{SCC}$ term is calculated via expansion of the energy corresponding to the Coulomb interaction in a Taylor expansion to second order thereby taking charge fluctuations into account:

$$E_{SCC} = \frac{1}{2} \sum_{AB}^{atoms} \gamma_{AB} \Delta q_A \Delta q_B$$ (2.3)

where, $\gamma_{AB}$ is the distance-dependent, mainly Coulomb charge-charge interaction functional which is directly proportional to the Hubbard U parameter (=twice the chemical
hardness). The variables $\triangle q_A$ and $\triangle q_B$ are induced Coulombic charges on atoms $A$ and $B$ respectively, and are derived from Mulliken population analysis.\textsuperscript{52} The charge term is essential to describe charge transfer phenomena in the system.

Instead of the usual nuclear repulsion energy, atom-atom pair repulsive potentials are used in SCC-DFTB. In $E_{\text{elec}}$, the interactions between atoms are only attractive and therefore they become too close to each other. The $E_{\text{rep}}$ term corrects for this deficiency. SCC-DFTB uses DFT-pre-calculated atom pair potentials based on inter-atomic distances obtained for a small reference set. Along with Slater-Koster tables, the pair potentials are stored in SCC-DFTB parameter files. Hence, $E_{\text{rep}}$ is an approximation to the difference between the more exact energy calculated by DFT and the electronic energy, $E_{\text{elec}}$:

$$E_{\text{rep}} = E_{\text{DFT}} - E_{\text{elec}} \quad (2.4)$$

By making use of these approximations and proper parameterizations, SCC-DFTB is hoped to maintain the accuracy of full DFT while being significantly faster and computationally less demanding.

In this project, we used the results from SCC-DFTB calculations for various ZnO structures in our parameterization scheme for ReaxFF. The SCC-DFTB calculations were performed with the DFTB+ quantum mechanical simulation package. The Slater-Koster tables for zinc and oxygen were taken from Moreira et al.\textsuperscript{53} with a zinc-oxygen repulsive potential from Hellström et al.\textsuperscript{14} called znopt. This combination has been shown to give good results for different ZnO polymorphs previously.\textsuperscript{14} Furthermore, we used a conjugate gradient driver to move the atoms in the structures to minimize the energy. The Fermi-smearing temperature was set to 0 Kelvin for all SCC-DFTB calculations.

### 2.1.2 ReaxFF

Non-reactive force-fields typically describe bonded interactions using a simple harmonic approximation. They cannot, however, be trained to accurately describe bond-energies at bond-dissociation distances. Introduction of interatomic distance-dependent bond-orders allows for force-field methods to make connectivity-dependent-interactions (e.g. valence and torsion angles), bond-order-dependent in energy calculations. In reactive force-fields such as ReaxFF, as the bond length increases (dissociation occurs), contributions from bond-associated interactions will smoothly go to zero in energy calculations. Non-bonded energy contributions (e.g. van der Waals and Coulomb interactions) are described in ReaxFF at all interatomic distances except for very short distances where an excessive repulsive energy is avoided by the introduction of a shielding term for the
charge-charge interactions. ReaxFF allows for reactive molecular dynamics simulations on systems of thousands of atoms. However, due to the inclusion of extra energy terms, it is 10 to 50 times slower than typical non-reactive force-fields. Nevertheless, ReaxFF constitutes a useful force-field family for simulations of the physics and chemistry of materials at temporal and spatial scales significantly beyond that of quantum chemical calculations, including SCC-DFTB.\textsuperscript{10}

The general principles used in ReaxFF to try to ensure that the potentials are transferable and can be applied to various configurations of a chemical system and their reactions are:

- Discontinuities in forces and energies must not exist, even during reactions.
- Irrespective of the environment (organic/inorganic), the force-field parameters for each atom type are unique (thus enabling application of the force-field to interfaces of different materials).
- ReaxFF does not necessitate definition of reaction coordinates, and potential functions should automatically manage to discover when/where reactions occur in the system.\textsuperscript{47,48,54}

These principles are achieved in ReaxFF by assuming that bond-orders can be calculated via pairwise interatomic distances (bond-order/bond-energy relation). This requires a definition of bond-orders for each atom prior to the calculation of interaction energies. Hence, the bond-order term for each atom is included in the ReaxFF potential expression. For each atom, the bond-order energy is defined as the summation of potential functions of the pairwise interatomic distances between that atom and all other atoms present in the system. It should be mentioned that interatomic distances of more than a few Ångströms are in practice not of significance in the bond-order potential function. This continuous function is treated independently for each pair of chemical elements in the parameterization process. Despite different bond-order functions for each element, it is assumed that every element has an optimal or preferred bond-order.

Treating bond-order/bond-distance provides a solution to the many-body problem by incorporating 2, 3 and 4-body interatomic interaction terms. This means that two-body terms in bond-order potentials are both a function of distances between the atoms as well as the local number density in the vicinity of each of the atoms. Valence angles (3-body) and torsion angle (4-body) terms are also bond-order dependent so their contribution approaches zero upon bond breaking. ReaxFF is an atomistic method, i.e. atoms are considered to be the smallest unit in the system and no distinction between the intermolecular and intramolecular atomic interactions is drawn. The total energy in ReaxFF is divided into several terms as shown in Eq. (2.5):
\[ E_{tot}^{ReaxFF} = E_{bond} + E_{lp} + E_{over} + E_{under} + E_{val} + E_{pen} + E_{tors} + E_{conj} + E_{vdW_aals} + E_{coulomb} \]

(2.5)

in which the partitioned energy contributions include bond energy, \( E_{bond} \), described above; lone-pair energy, \( E_{lp} \), from which the number of lone electron pairs for each atom is calculated (by calculating total number of valence electrons and the sum of atom’s bond-orders); over-coordination penalty energy, \( E_{over} \), which is inversely proportional to the number of broken-up lone pairs around an atom; under-coordination penalty energy, \( E_{under} \), given by the energy contribution of \( \pi \)-bonding between interacting under-coordinated atom pairs; valence angle energies, \( E_{val} \), which smoothly converges to zero as the bond order is decreased as a result of, for instance, increased \( \pi \)-bond character based on the geometry of the central atom in the 3-body interatomic distance; penalty energy, \( E_{pen} \), an additional penalty in energy for systems with two double bonds with an atom in the valency angle; torsion angle energies, \( E_{tors} \), for the 4-body torsion angle which like \( E_{val} \) inclines towards zero as bond-order approaches zero; and conjugation energy, \( E_{conj} \), which expresses energy contribution from conjugated molecules.\(^{47, 48, 54, 55, 56}\)

The coulomb energy term, \( E_{coulomb} \), is calculated using the Electron Equilibrium Method (EEM) in which charge on each atom depends on the system structure. Considering that chemical reactions change the structure of the system, this feature allows for charge calculations for varying time/geometries as occur in chemical reactions. \( E_{vdW_aals} \) constitutes other non-bonded interactions (long-range dispersion and short-range Pauli principle attraction). \( E_{vdW_aals} \) and \( E_{coulomb} \) collectively constitute non-bonded interatomic interactions and unlike bond-order-dependent terms (all other terms mentioned), they do not go to zero as the bond breaks. Thus by including the \( E_{vdW_aals} \) and \( E_{coulomb} \) terms, awkward changes in the energy description as the bonds are dissociated are avoided. They are screened by a Tapering function to avoid energy discontinuities as the interatomic distance exceeds non-bonded cutoff radius.\(^{47, 48, 54, 55, 56}\)

### 2.1.3 ReaxFF force-field parameterization

Parameterization in ReaxFF is performed by optimizing the force-field parameters (with an initial guess based on previous calculations or chemical intuition) against a set of pre-calculated data (called the “training set”) derived for example from first-principles calculations or SCC-DFTB, or experiment. For optimization, a least-squares fitting procedure of the given data sets is used. The data comprising the training set might include atomic charges, formation energies, heats of formation and energy as a function of angle strains and bond lengths for various crystal structures.\(^{48}\)
One would hope and expect that the derived force-field is reliable for systems similar to those included in the training set. To make the force-field transferable (for the future application in mind), information on all states of the chemical compound should be included. Furthermore, to allow for a good bond-order description, phases with different coordination numbers and different bond lengths should also be in the training set. In this work, the coordination-dependent data were acquired by performing a volume scan for each polymorph as well as an angle-energy scan (by skewing the unit cells) for each of the ZnO bulk structures. The ZnO polymorphs used in the parameterization of the force-field were wurtzite, zincblende, rocksalt, CsCl, BCT and cubane (see Figure 1.1).

The parameterization procedure is analogous to finding a solution to a sophisticated function (force-field) with many parameters. In the ReaxFF program, the standard procedure used to fit force-field parameters is a successive one parameter minimization routine (least squares) in which at each step the force-field parameters are changed to minimize the following sum of squares (defined as the total error):

\[
Total\ Error = \sum_{i=1}^{N} \left( \frac{\chi_{i,SCC-DFTB} - \chi_{i,ReaxFF}}{\sigma_i} \right)^2
\]  

(2.6)

\(\chi_{i,SCC-DFTB}\) and \(\chi_{i,ReaxFF}\) are, respectively, SCC-DFTB and ReaxFF calculated properties (energies, charges, etc.) and \(\sigma\) is the weight given to a certain structure. For most cases, the most important part of the data set that needs to be well-described by ReaxFF are also the most stable structures. However, based on the intended application of the force-field, one can modify weights to prioritize certain structures.

The stages used in the ReaxFF parameterization procedure in this work are schematically listed below:

(i) Determine which energy terms should constitute the ReaxFF energy expression. For the ZnO systems in this project \(E_{\text{bond}}, E_{\text{lp}}, E_{\text{val}}, E_{\text{vdWaals}}, E_{\text{coulomb}}, E_{\text{over}}\) were used (see Section 2.1.2).

(ii) Determine which force-field parameters need to be optimized. All parameters are given an initial value in a physically meaningful range (based on previous calculations or chemical intuition).

(iii) Generate an initial training set of an array of data points comprising energies, atomic charges and geometric information for a range of selected structures. For the polymorphs, volume-energy and unit cell angular scans are typically included in the data set.

(iv) Optimize the force-field parameters by fitting them to the training set using the built-in one-parameter technique.
(v) Check if the force-field is able to reproduce the training set data in a satisfying manner or if the parameters that are optimized need to be modified [if satisfied, go to step (vii)].

(vi) Repeat steps (iv) and (v) until the fitted force-field is satisfactory.

(vii) Validate the force-field by running calculations on structures not included in the training set. If ReaxFF and SCC-DFTB results do not correspond, these structures should be added to the training set and the procedure repeated [back to step (iii)].

2.2 Properties

Several properties were examined in this thesis, especially structure stability and elastic properties. In this section we mention two properties that warrant some extra elaboration. Thus, this section is organized in two parts to discuss definitions and procedures for surface energy and bulk modulus calculations. In steps (v) and (vii) of the ReaxFF parameterization process, properties are calculated and compared for ReaxFF optimized structures.

2.2.1 Murnaghan equation of state

For each of the ZnO crystal polymorphs considered in this project, a volume-energy curve was generated. The bulk modulus, $B_0$, for each of the polymorphs was calculated by fitting the data points to the Murnaghan equation of state. The general formula for this equation of state is:

$$E_T(V) = E_T(V_0) + \frac{B_0 V}{B'_0} \left[ \frac{(V_0/V)^{B'_0}}{B'_0 - 1} + 1 \right] - \frac{V_0 B_0}{B'_0 - 1}$$ (2.7)

where $E_T(V)$ is the total energy at volume $V$, $B_0$ is the bulk modulus and $V_0$ and $B'_0$ are volume and bulk modulus pressure derivative for the optimized structure (system at the equilibrium).

It should be noted that for the cubic crystalline structures of ZnO (CsCl, rocksalt, zincblende, cubane), volumes were changed isotropically, i.e. it was assumed that the unit cell is a perfect cube, i.e. that $a = c$. For structures with an internal parameter $u$ where $a \neq c$, the anisotropy of the underlying crystal structure has been taken into consideration. Thus, for each volume tested for each of these structures (wurtzite, graphitic, BCT), an $a/c$ scan to find the optimum cell parameters was performed.
2.2.2 Surface cleavage energies

By definition, the cleavage energy, \( E_{\text{cleavage}} \), and the surface energy, \( E_{\text{surf}} \), are interlocking concepts. The latter is the energy penalty to build a unit area of a surface from a bulk system. The cleavage energy for non-polar surfaces (identical faces) is the amount of energy required to cleave a surface from the bulk and is therefore twice the surface energy:

\[
E_{\text{surf}} = \frac{1}{2} E_{\text{cleavage}} = \frac{1}{A} (E_{\text{slab}} - (N_{\text{cell}} \times E_{\text{ZnO-bulk}}))
\]  

(2.8)

where \( A \) is the area of the surface, \( E_{\text{slab}} \) is the total energy of the surface slab in Joule, \( N_{\text{cell}} \) is the number of zinc oxide formula units present in the slab and \( E_{\text{ZnO-bulk}} \) is the total energy of the bulk unit cell divided by its number of formula units. The unit for \( E_{\text{surf}} \) is hence J/m\(^2\).

Periodic three-dimensional non-polar and polar surface slabs (surface supercells) were cleaved from wurtzite-ZnO bulk structure repeated in the \( a \) and \( b \) directions of the unit cell. Slabs were constructed with a large vacuum gap (50 Å) to make sure they do not interact with each other. They were optimized using the SCC-DFTB method and all atomic positions were optimized (relaxed), but in order to mimic the bulk wurtzite-ZnO behavior, the equilibrium lattice parameters of the reference state (wurtzite) were maintained in the \( a \) and \( b \) directions.

2.3 Program Modules

In this section, a brief overview of the Python programming language and the Atomistic Simulation Environment used throughout this project is given.

2.3.1 Python programming language

Python is an interpreted,\(^{59}\) dynamic typing language, i.e. type checking is performed at run time instead of compilation time. It is also a high-level language, i.e. it is with strong abstraction from computer level and thus closer to human language. Python can be used in object-oriented programming to develop a full-fledged interactive software. Since Python is an interpreted language, i.e. no compilation of the code is needed, it can as well be used as a scripting language. Its rich and comprehensive set of standard libraries allows an effective applicability of Python in scientific computing. In this project Python has both been used in structured programming to facilitate the utilization of ASE and DFTB+ calculations and as a scripting language.
2.3.2 Atomistic Simulation Environment (ASE)

The ASE (Atomistic Simulation Environment) is a set of programming libraries containing modules that altogether provide an easy to use environment for atomistic simulations and greatly facilitate performing atomistic simulations and the analysis of such. ASE is highly customizable and a result of collaboration between many scientists. There are therefore modules for energy, force and stress calculations using different computational chemistry programs including DFTB+, as well as modules for structure optimization and other standard simulation tasks. ASE also provides a graphical user interface (GUI) to create, visualize, modify and perform calculations on structures. ASE is written in Python and the official reference to the ASE suite of programs is available on the internet (wiki.fysik.dtu.dk/ase).

ASE features such as flexibility, customizability and its pythonic nature qualifies it to be used as a core platform that can be expanded for a particular purpose. In this project, ASE was used as the underlying environment to which our multiscale modeling features were added. In this way we have taken the first steps towards building a tailored multiscale environment for surface science problems, based on the ASE environment. This means that ASE modules created by other developers were extensively used and modified where necessary. These involve the creation of the structures (as Atoms objects\textsuperscript{*}), optimizing them, preparing DFTB+ input files, performing the SCC-DFTB calculations, plotting and presentation of results and storing data (see Figure 2.1).

\textsuperscript{*}In object-oriented programming (OOP), objects represent instances of classes that have data fields describing the attributes of the object and also methods that are the subroutines or procedures associated with that object. Thus, an ASE Atoms object refers to an instance of ASE Atoms class that has attributes such as chemical symbols, geometric positions of atoms and methods to set a calculation method for that Atoms object and calculate total energy, charges and forces (the calculation methods are implemented into ASE calculator objects which take the Atoms object as their input and, depending on the calculation method, the energies, charges and optimized atomic positions as their output).

The ASE object provides large flexibility in the multiscale simulation environment. In principle, the ASE Atoms object (hereafter simply referred to as Atoms object) stores data irrespective of the simulation method used. It is, however, updated and its attributes expanded to contain all the information necessary which is essential when multiple methods are used (as is the case in our multiscale simulation environment).
Figure 2.1 – A schematic representation of the role of ASE in the Multiscale Simulation Environment. Using a cloud to signify the environment implies that the platform is independent of the simulation method and allows for transference of data in and out of methods in the size/time hierarchy (Figure 1.4) in an effective way via tools and utilities developed based on ASE.
Chapter 3

Development of the Multiscale Simulation Environment

In this project, we have developed software for the Multiscale Simulation Environment (MSE) to handle part of the calculations that can be generated automatically and that otherwise are error-prone and complicated during, e.g. a parameterization procedure. The environment software provides an interactive interface through which the settings of different calculations can be defined. Our environment software calls ASE library routines to perform calculations and retrieve data as needed. These library routines are either those provided by ASE and used by us in their original form, or routines customized by us to meet an intended task or written from scratch by us for a purpose formerly not yet implemented in ASE. Moreover, for the latter we integrated our newly developed modules into the ASE package, both to make them easy to use, and as contributions to that package, potentially upon discretion of the ASE developer team.

In this chapter, we will first discuss the overall project workflow to explain the role of the developed MSE. This is followed by a description of the implementation of the MSE software in terms of the functionality and the graphical user interface (GUI).

3.1 Project workflow

The work in this thesis is based on four interdependent steps as illustrated in Figure 3.1. The first step concerns the development of computer tools and a software framework to systematically carry out SCC-DFTB calculations and automatically pass computational data from SCC-DFTB to ReaxFF. The second step will provide a fast route to enabling parameterization of ReaxFF. The third step will emphasize the application of the acquired force-field on ZnO structures not included in the training set and vali-
dation of consistency of results with SCC-DFTB (1:1 relationship in both energy and structure). The MSE environment handles the first three steps. In the fourth step, the resulting ReaxFF is ready to be used for production.

In the first step, we use the SCC-DFTB method to perform calculations on ZnO bulk structures and prepare a training set for ReaxFF. To this end, we developed tools to automatically calculate, for each bulk polymorph, the optimized lattice parameters, crystal geometry, total energies and energy/volume scan profiles for each structure. The computational data from SCC-DFTB, once sufficient accuracy is obtained, are then used in an automatic way to construct a training set for ReaxFF*.

In the second step, the calculated energies and structures from step one are used to parameterize ReaxFF. Step two is continued until the total error given by the ReaxFF program (defined in Section 2.1.3) is minimized.

![Figure 3.1](image)

**Figure 3.1** – A schematic illustration of the project workflow. All SCC-DFTB calculations as well as training set preparation are maintained using the MSE.

In the third step, some structures initially absent from the training set, are first calculated using the new force-field and the resulting geometries and energies are compared with the reference SCC-DFTB results. If the difference in energetic and structural data between the two methods is sufficiently small, the force-field is considered to work also for other similar structures. Otherwise, we extend the training set by taking the structures one step back on the size-temporal scale to SCC-DFTB (validation loop, see Figure 3.1) and repeat the parameterization procedure. This validation

*For the sake of simplicity, “training set” is used in this text to exclusively denote “ReaxFF training set” unless stated otherwise.
loop is continued until a one-to-one relationship of descriptions of the structures in the training set in terms of their energetic and structural information is achieved between ReaxFF and SCC-DFTB.

In the fourth step, the developed force-field is ready to be used for production runs. Here, we study ZnO nanowires and nanotubes in an array of experimentally relevant sizes and the results are provided in Chapter 5.

3.2 MSE overview

The workflow to create a ReaxFF training set using the MSE environment is illustrated in Figure 3.2. We will here go through the different steps involved and the developed software packages used. The MSE has a GUI in which the specifications for the systems in the training set are given. The data are then passed to objects and functions to automatically generate structures, perform SCC-DFTB calculations when needed and translate the data into the ReaxFF training set. We use the DFTB+ implementation of the SCC-DFTB, which is integrated in the ASE (see Section 2.3.2) package to prepare inputs, perform calculations and extract results. In this environment, geometry and energetic data as well as other information such as charges and forces for each structure are contained in the ASE Atoms object. This object facilitates transfer of data between SCC-DFTB and ReaxFF in a uniform fashion without losing any information and also provides an efficient way to store the results. Hence, the software pickles\(^\dagger\) the Atoms objects into the ASE Trajectory file format, which is the standard format used in ASE, after performing SCC-DFTB calculations. This format is binary and thus requires less storage capacity than non-binary storing formats and can contain one or many Atoms objects. At every run, the software checks the existing results so as to avoid repeating the same calculations that have been done before.

In this project, I developed the GUI using the Tkinter library\(^\ddagger\) and automates SCC-DFTB calculations and the preparation of the training set for ReaxFF. The graphical environment provides tools to choose the chemical elements and crystal structures for automatic inclusion in the training set for ReaxFF parameterization. On the chosen bulk polymorphs, the environment facilitates SCC-DFTB calculations and have the ability to extend the training set by importing pre-calculated SCC-DFTB data when

\(^\dagger\)Pickling is the process of serializing a Python object into a byte stream. Unpickling is the inverse of this process in which the serialized data are read back into a Python object.

\(^\ddagger\)A Python library is a set of modules that provide access to specific system functionality. Tkinter is Python’s standard GUI toolkit with a set of graphical widgets (e.g. check buttons, textboxes etc.). A window is one such graphical widget with a visual area (or a graphical interface), usually with a rectangular shape, in which graphical items, such as clickable buttons and drop-down menus and buttons are placed.

In this chapter, the word button is used to denote a clickable, usually rectangular area placed on a window that is used to trigger a programming procedure e.g. start calculations, show another window etc.
Figure 3.2 – The GUI workflow to prepare training set for ReaxFF (step 1 in Figure 3.1). For the classes and graphical interfaces, functionality and the method names are listed. Rounded rectangles show activities of the methods and the numbers in parentheses signify the order in which methods are invoked. Solid arrows show forward flow while the dashed lines mean that the method invoked, returns the results to its containing object.

necessary. Also a graphical tool is provided to perform SCC-DFTB calculations and produce the bulk moduli for the selected bulk crystals.
3.3 MSE workflow

In the following, I will expand on the two main graphical interfaces, the Main window and the Import window and discuss the procedure through which the GUI buttons call functions to perform calculations and prepare a ReaxFF training set in this environment (as illustrated in Figure 3.2). I will discuss the most important functions associated with each graphical tool, class§ and its methods¶. The full documentation of the various classes, variables, methods and functions implemented in the MSE can be found in Appendix A.

3.3.1 Main window

A screenshot of the Main window is shown in Figure 3.3. This interface appears after running the Python executable file. Here the user specifies the chemical elements and the bulk polymorphs to be included in the training set. The chemical elements that can be chosen are those for which the Slater-Koster tables and corresponding repulsion potential (contained in the SK-files) are available, i.e. Zn, O, C, H, P, Cu, S, N, Fe and Ti. The bulk structures that are currently available to choose for automatic generation of the unit cell are rocksalt, CsCl, zincblende, graphitic, cubane, BCT and wurtzite. For each of the bulk polymorphs, the user can set a weight to use (for the equilibrium structure) and decide whether to include SCC-DFTB calculated charges∥ and perform angular-scan of the structure too. The number of parameterization steps can be specified in the PARAMS steps textbox. This variable defines how many steps that should be taken in the ReaxFF parameterization which is written into the corresponding input file for Reax.

3.3.2 Import window

Apart form the Main window which is used to calculate bulk polymorphs, we developed the Import window with the purpose of importing the pre-calculated SCC-DFTB results

§In object-oriented-programming, a class is a template from which objects are created. In a class, one defines what properties and methods the objects might have whereas when an object is instantiated, the properties are given variables. In other words, a class might have a field called 'elements' whereas, for an object this field is given a value of, e.g. ZnO.

¶In the programming terminology, a method is usually a piece of code associated with an object. An object itself is an instance of a class and can contain different methods and attributes. To make a distinction between a function and a method, we can say that the former is explicitly called and is not part of an object whereas, we pass the data implicitly [by calling the Object.Method()] to a method. To avoid ambiguity between a computational method and a programming method (defined above), the word method is italicized throughout this thesis work whenever the latter meaning is intended.

∥Charges were not read by the original ASE calculator for SCC-DFTB, but I wrote a function to extract Mulliken charges from the DFTB+ output into the Atoms object. In this capacity, I made adjustments to the original ASE code when necessary.
Figure 3.3 – Screenshot of the Main window of the GUI. This is the interface that is shown when the program is executed.

into the training set. Here, the user is prompted to choose the SCC-DFTB output files; data are read from the files and converted into Atoms object. The information that is read from the SCC-DFTB include, energies, geometry, forces and Mulliken charges. In the Name textbox the user determines what the corresponding Atoms object is referred to in the ReaxFF input files; the weight for an imported structure is also set. In the selection panel in the window, one or multiple structures can be chosen. The user can also edit the previously entered data using the “Update” button which automatically saves the new settings in the corresponding stored files. Upon clicking the “Close” button, the imported structures are appended to the Atoms list which will be written in the ReaxFF training set and the Main window is shown again. The previously imported objects are stored in 'extensions' directly and are available every time the program is started.

3.3.3 "Generate Training-set" button

The "Generate Training-set" button in the Main window triggers the SCC-DFTB calculation and training set generation process, i.e. step in Figure 3.1. As seen in Figure 3.2, when this button is clicked, a checker method is invoked. This method creates a folder name from the chemical elements chosen in the alphabetical order using underscore as delimiter (e.g. O_Zn if oxygen and zinc are chosen) and checks if it already exists in the root directory. If it does not, it calls initialBulks class for all the structures. Otherwise, if a folder with data for the selected structures is present, the checker method reads the data from the previous calculations into a list of Atoms objects and passes it to the reaxCalculator module (for documentation of this module see Appendix A)
3.3.4 InitialBulks class

The *InitialBulks* class generates initial guesses for the crystal structures and perform a k-point sampling convergence test to find the optimal k-point sampling for each specific bulk polymorph. As shown in Figure 3.2, to perform a SCC-DFTB calculation for each polymorph, first the *initialSt* method is called. This method uses the ASE to automatically create a unit cell, given the initial lattice parameters and the space group for that bulk crystal. The initial structures are then passed to the *calcKP* method where single point SCC-DFTB calculations, starting with 1x1x1 k-point sampling is performed. The number of k-points in each calculation is increased (2x2x2 for the second iteration and so on) until convergence is reached. The default convergence criterion for the k-point sampling to be deemed sufficient for SCC-DFTB optimization was set to 0.001 eV or the maximum of 8x8x8 k-points in this thesis work. This means that if the total energy between two iterations is lower than this value, or if the number of k-points has reached the maximum, *calcKP* will stop running further and return the k-point sampling to its containing object. The input structures and the k-points meshes are then passed to the *calculateMain* Class.

**We use medial capitals (camel case) naming convention throughout the GUI code.**
3.3.5 CalculateMain class

Figure 3.2 shows the methods associated with the calculateMain class. The first method called in this class is setRef. For each chemical system, this method performs a spin polarized SCC-DFTB single point calculation. The results, similar to all SCC-DFTB calculations done in the calculateMain class, are stored in the ASE Trajectory format.††

Next, the initial structures and the optimal number of k-points from the previous stage are fed into the optimize method of the calculateMain object instance. Here, we use the ASE to invoke DFTB+ and perform SCC-DFTB calculations with optimization of the lattice and geometry. For the calculations in this thesis work, the Fermi-temperature was by default set to zero Kelvin; the geometry optimization maximum number of steps to 225; and the allowed maximal force component to $10^{-4}$ a.u..‡‡ The optimized structure is then passed to the volumeScan method. Here, an isotropic volume scan of ±5% of the equilibrium structure volume using SCC-DFTB calculations is performed. For each volume, the lattice is kept constant while the internal coordinates are allowed to relax. The Atoms objects are then stored in the ASE Trajectory format.

The next phase in step 1 is to calculate an angle-scan starting from the optimized structure. This is only done if the corresponding checkbox in the Main window interface is selected for the bulk structure. Skewing of the bulk structures is done to obtain the angular dependence on the bond strengths, which is of importance when parameterizing ReaxFF. The angleScan method changes cell coordinates and angles with all ±10 degrees combinations of alpha, beta and gamma cell angles and performs single-point SCC-DFTB calculations.

The results of the geometry optimization, volume scan, angle-scan and charges (in an Atoms list) are then passed to the reaxCalculator module to be written into the ReaxFF input files (for details of the reaxCalculator module refer to Appendix A).

3.3.6 “Calculate Bulk Modulus” button

This button in the Main window is used to perform necessary SCC-DFTB operations to calculate the bulk-moduli of the selected bulk polymorphs. The default mode for the bulk modulus calculation is ACscan. As illustrated in Figure 3.5, in this mode the function first checks if the crystal unit cell is cubic or not. For the cubic unit cells, an isotropic volume scan as described in Section 3.3.5 is done. If the unit cell is non-cubic, an

††Here as well I wrote a function for the SCC-DFTB calculator in the ASE which reads the orbital population from the SK-files and assigns the number of unpaired electrons to the Atoms object.
‡‡The geometry optimization stops if the maximal force component is smaller than $10^{-4}$ a.u.
first an isotropic volume scan of the equilibrium structure is performed and then for each volume, the energies for combinations of lattice \(a\) and \(c\) parameters (90-120\% of optimized unit cell \(a\) parameter) which have that volume are scanned. The method then uses ASE to invoke DFTB+ and the energies and structures are stored. The lattice with the minimum energy at each of the volumes is then selected and passed to the ASE Murnaghan Equation of State to calculate the bulk modulus for that structure and to plot data.

\[
\text{Figure 3.5} - \text{The flow of events when \"Calculate Bulk Modulus\" button is clicked in the Main window. For the calculations on ZnO systems in this project we used the ACscan mode.}
\]

If the mode for the bulk modulus calculation is set to pressure mode, a method is called to perform SCC-DFTB calculations to generate energies and optimized geometry of the structures by applying positive and negative pressures. By negative pressure we mean the negative number we pass to the DFTB+. We did not use this mode in our calculations in order to keep the results for the bulk modulus in accordance with the ReaxFF method which does not have pressure as a variable.
In this chapter, we present steps 2 and 3 of the procedure presented in Figure 3.1, i.e. the generation of a ReaxFF force-field for ZnO based on SCC-DFTB data.

4.1 Parameterization

I have used the developed MSE described in Chapter 3 to try to generate an optimal force-field for ZnO with the goal of obtaining a one-to-one correspondence in both energetics and structure between ReaxFF and SCC-DFTB data. First, I generated a training set based on the ideal bulk structures of ZnO and the most stable surfaces (step 1 in Chapter 3). This step was followed by a parameterization of a ReaxFF force-field (step 2 in Chapter 3) which is validated through calculations on ZnO structures outside the training set (step 3 in Chapter 3). In this third step, structures that were not well described with the generated force-field were added to the training set followed by re-parameterization (i.e. back to step 1). In practice, steps 1, 2 and 3 are iterated until I obtained a force-field describing different ZnO structures with good accuracy in the validation loop illustrated in Figure 3.1. I will here present one full cycle leading to the force-field I use to study ZnO nanowires in Chapter 5.

4.1.1 Step 1: Generation of the training set

The MSE was used in the first step to perform SCC-DFTB calculations on ZnO structures and generate a training set of data points for ReaxFF parameterization. The main computational details (choice of SK-files, Fermi-smearing, convergence criteria, etc.) for
the SCC-DFTB calculations were described in Section 2.1.1. The k-point sampling for the different crystal structures is checked and set automatically within the MSE. As mentioned above, for the different ZnO polymorphs, we used a convergence criterion on the total energy of 0.001 eV, i.e. the number of k-points were incrementally increased until the total SCC-DFTB single point energy between two consecutive iterations were below this value. The resulting k-points grids obtained for each ZnO bulk polymorph are shown in Table 4.1.

Table 4.1 – k-point sampling used in optimization of ZnO bulk crystals at SCC-DFTB level.

<table>
<thead>
<tr>
<th>ZnO bulk type</th>
<th>Wurtzite</th>
<th>Zinblende</th>
<th>Rocksalt</th>
<th>BCT</th>
<th>Cubane</th>
<th>CsCl</th>
<th>Graphitic</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-point mesh</td>
<td>6x6x6</td>
<td>4x4x4</td>
<td>5x5x5</td>
<td>7x7x7</td>
<td>4x4x4</td>
<td>8x8x8</td>
<td>6x6x6</td>
</tr>
</tbody>
</table>

With this computational setup, SCC-DFTB calculations for the different bulk polymorphs were made. Optimized structures, volume scans and angle scans for wurtzite, zinblende, rocksalt, CsCl-type, BCT-type and cubane-structure were added to the training set. The SCC-DFTB data for the graphitic ZnO structure was preserved for testing purposes. The volume-energy scans were made by calculating single point SCC-DFTB energies for structures within 95% to 105% of the optimized bulk phase volumes. The angle-energy scan data were produced by performing single point SCC-DFTB calculations for unit cells with the alpha, beta and gamma angles skewed by ±10 degrees. For the wurtzite structure, we additionally added SCC-DFTB data for the low index (10\bar{1}0) and (11\bar{2}0) non-polar surfaces modeled through slabs with different thicknesses to the training set.

4.1.2 Step 2: Parameterization of ReaxFF

Steps taken to optimize a set of ReaxFF force-field parameters are discussed in Section 2.1.3. To fit the force-field parameters to the data in the training set, we used the built-in ReaxFF one parameter routine described in Section 2.1.3. The initial guess for the different parameters to be optimized were those fitted against hybrid DFT calculations with the B3LYP functional developed by Raymand et al. (2010), henceforth referred to as FF0. In each fitting iteration, ReaxFF changes the chosen parameters and, based on the weights of the structures, calculates a total error as described in Section 2.1.3. Figure 4.1 shows the ReaxFF total error as a function of the fitting iteration number. We should mention that the numerical values of the total error in Figure 4.1 for regions with different weights (bordered with vertical lines) are not absolute. During parameterization we monitored this total error as well as the energy comparison of the structures during the fitting procedure. When for each set of weights, the total error had converged, we changed the weights to add more emphasis on the structures which were contributing more to the error and thus had a lower quality energy description in ReaxFF compared to the SCC-DFTB.
The training set for the force-field, optimized for bulk polymorphs and non-polar wurtzite ZnO surface structures, included 117 data points and in total 29 force-field parameters were fitted to the data set. We call this force-field FF1.

![Graph](image)

**Figure 4.1** – ReaxFF total error as a function of parameterization steps (iterations). The fluctuations in the total error are resulted from major changes made to the weights of the structures (ReaxFF weight and total error are defined in 2.1.3). The numerical values of the total error is not comparable for parameterization iterations with different weights.

### 4.1.3 Step 3: Validation and extension of the training set

The next stage was the validation of the FF1 parameters with the SCC-DFTB method. The purpose of this stage was to see how the structural and energetic information for the FF1 optimized structures compare with the results given by SCC-DFTB. For the energy validation, I optimized the structures with the FF1 force-field and compared the energies of formation given by the ReaxFF with those calculated for the optimized SCC-DFTB structures (optimization of the lattice and/or atomic positions, henceforth referred to as SCC-DFTB Opt). To validate the performance of FF1, I performed SCC-DFTB SP calculations for the FF1 optimized (FF1-opt) systems (for the sake of simplicity, we henceforth refer to single point SCC-DFTB calculation of ReaxFF optimized structures as SCC-DFTB SP) and compared their relative energies against the relative energies from SCC-DFTB Opt calculations. This method is the best practice to examine the one-to-one correspondence mentioned earlier between the two levels of theory. However, for the larger nanosystems, SCC-DFTB Opt calculations are computationally too expensive. Therefore, in such cases the validation is done by comparing the ReaxFF energies with those given with SCC-DFTB SP calculations.
Figure 4.2 compares the formation energies ($\Delta E_f$) for the structures in the FF1 training set calculated with methods at different levels. In the figure I have used the first validation method mentioned above. The figure shows how the FF1 compares for the structural (left picture) and the energetic (right picture) data with SCC-DFTB. The proximity of the FF1 data points to the blue line compared to FF0 in the left picture in Figure 4.2 suggests that the SCC-DFTB Opt and FF1 optimized structures are quite similar. The right picture in Figure 4.2 confirms that for the ZnO bulk-polymorphs and surfaces, the energies are also well reproduced.

As FF1 describes ZnO bulk and non-polar surface energies close to the SCC-DFTB, we used this force-field for the next stage in our parameterization scheme. Here, we tested the force-field for structures not included in the training set for FF1. To this aim, the force-field was used to optimize ZnO solid nanowires and hollow nanotubes with different diameters cut from the SCC-DFTB optimized wurtzite-ZnO. Here, we used 18 different nanorods with a diameter between 10 to 23 Å, optimized them using ReaxFF and SCC-DFTB and compared the results. Figure 4.3 shows relative SCC-DFTB SP formation energies, $\Delta E_f^{SCC-DFTB SP}$ (for FF1-Opt structures) versus $\Delta E_f^{SCC-DFTB Opt}$ for these structures. Any significant discrepancy in the total energies implies inadequacy of the force-field to describe those structures correctly (as correct as the reference method). We defined the acceptance threshold for this energy difference for the structures in the test suite to be 0.05 eV. Hence, we used the MSE to update the training set for the data-points where $|\Delta E_f^{SCC-DFTB Opt} - \Delta E_f^{SCC-DFTB SP}|$ was larger than 0.05 eV and repeated the parameterization process. The re-optimization

\[ \text{SDE} = \frac{1}{N} \sum_{i,j} (E_{at}^{SCC-DFTB Opt} - E_{at}^{SCC-DFTB SP \ for \ ReaxFF-Opt})^2 \]

where N is the number of data points. The higher numerical value for SDE indicates higher deviation error from the reference formation energies.

\*Standard deviation error (SDE) here used to signify how close the results from each method to the reference data are and is defined as:
of FF1 parameters for the ZnO nanorods resulted in a new force-field referred to as FF2. In the training set for FF2, we also included data for the graphitic-ZnO bulk by including both an volume-energy scan and an angle-energy scan.

Figure 4.3 – Formation energies relative to $E_{\text{wurtzite}}$, calculated with SCC-DFTB with optimization versus single point SCC-DFTB on FF0, FF1 and FF2 optimized nanowires and nanotubes. Standard deviations from SCC-DFTB Opt are 0.046, 0.020, 0.0.25 for FF0, FF1 and FF2 respectively.

4.2 Choosing the optimal force-field

In the production phase (step 4 as discussed in Section 3.1), in which I used ReaxFF to study larger nanosystems, I needed to use the most optimal of the two force-fields for the target systems. Therefore, I conducted the validation procedure for the ZnO bulks and surfaces (see Figure4.2) and a test-suite of ZnO nanorods (see Figure4.3), as described above, using both FF1 and FF2. I also calculated structural and energetic properties for the bulk polymorphs (tabulated in Table4.2) and the non-polar surfaces (shown in Table4.3) using FF1 and FF2, and for comparison I also used FF0. In the following, I will discuss these results and assess the adequacy of each of the force-fields for the descriptions of ZnO bulks, non-polar surfaces and nanostructures.

4.2.1 Bulk structures

In this thesis work, the bulk systems considered are wurtzite, rocksalt, zincblende, graphitic, CsCl, BCT and cubane phases of ZnO. The volume-energy curves of these bulk phases calculated with SCC-DFTB, FF0, FF1 and FF2 are shown in Figure4.4.
In general, the FF1 results and the reference SCC-DFTB are in a good agreement for the bulk structures. The most stable phase is wurtzite which is also the case in experiments. Zincblende or sphalerite ZnO is energetically very similar to wurtzite-ZnO but with a cubic crystal system. Furthermore, under high pressures rocksalt phase transition occurs. Transitions from wurtzite to the BCT and the cubane phases requires negative pressures.

**Figure 4.4** – Formation energies relative to $E_{\text{wurtzite}}$ for volume scans of seven of ZnO bulk polymorphs calculated with SCC-DFTB, FF0, FF1 and FF2. Except for graphitic ZnO, the other polymorphs were included in the parameterization of FF1. Graphitic data from SCC-DFTB was also included in the re-optimized FF2 training set.

The lattice parameters, bulk moduli, $B_0$, and atomization energies, $E_{\text{at}}$, of the optimized ZnO bulk polymorphs calculated with SCC-DFTB, FF0, FF1 and FF2 are shown in Table 4.2. In addition, I have calculated band gap energies $E_{\text{gap}}$ using SCC-DFTB SP (single point SCC-DFTB for the structures optimized using ReaxFF). In general, the energetics and lattice parameters calculated with the optimized FF1 and FF2 are close to the corresponding SCC-DFTB calculated values for all seven bulk structures. However, there are some significant discrepancies between the bulk properties obtained from the two methods. For instance, the FF1 lattice parameters for wurtzite-ZnO are closer to the experimental values of $a = 3.250$ Å, $c = 5.207$ Å and $u = 0.3825$ Å than SCC-DFTB and FF2.$^{61}$ This is not the case for the $E_{\text{gap}}$.

$^{61}$To calculate the bandgap energies for FF0, FF1 and FF2 optimized bulk polymorphs, we first optimize the structures with ReaxFF and then attain the electronic properties by performing single point
which is measured experimentally to be 3.4 eV for wurtzite-ZnO. However, the close proximity of $E_{\text{gap}}$ for the FF1 optimized wurtzite structure (4.36 eV versus 4.44 eV for FF2 optimized one) to the corresponding value for the SCC-DFTB optimized structure (4.33 eV) suggests that the electronic structure is better preserved by the FF1 optimized structure and the one-to-one correspondence (in both structure and energy) between the methods has to a great extent been achieved. In general, also for the other ZnO polymorphs, the $E_{\text{gap}}$ for the structure obtained with FF1 values are closer to the SCC-DFTB ones compared to FF2 (see Table 4.2).

As for the atomization energies, with SCC-DFTB the $E_{\text{at}}$ of wurtzite and zincblende ZnO bulks are close whereas both FF1 and FF2 slightly widen this energy difference by ~0.04 eV (with wurtzite phase still the most stable structure). The $E_{\text{at}}$ for all of the bulk polymorphs calculated with FF1 and FF2 are ca. 0.1 eV higher than those predicted by SCC-DFTB. A more serious discrepancy, is that both FF1 and FF2 predict the BCT phase to be more stable than the graphitic structure, which is contrary to the more stable graphitic phase in the reference method. This indicates that the graphitic ZnO polymorph is not well described by either the FF1 and FF2 force-fields although energetic and structural data for this phase were included in the training set for FF2. As seen in Figure 4.4, apart from the graphitic phase, the order of the atomization energy values for the bulk phases are the same with SCC-DFTB and the force-fields, with the wurtzite structure being the most stable phase followed by zincblende, BCT, rocksalt, cubane and the CsCl.

The bulk moduli ($B_0$) were calculated using the Murnaghan equation as described in Section 2.2. The bulk moduli calculated with FF1 and FF2 are in accordance with the SCC-DFTB calculated values. However, except for the graphitic phase, FF1 values for $B_0$ are closer to the reference method compared to FF2.

4.2.2 Surface structures

To further compare the force-fields, I also calculated the surface energies $E_{\text{surf}}$ (discussed in Section 2.2.2), relaxation angles $\theta$ and surface Zn-O distances (defined in Figure 1.2) for the low-index non-polar ZnO (10\,\overline{1}0), (1\,\overline{1}2\,0) and cleavage energy $E_{\text{cleavage}}$ for polar (±0001) ZnO surfaces using FF0, FF1, FF2 and SCC-DFTB (see Table 4.3). The relaxation angle and Zn-O surface distance give information on how well the surface structural properties calculated with methods at different levels compare. As can be seen in Table 4.3, surface energies from FF1 and FF2 for both non-polar surfaces are close to those of SCC-DFTB. However, for the (10\,\overline{1}0) non-polar ZnO surface, $E_{\text{surf}}$ calculated for FF1 optimized structure is in better agreement with SCC-DFTB while FF2 performs better for the (11\,\overline{2}0) surface. Structurally, both FF1 and FF2 give too

---

SCC-DFTB calculations on them.
Table 4.2 – Lattice parameters $a$, $c$ and $u$ (defined in Figure 1.1), atomization energy $E_{at}$, bulk modulus $B_0$ and bandgap energy $E_{gap}$ for various ZnO polymorphs calculated for SCC-DFTB, FF0, FF1 and FF2 optimized structures.

<table>
<thead>
<tr>
<th></th>
<th>SCC-DFTB</th>
<th>FF0</th>
<th>FF1</th>
<th>FF2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Wurtzite</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>3.21</td>
<td>3.27</td>
<td>3.24</td>
<td>3.25</td>
</tr>
<tr>
<td>$c$ [Å]</td>
<td>5.26</td>
<td>5.31</td>
<td>5.15</td>
<td>5.06</td>
</tr>
<tr>
<td>$u$</td>
<td>0.374</td>
<td>0.375</td>
<td>0.386</td>
<td>0.390</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>162</td>
<td>162</td>
<td>162</td>
<td>159</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>4.33</td>
<td>3.96°</td>
<td>4.36°</td>
<td>4.44°</td>
</tr>
<tr>
<td><strong>NaCl</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>4.35</td>
<td>4.26</td>
<td>4.35</td>
<td>4.37</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>9.56</td>
<td>7.39</td>
<td>9.66</td>
<td>9.73</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>229</td>
<td>329</td>
<td>248</td>
<td>205</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>2.33</td>
<td>2.88°</td>
<td>2.35°</td>
<td>2.33°</td>
</tr>
<tr>
<td><strong>Zincblende</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>4.54</td>
<td>4.60</td>
<td>4.54</td>
<td>4.53</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>9.88</td>
<td>7.18</td>
<td>9.96</td>
<td>9.95</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>162</td>
<td>112</td>
<td>147</td>
<td>150</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>4.27</td>
<td>3.95°</td>
<td>4.32°</td>
<td>4.35°</td>
</tr>
<tr>
<td><strong>Graphitic</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>3.33</td>
<td>3.39</td>
<td>3.34</td>
<td>3.38</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>4.87</td>
<td>4.72</td>
<td>4.72</td>
<td>4.72</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>8.75</td>
<td>7.52</td>
<td>9.84</td>
<td>9.82</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>187</td>
<td>198</td>
<td>163</td>
<td>164</td>
</tr>
<tr>
<td><strong>CsCl</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>2.71</td>
<td>2.81</td>
<td>2.71</td>
<td>2.70</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>8.87</td>
<td>6.41</td>
<td>8.98</td>
<td>8.99</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>267</td>
<td>332</td>
<td>261</td>
<td>242</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>0.48</td>
<td>1.20°</td>
<td>0.53°</td>
<td>0.56°</td>
</tr>
<tr>
<td><strong>BCT</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a$ [Å]</td>
<td>5.51</td>
<td>5.68</td>
<td>5.50</td>
<td>5.50</td>
</tr>
<tr>
<td>$c$ [Å]</td>
<td>3.25</td>
<td>3.25</td>
<td>3.25</td>
<td>3.25</td>
</tr>
<tr>
<td>$u$</td>
<td>0.367</td>
<td>0.361</td>
<td>0.365</td>
<td>0.364</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>9.71</td>
<td>7.66</td>
<td>9.88</td>
<td>9.86</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>137</td>
<td>166</td>
<td>160</td>
<td>161</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>4.52</td>
<td>4.00°</td>
<td>4.47°</td>
<td>4.45°</td>
</tr>
<tr>
<td><strong>Cubane</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$u$</td>
<td>0.329</td>
<td>0.327</td>
<td>0.328</td>
<td>0.326</td>
</tr>
<tr>
<td>$E_{at}$ [eV]</td>
<td>9.71</td>
<td>7.27</td>
<td>9.46</td>
<td>9.42</td>
</tr>
<tr>
<td>$B_0$ [GPa]</td>
<td>109</td>
<td>100</td>
<td>100</td>
<td>97</td>
</tr>
<tr>
<td>$E_{gap}$ [eV]</td>
<td>5.34</td>
<td>4.44°</td>
<td>5.18°</td>
<td>5.15°</td>
</tr>
</tbody>
</table>

*The values are calculated by performing SCC-DFTB SP on Reax optimized structures.*
low values for theta and too high values for the surface Zn-O distance compared to the SCC-DFTB reference.

Polar surfaces of ZnO exhibit significant reconstructions and even metallic phases.\textsuperscript{20,62} Modeling them thus is usually very demanding. With the purpose of generating a reactive force-field to study MD in ZnO nanoparticles, we endeavored to fit ReaxFF to SCC-DFTB data for polar Zn/O-terminated (±0001) polar surfaces. Our efforts in the scope of this project, to parameterize a force-field to describe both polar and non-polar surfaces, was unsuccessful. As seen in Table 4.3, FF1 and FF2 cleavage energies for these polar surfaces are not “compatible” with SCC-DFTB results.

Table 4.3 – Surface energies $E_{surf}$, relaxation angles $\theta$, and surface Zn-O distance (defined in Figure 1.2) for the ZnO (1010) and ZnO (1120) surfaces of the ZnO wurtzite structure. Cleavage energy $E_{cleavage}$ for the non-polar surface (±0001) is also included. (znopt values are taken from Hellström et al. (2013)\textsuperscript{14})

<table>
<thead>
<tr>
<th></th>
<th>SCC-DFTB</th>
<th>FF0</th>
<th>FF1</th>
<th>FF2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_{surf}(10\bar{1}0)\ [J/m^2]$</td>
<td>1.22</td>
<td>1.27</td>
<td>1.22</td>
<td>1.25</td>
</tr>
<tr>
<td>$E_{surf}(11\bar{2}0)\ [J/m^2]$</td>
<td>1.29</td>
<td>1.40</td>
<td>1.33</td>
<td>1.31</td>
</tr>
<tr>
<td>$E_{cleavage}(0001)\ [J/m^2]$</td>
<td>5.72</td>
<td>2.60</td>
<td>2.33</td>
<td>2.34</td>
</tr>
<tr>
<td>$\theta(10\bar{1}0)\ [deg]$</td>
<td>11.1</td>
<td>5.9</td>
<td>6.6</td>
<td>5.5</td>
</tr>
<tr>
<td>$\theta(11\bar{2}0)\ [deg]$</td>
<td>11.9</td>
<td>4.2</td>
<td>6.8</td>
<td>6.3</td>
</tr>
<tr>
<td>Zn-O(10\bar{1}0) \ [Å]</td>
<td>1.81</td>
<td>1.94</td>
<td>1.89</td>
<td>1.90</td>
</tr>
<tr>
<td>Zn-O(11\bar{2}0) \ [Å]</td>
<td>1.83</td>
<td>1.94</td>
<td>1.90</td>
<td>1.91</td>
</tr>
</tbody>
</table>

4.2.3 Nanostructures

Validation of the ReaxFF results with those of SCC-DFTB Opt as shown in Figure 4.2 suggests that FF2 re-optimization has resulted in loss of accuracy and further $\Delta E_f$s from the SCC-DFTB compared to FF1 calculated values for the bulk and surface systems. To make sure we choose the best of the two force-fields, I also performed the validation stage for the ZnO nanowires and nanorods.

As shown in Figure 4.3, despite the re-optimization, the standard deviation error from SCC-DFTB Opt energies is slightly higher for FF2 compared to FF1 for the smaller nanorods. We also used the second validation method (relative energies calculated with single point SCC-DFTB for ReaxFF optimized structures versus the relative energies for these structures generated using the ReaxFF) described earlier to see if this trend holds true as the size of the nanowires increases. In Figure 4.5 the relative energies for 139 ZnO nanowires and nanotubes with diameters from 10 to 63 Å are compared. Again, as for the bulk and the surfaces, we conclude that the structural and energetic properties generated by the FF1 force-field for the these nanostructures are in better agreement with the reference method than those produced by the FF2.
Figure 4.5 – Single point SCC-DFTB versus ReaxFF calculation of formation energies relative to $E_{\text{wurtzite}}$ for ZnO nanowires. The structures were optimized with FF0, FF1 and FF2 and subsequently single point SCC-DFTB calculations were performed on them. Standard deviation errors are 0.17, 0.09 and 0.14 for FF0, FF1 and FF2, respectively.

4.2.4 Optimal force-field

The one-parameter minimization mechanism built into ReaxFF implies that to fit the parameters to some of the data-points, some others can be compromised. As will be discussed in the next chapter, some of the nanorods studied have jagged shaped walls which has a destabilizing effect on the energy. Therefore, we believe, inclusion of these structures in the training set for FF2, has negatively affected the quality of the results from FF2 when used for predicting other structures. As illustrated in Figure 4.2, FF2 produces slightly better results for some of the bulk and surface structures. This improvement, however, is insignificant in description of ZnO wurtzite nanorods.

In conclusion, we found the FF1 ReaxFF force-field to be the best one for our purpose, and the production runs described in Chapter 5 have all been based on FF1.
Chapter 5

Results and discussion

In this chapter we will discuss the results of applying the generated force-field (FF1) to ZnO nanosystems. Figure 5.1 illustrates the four different shapes of ZnO wurtzite nanosystems that were studied. All the nanosystems chosen are stoichiometric, i.e. the same number of zinc and oxygen atoms are present in the structure. For each of these structures we investigated how the various dimensions of the nanosystem affects its energy and structural relaxation.

In Section 5.1 we will discuss the results for nanowires (solid rods) and nanotubes (hollow rods) with a hexagonal cross-section [Figure 5.1(a,b)]. For the solid nanowires we studied systems with diameters of 10 to 90 Å and for the faceted nanotubes, which have larger surface to volume ratio, I studied how the wall thickness of these structures affects their properties. The largest nanowire contains 630 ZnO formula units in one layer and 1260 formula units per supercell. In Section 5.2, we will present the results of studying the size effects on energetic, structural and electronic properties for the parallelogram and for the elongated nanowire morphologies [Figure 5.1(c,d)].

5.1 Nanorods with a hexagonal cross-section

The ZnO wurtzite nanorods with a hexagonal cross-section are reported to be more stable than other nanotube morphologies. Hence, I first calculated properties for this shape of ZnO nanorods (and later also for hexagonal rods with jagged outer surfaces)*.

The nanostuctures were initially cut from the SCC-DFTB optimized wurtzite structure using a supercell model with application of periodic boundary conditions (PBCs). This means that to avoid interaction between the adjacent supercells, we need to apply a

*In this current work, we use ‘nanorods’ to collectively refer to the solid ZnO nanowires and the faceted ZnO nanotubes.
vacuum. Hence, we applied a vacuum size of ~40 Å in $a$- and $b$- directions modeling a structure growing in the $c$ direction.

After constructing each solid nanowire with a certain diameter from the wurtzite ZnO, we constructed hollow nanotubes having the same diameter as the solid nanowire but varying their wall thickness [wall thickness defined in Figure 5.1(b)]. This allows us to study the effect of wall thickness size on the stability, while keeping the outer diameter of the rod constant. In total 383 hexagonal nanorods were prepared and the MSE was used to create input for the ReaxFF program. FF1 was then used to optimize these structures.

Figure 5.2(a) shows the calculated formation energies for these structures as a function of the inverse number of ZnO formula units. It can be seen that for the largest systems, $\Delta E_f$ approaches zero, i.e. the rods behave more like wurtzite-bulk structure. In Figure 5.2(b) relative formation energies from ReaxFF and from the SCC-DFTB SP calculated for the FF1 optimized structures are compared. It can be seen in this figure that the energy difference between the two methods is larger for the less stable structures (irrespective of their size) and decreases as the systems increase in stability. To illustrate what the cause for the difference in behavior is, we present data for the hexagonal nanowires and nanotubes separately.

Figure 5.3 shows the calculated FF1 formation energies as a function of wall-thickness for tubular nanotubes. It can be seen that as the nanotube wall thickness
Figure 5.2 – Formation energies relative to $E_{\text{wurtzite}}$ of ZnO hexagonal nanowires and nanotubes as a function of inverse of number of ZnO formula units per supercell in the FF1 optimized nanorods. Figure a shows the formation energies relative to $E_{\text{wurtzite}}$ of ReaxFF optimized structures with their respective energies in SCC-DFTB using single point calculation for 191 solid and tubular nanorods. In b calculations performed on 383 structures in the production phase.

increases and approach the solid nanowire diameter, the structures become more stable, i.e. they approach the bulk energy and the energy penalty associated with relaxation of the surfaces decreases. It is also seen that for hexagonal nanorods with the same diameter, the formation energy is proportional to the wall-thickness (see Figure 5.4). However, there appears to be more than one trend for the energy vs wall thickness variation, which indicates, there are also other factors that should be considered. One such factor is the diameter of the nanotubes.

Figure 5.3 – Formation energies relative to $E_{\text{wurtzite}}$ for ZnO-wurtzite nanotubes with hexagonal cross-section as a function of inverse wall thickness, calculated with FF1 for the total of 356 nanotubes optimized with ReaxFF.
Figure 5.4 shows the FF1 calculated formation energies as a function of nanotube diameter and wall thickness for the ZnO. It can be seen that for nanotubes with a certain wall thickness (e.g. the red and cyan points with a wall thickness of ~20 Å), there are two distinct energy trends as the diameter changes. In the trend following the red data points, as the diameter is increased, the energy lowers and changes a little at $d > ~40$ Å. However, for the trend following the cyan data points, the formation energies remain mainly unchanged, which can be rationalized by considering that as the diameter increases at a constant wall thickness, the surface to volume ratio will not change much. Thus it appears that the energy cost to relax surfaces is offset by the stabilizing effect of bulk-like behavior of the structure.

The trend for the red points, on the other hand, suggests that some energetically unfavorable effect exists among these structures and is largest for structures with lower diameter. We found that this effect originates from the metastable nanotubes with jagged surfaces. As mentioned, there exist two different surface morphologies among our structures: smooth and jagged facets. The latter is metastable compared to smooth (1010) facets. This effect is also seen in the solid hexagonal nanowires. Figure 5.5 shows $\Delta E_{FF1}^f$ for the solid hexagonal nanowires as a function of their size. The nanowires with the jagged surfaces have higher formation energies than the smooth surfaced nanowires with similar size.

![Figure 5.4](image)

**Figure 5.4** – FF1 energies of formation relative to $E_{wurtzite}$ as a function of nanotube wall-thickness and diameter (defined in Figure 5.1). The red and cyan points belong to structures with the same wall thickness but different diameters. The red points are for the jagged-surfaced structures and the lower one for the smooth surfaced nanotubes.

Figure 5.6 shows the formation energies of FF1 optimized nanorods (one nanowire [the point with the lowest 1/n(ZnO)] and multiple nanotubes with the same diameter)
Figure 5.5 – Formation energies relative to $E_{\text{wurtzite}}$ of ZnO solid hexagonal nanowires as a function of the inverse of their number of ZnO units calculated with FF1. For the smaller structures SCC-DFTB Opt energies are also shown. Nanowires with jagged outer surface show higher energy while the smooth-surfaced nanowires are more stable. The smallest $1/n(\text{ZnO})$ value is 7.936 $\times 10^{-4}$ and corresponds to 1260 ZnO formula units in the supercell.

with $d = 90$ Å (a) and 87 Å (b). In Figure 5.6(a), all the structures have a jagged outer surface whereas they have a smooth outer surface in Figure 5.6(b). In both cases the solid nanowire is the most stable structure, which is always the case for the structures with the same diameter in our study. Introduction of inner walls (surfaces) in the nanotubes increases the surface to volume ratio, destabilizes the structure and the formation energies continue to increase as the size of the inner hole in the nanotubes increases (also seen in Figure 5.4). In both Figure 5.6(a) and (b), there are two trends for the nanorods with the same type of outer surface. If the inner-surface is jagged, the energy is increased even further. Hence, the formation energy trend for the nanotubes having jagged surfaces in both their inner and outer walls, is highest. For the structures with similar diameters, the jagged outer surface increases the energy more than the inner wall jagged surfaces. This is because the inner walls have less surface area compared to the outer walls of the nanotubes.

The structures with these metastable jagged surfaces are expected to be more reactive than the smooth faceted nanorods. This could be particularly interesting to study in molecular dynamics simulations of these nanorods in solutions. However, such calculations were beyond the scope of this project. Furthermore, we analyze how the SCC-DFTB Opt energies compare with FF1 results for these jagged surfaces. Figure 5.7 shows relative formation energies for the nanorods with jagged outer surfaces.
Figure 5.6 – FF1 and SCC-DFTB optimized formation energies relative to $E_{\text{wurtzite}}$ for nanorods (including a nanowire and nanotubes cut from it) with $d=90$ Å (a) and 87 Å (b) are shown. Two trends in $\Delta E_f$ exist for nanotubes cut from jagged (a) and smooth (b) surfaced structures which illustrates the destabilizing effect of the jagged surfaces. The trend lower in formation energy belongs to nanotubes with smooth inner walls and the higher trend is related to the jagged inner-surface nanotubes.

(a) and smooth outer surfaces (b) with $d=20$ and 23 Å, respectively. For the structures where there exists a jagged surface (be it on inner walls, outer walls or both) ‘FF1 energies for FF1-Opt structures’ (“FF1”) are lower than ‘SCC-DFTB energies for SCC-DFTB Opt structures’ (“SCC-DFTB opt”). This suggests that the FF1 is not well trained to describe such surfaces and thus further parameterization of the force-field to include a sufficient description of these surfaces at the same level as the SCC-DFTB ought to be done in the future.
Figure 5.7 – Formation energies relative to $E_{\text{wurtzite}}$ for hexagonal nanorods with jagged [(a); $d=20.4$ Å] and smooth [(b); $d=22.9$ Å] outer surfaces calculated using FF1 and SCC-DFTB SP (on FF1-Opt structures) and SCC-DFTB Opt. Snapshots of the nanorods corresponding to each data point (from left to right in the order of energy increase) are also shown. In (a) the nanorods have jagged outside surface which is energetically unfavorable. Nanotubes with jagged inner-wall surfaces [e.g. smallest nanotube in (b)] are also higher in energy in SCC-DFTB compared to FF1.

5.2 Parallelogram and elongated ZnO nanostructures

To investigate the effect of the shape of ZnO nanostructures, we constructed parallelogram and elongated shaped nanowires from SCC-DFTB optimized ZnO wurtzite phase in the same way described for the nanorods. As seen in Figure 5.8(a), the elongated structures are more stable than the parallelograms for the smaller structures, but the formation energies of the latter decrease faster as a function of rod diameter. In the elongated structures, the kinks on the edges of the structure are energetically unfavorable but as the size of the system, and thus the amount of bulk compared to the surfaces, increase in the system, the proportion of their contribution to the total energy of the system becomes smaller. For the parallelogram shaped structure, the edges have a greater destabilizing effect which is evident for the smaller structures but disappears quickly for the larger systems.

Also in Figure 5.8(b), we note that the energy description of the elongated structures in FF1 is very close to the SCC-DFTB Opt values. For the parallelogram ones, however, FF1 overstabilizes these structures which, as discussed for the jagged surfaces of the nanorods, indicates that the parallelogram shaped nanostructures are not well-described by the force-field and the training set should be updated to include these.

We also studied the electronic structure of these nanosystems. To acquire the electronic information, we performed SCC-DFTB SP calculations on the FF1 optimized structures and compared it with the corresponding values for the SCC-DFTB
Figure 5.8 – Energies of formation relative to $E_{\text{wurtzite}}$ for ZnO wurtzite parallelogram and elongated nanowires as a function of inverse of number of ZnO formula units in the structure. In (a) ReaxFF formation energies of these structures relative to $E_{\text{wurtzite}}$, optimized by FF1 are shown. It can be seen that smaller sized elongated nanowires are more stable than parallelogram shapes ones; In (b) and (c), formation energies calculated with ReaxFF, SCC-DFTB SP and SCC-DFTB Opt are compared for the elongated and the parallelogram shaped nanowires, respectively. FF1 calculated energies on elongated nanowires are closer to the reference SCC-DFTB method than those for the parallelogram shaped nanowires.

Figure 5.9 shows that there is a constant shift of around ~0.3 eV for the elongated nanowires bandgap energy compared to the optimized SCC-DFTB structures. For the parallelogram ones, the difference in bandgap energy is much larger (around 1.2 to 1.4 eV) which indicates that further extension of the training set and re-parameterization of ReaxFF will be needed to describe these structures.
Figure 5.9 – Bandgap energies as a function of nanowire diameter calculated with SCC-DFTB SP on FF1 optimized structures and SCC-DFTB results with optimization. FF1 calculated bandgap for larger elongated nanowires approaches that of ZnO bulk wurtzite with a constant shift of ~0.3 eV to SCC-DFTB opt. This effect is not visible for the parallelogram shaped nanowires.
Chapter 6

Concluding remarks and future work

This master thesis describes the development of a seamless scheme to parameterize ReaxFF with SCC-DFTB results. The approach extends the possibilities to study large ZnO structures, including various bulk polymorphs, surfaces and nanostructures. The key to an efficient parameterization is to constantly take one step back to the reference method and update the training set. Thus, at each stage, the generated ReaxFF force-field was tested on structures absent from the training set and the results were fed back to SCC-DFTB to ensure they sufficiently agree. This circular process requires fast and efficient maintenance of the calculated data. Hence, I developed the MSE to automate SCC-DFTB calculations, prepare ReaxFF training set and transfer force-field results back to the SCC-DFTB.

The MSE is developed in a modular fashion, therefore it can be used generally in study of different materials. However, in this project we applied the scheme on ZnO structures with various shapes and sizes. Initially I prepared a training set for ReaxFF by including the ZnO crystal phases and non-polar surfaces that are most important in the description of ZnO solid and tubular nanorods. A parameterization was followed and continued until convergence was reached. I used the resulting force-field to study ZnO nanorods and validated that a one-to-one correspondence in energy and structure between the two levels of theory holds. To this end, I compared the energetic and structural properties of ReaxFF results with those of the SCC-DFTB. I then expanded the training set with ZnO nanorods for which the ReaxFF was insufficient. I then calculated properties for ZnO bulk polymorphs and non-polar surfaces using the different force-fields and chose the optimal one for the next step. Overall this force-field was able to satisfactorily reproduce SCC-DFTB results for ZnO bulk crystals, non-polar surfaces and nanorods. The resulting force-field was then used to study ZnO nanostructures at experimentally relevant sizes.
The study of ZnO with different shapes showed that in the case of nanorods with a hexagonal cross-section, the stability of the system is inversely related to its surface to volume ratio. Therefore, the larger systems, and nanotubes with larger wall-thickness were more stable. We also found that the morphology of the outer and inner surface of the nanotubes, as well as the outer facet of the solid nanowires, strongly affect their energetic properties. To this end, I discovered that nanosystems with a jagged surface cut are metastable compared to the more stable smooth walled structures. As for the nanowires with parallelogram and elongated hexagonal cross-sections, we find that the sharp edges of the latter is the main reason for their instability whereas the energetically unfavorable effect of surface kinks for the former vanishes as the structure grows in diameter. I also studied the bandgap energy of these structures by performing static SCC-DFTB calculations for the ReaxFF optimized structures and observed that the difference between the two levels of theory is larger for the parallelogram shaped nanowires.

Finally, by utilizing the MSE, this thesis describes the creation and development of a procedure that to a large degree manages to maintain the accuracy associated with the SCC-DFTB level and transfer it to the more coarse-grained ReaxFF force-field which was applied to ZnO nanowires. This strategy allowed for an order of magnitude larger systems and sizes to be treated. Thus, the fast and seamless automation environment can be deemed to have reached its goal within the scope of the project.

Future work in this field should be centered around extending the MSE to accommodate more computational methods at different levels of theory. To achieve this, several steps need to be taken so as to add the flexibility necessary to the MSE both in terms of computational details and graphical features. The tools provided by the ASE package can also be used in such an integrated environment in which theoretical methods of different temporal and dimension scales can communicate in a straightforward fashion.


APPENDICES
Appendix A

The MSE Software code documentation

In the following we present the documentation for the MSE software. Description of software modules in terms of classes, variables and functions are provided.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 Module MSE</strong></td>
<td>70</td>
</tr>
<tr>
<td>1.1 Variables</td>
<td>70</td>
</tr>
<tr>
<td>1.2 Class mainWindow</td>
<td>70</td>
</tr>
<tr>
<td>1.2.1 Methods</td>
<td>70</td>
</tr>
<tr>
<td>1.2.2 Instance Variables</td>
<td>73</td>
</tr>
<tr>
<td>1.3 Class importFrame</td>
<td>74</td>
</tr>
<tr>
<td>1.3.1 Methods</td>
<td>74</td>
</tr>
<tr>
<td>1.3.2 Instance Variables</td>
<td>76</td>
</tr>
<tr>
<td>1.4 Class messageDialog</td>
<td>77</td>
</tr>
<tr>
<td>1.4.1 Methods</td>
<td>77</td>
</tr>
<tr>
<td>1.5 Class InitialBulks</td>
<td>78</td>
</tr>
<tr>
<td>1.5.1 Methods</td>
<td>78</td>
</tr>
<tr>
<td>1.5.2 Properties</td>
<td>79</td>
</tr>
<tr>
<td>1.6 Class calculateMain</td>
<td>79</td>
</tr>
<tr>
<td>1.6.1 Methods</td>
<td>80</td>
</tr>
<tr>
<td>1.6.2 Properties</td>
<td>81</td>
</tr>
<tr>
<td><strong>2 Module ReaxCalculator</strong></td>
<td>82</td>
</tr>
<tr>
<td>2.1 Functions</td>
<td>82</td>
</tr>
<tr>
<td><strong>Index</strong></td>
<td>85</td>
</tr>
</tbody>
</table>
1 Module MSE

1.1 Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| structures | Global dictionary of structure names for which checkbuttons will be defined. Caution: Changing this changes other parts of the program too.  
   **Value:** `{‘BCT’: ‘BCT’, ‘CC’: ‘CsCl’, ‘CUB’: ‘cubane’, ‘GRF’: ‘gra...` |
| symbols | Atoms for which we have skf znopt SCC-DFTB parameter set files are currently available (April 2014)  

1.2 Class mainWindow

Tkinter.Misc

Tkinter.Wm

Tkinter.Tk

MSE.mainWindow

This is the main GUI window of the Multiscale Modeling Environment. Inherits Tk.Tk class. Contains Checkbuttons to choose which phases to be included in the calculations and whether charges and skewing of the unit-cell angles for that phase should be included. The Menu bar includes items to create initial configurations for the nanostructures with some options, help menu cascade to userGuide and about pages. Calculate button performs DFTB calculations on the selected structures and writes Reax input files. Bulk modulus button runs SCC-DFTB to calculate bulk modulus for the selected structures. Import button opens a window to import pre-calculated SCC-DFTB results into the training set.

1.2.1 Methods

```python
__init__(self, parent)
```

Constructs items for the mainWindow window.

Overrides: Tkinter.Tk.__init__

```python
addMenu(self, parent)
```

Adds menu bars to mainWindow parent tk instance. Currently there are buttons to create nanostructures, help and exit the program.
**nanoFrame(self)**

Creates instance of nanoFrame class. Appends nanoStructures to newStructure list onClose. The nanoFrame class is yet to be fully implemented into the mainWindow.

**about(self)**

Display contents of about.txt in a Tkinter text widget.

**userGuide(self)**

Opens userGuide.pdf created by epydoc.

**bulkModulus(self, mode='ACscan')**

Runs SCC-DFTB volume-scan with geometry optimization for selected structures. It tries to read previously stored $structure+_BM.traj (if mode is pressure) or $structure+_ACscan.traj (if mode is ACscane) file. If not-exists >> performs calculations.

If the structures is cubic: isotropic scan is performed otherwise, uses ACscan function to perform A/C scanning for each given volume.

**mode:**

ACscan: changes A and C cell parameters for each volume thus giving non-isotropic scan of V/E pressure: changes volume but applying negative and positive pressures in DFTB+

**eosMurnaghan(self, name, traj)**

Calculates Bulk Modulus (BM) using Murnaghan equation of state and plots data.

**calcBMpressure(self, v_min, st_min, calcs, structure, structs)**

Return SCC-DFTB optimized structures under different pressures.

**v_min:** float, optimized structure volume

**st_min:** Atoms, optimized structure

**calcs:** list of SCC-DFTB calculator objects

**structure:** str, name of bulk phase as in structures dictionary

**structs:** a list of Atoms to include in bulk modulus calculation.

If the energy of the optimized structure is lower than minimum energy structure for that crystal, phase transition has occurred and it returns.
setMainDir(self)
Return values on symbols optionMenus: om1, om2.
>> Example: Zn, O. creates the directory for the selected elements on an alphabetical order. Delimiter: underscore: e.g. O_Zn

isSelected(self)
Return bool. Checks if any of the checkbuttons for the structures is selected.

btnCalcClick(self)
Is evoked when the click event on the 'Calculate' button occurs, checks what is selected in the main window, checks availability of previous calculations, if not available instantiates InitialBulkClass to perform SCC-DFTB.

calcReax(self, trajs, calcMode)
Creates folder and copies initial ReaxFF input files. If params Entry is set to zero iopt and params files are not created. Modifies ReaxFF params file for parameterization/calculation. Repeats original params file to contain the number of lines as in params Entry at GUI.

trajs: dictionary of Atoms described in read_trajs method docstring.

>>calcMode: str, mode (minima, atomic) in ase.io.geo.write_geo function to write trajs to geo, trainset.in

update_line(self, h1, x, y)
Animates the graph as it’s being read from fort.83. Compatible with later versions of Matplotlib.

setWeights(self, trajs)
Adds 'weight' key to Atoms.info dictionary in trajs (described in read_trajs method). Weights are set by the following formula:

\[ w_{\text{min}} + (w_{\text{min}} \times |\text{emin} - e|^{0.1}) \]

where, \( w_{\text{min}} \) is the weight given at GUI for each structure; \( \text{emin} \) is the energy of the minimum energy structure; \( e \) is energy of structure at hand. This allows for ReaxFF higher weights for structures further from equilibrium.

validateWeightEnt(self, action, index, value_if_allowed, prior_value, text, validation_type, trigger_type, widget_name)
Returns True if text given in weight Entry is in '0123456789'. False for any other value.

>>text: key-stroke value in the Entry.

onClose(self)
Quit program
hide(self)
Hide mainWindow when subFrame is evoked.

openFrame(self)
Generates an instance of import frame.

show(self)
Unhide mainWindow when sub-frame is closed.

1.2.2 Instance Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newStructure</td>
<td>List: stores the imported structures (extensions) in tuple &gt;&gt; [(atoms1.info['name'],atoms1),...].</td>
</tr>
<tr>
<td>mainDir</td>
<td>String: Main directory. Based on alphabetical order of Atoms &gt;&gt; O,Zn.</td>
</tr>
<tr>
<td>whichBulks</td>
<td>Dictionary: Keeps the info on which of selected structures checkbuttons.</td>
</tr>
<tr>
<td>whichWeights</td>
<td>Dictionary: Keeps the respective weights which are used in ReaxFF trainset.in file.</td>
</tr>
<tr>
<td>whichAngles</td>
<td>Dictionary: selected structures angular scans.</td>
</tr>
<tr>
<td>whichCharges</td>
<td>Which structures’ charges should be in the training-set.</td>
</tr>
<tr>
<td>frameBulk</td>
<td>Frame: holds structure names and checkbuttons.</td>
</tr>
<tr>
<td>frameAtm</td>
<td>Frame: holds option menus for the atoms.</td>
</tr>
<tr>
<td>om1</td>
<td>OptionMenu: contains chemical symbols.</td>
</tr>
<tr>
<td>parameterVar</td>
<td>int: signifies the number of lines that in the ReaxFF PARAMS file.</td>
</tr>
<tr>
<td>parameterEnt</td>
<td>Entry: Setting params Entry box validation command to vcmd.</td>
</tr>
<tr>
<td>omMode</td>
<td>Mode of preparing ReaxFF input files. It could be set to minima (formation energies) versus the minimum) or atomic (atomization energies)</td>
</tr>
<tr>
<td>path</td>
<td>Creates checkbuttons, labels and entries in the frameBulk of the mainWindow</td>
</tr>
<tr>
<td>root</td>
<td>Tkinter module instance.</td>
</tr>
<tr>
<td>frame</td>
<td>Frame: holds import, calculate, bulk modulus and exit buttons.</td>
</tr>
</tbody>
</table>
1.3 Class importFrame

Tkinter.Misc

Tkinter.BaseWidget

Tkinter.Wm

Tkinter.Toplevel

MSE.importFrame

Class for importing DFTB pre-calculated structures into training set. SCC-DFTB output files are converted to ASE compatible Trajectory format and stored for further use in the extensions folder created in the root directory of the current run of the program:

e.g.: O\text{Zn}/extensions.

1.3.1 Methods

\_\_init\_\_(self, parent, newStructure)

Constructor. >>>newStructure: a dictionary containing the manually added structures that will be in ReaxFF training-set.

The values in this dictionary are maintained and if it is not empty, the corresponding items from the ListBox are highlighted everytime the import window is shown.

>>>parent: the mainWindow instance.

Overrides: Tkinter.BaseWidget..\_\_init\_\_.

tryint(self, s)

Return int(s) if s is int, else return s.

alphanum_key(self, s)

Turn a string into a list of string and number chunks. ”z23a” -> [”z”, 23, ”a”]

sort_nicely(self, l)

Sort the given list in the way that humans expect, e.g.: built-in Python sort function puts Surf\text{10}_\text{ZnO} before Surf\text{02}_\text{ZnO}. This method makes it easier to handle sorting.

onClose(self)

Destroy importFrame instance and unhides parent frame. Extensions selected in instance’s `select` ListBox are passed to mainWindow in the newStructure list.

whichSelected(self)

Return index of selection on select ListBox. If nothing is selected it passes.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>addEntry</strong>(self)</td>
<td>Bind to Add button on import frame GUI. Appends selected structure to the select ListBox.</td>
</tr>
<tr>
<td><strong>updateEntry</strong>(self)</td>
<td>Accepts single or multiple items. Updates weight and reference for multiple selected listbox items. Updates information in newStructure list and the stored trajectory files. For single selected item, if fileVar is changed, source .traj file is read and removes old files after updating.</td>
</tr>
<tr>
<td><strong>deleteEntry</strong>(self)</td>
<td>Removes single/multiple items from select listbox, newStructure list and hard-disk. Prompts a yes/no question.</td>
</tr>
<tr>
<td><strong>loadEntry</strong>(self, e)</td>
<td>Reads data from selected item on listbox and writes them to respective fields on importFrame instance.</td>
</tr>
<tr>
<td><strong>validateName</strong>(self, d, i, P, s, v, V, W)</td>
<td>Returns True and the character is printed in the name Entry on keypress if: it is not whitespace, asterisk or percent sign and that it is upperCase naming.</td>
</tr>
<tr>
<td><strong>makeWindow</strong>(self)</td>
<td>Instantiates and packs widgets into Import Frame.</td>
</tr>
<tr>
<td><strong>setSelect</strong>(self)</td>
<td>Updates listbox from newStructure. Invoked by add, update or delete events.</td>
</tr>
<tr>
<td><strong>addFile</strong>(self)</td>
<td>Prompts a browse window (Tkinter askopenfilename object). Filetypes: detailed.out, gen, <em>.</em></td>
</tr>
<tr>
<td><strong>readNewStructure</strong>(self)</td>
<td>Reads selected detailed.out file, checks the output geometry file (gen file) if geometry is converged in detailed.out file. Total energy, net charges and forces are also read from detailed.out. stress, magentic-momentums are defined as arrays of zero and along with charges and total energy are assigned to ase.calculator.singlepoint.SinglePointCalculator. Adds geometry and the calculator (containing total energy, charges and forces) data to Atoms object. Returns None if geometry not converged in detailed.out</td>
</tr>
</tbody>
</table>
write_info(self, atoms)

Return Atoms with updated info directory.

reference: from refVar

variable name : from nameVar variable

fname : from fileVar

variable remark : String : chemical formula of the structure + name + fname + reference + hashed Atoms (15 digits)

>> e.g. 'O72Zn72 SURF_4LR_110 /home/Surfaces/detailed.out reference: minima 90a21f8b189cb05'

write_charge : bool: True if chargeVar is selected

weight : from weightEnt, default weight is 1

write_traj(self, atoms)

atoms : ase.Atoms object. Writes atoms to PickleTrajectory: .traj

1.3.2 Instance Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Program root directory.</td>
</tr>
<tr>
<td>mainDir</td>
<td>Current calculation directory &gt;&gt;&gt; e.g. O,Zn</td>
</tr>
<tr>
<td>exPath</td>
<td>Imported structures directory &gt;&gt;&gt; e.g. O,Zn/extensions</td>
</tr>
<tr>
<td>newStructure</td>
<td>&gt;&gt;&gt; [[structure name, Atoms obj]...]</td>
</tr>
<tr>
<td>mySelection</td>
<td>list of int: indices of extensions selected in a previous instance of newStructureFrame.</td>
</tr>
<tr>
<td>frame1</td>
<td>Tkinter Frame: main frame for importFrame instance.</td>
</tr>
<tr>
<td>nameLbl</td>
<td>Tkinter Label: structure name label ('Name').</td>
</tr>
<tr>
<td>nameVar</td>
<td>String var: structure name variable</td>
</tr>
<tr>
<td>vcmd</td>
<td>validation command for name Entry.</td>
</tr>
<tr>
<td>name</td>
<td>Tkinter Entry: name Entry.</td>
</tr>
<tr>
<td>fileLbl</td>
<td>Tkinter Label: detailed.out URL label.</td>
</tr>
<tr>
<td>fileVar</td>
<td>String var: .traj URL variable.</td>
</tr>
<tr>
<td>fname</td>
<td>Entry: readonly Entry for file URL.</td>
</tr>
<tr>
<td>btnAddFile</td>
<td>Tkinter Button: Add button.</td>
</tr>
<tr>
<td>refLbl</td>
<td>Label: 'Reference' mode label.</td>
</tr>
<tr>
<td>refVar</td>
<td>String var: reference mode variable (atomic, minima or any of the bulk structures)</td>
</tr>
<tr>
<td>refs</td>
<td>list: list of reference modes.</td>
</tr>
<tr>
<td>om1</td>
<td>Tkinter OptionMenu: holds reference states.</td>
</tr>
<tr>
<td>weightLbl</td>
<td>Label: 'Weight' label.</td>
</tr>
<tr>
<td>weightVar</td>
<td>Double var: weight variable.</td>
</tr>
<tr>
<td>weightEnt</td>
<td>Entry: weight Entry.</td>
</tr>
<tr>
<td>chargeVar</td>
<td>Integer var: charge inclusion/exclusion variable.</td>
</tr>
<tr>
<td>chargeLbl</td>
<td>Label: 'Charges' label.</td>
</tr>
</tbody>
</table>

continued on next page
1.4 Class messageDialog

MSE.messageDialog

Class for versatile message/question prompts. Pops open a customized message/question dialog on top of parent frame.

text: str, text of the question/warning to be shown in the dialog

question: bool, if True, question pops up with yes and ok buttons, else warning message is shown with ok button

1.4.1 Methods

```python
__init__(self, parent, txt, question=False)
```

parent: Tkinter Frame: frame in which the messageDialog has been instanciated.

```
txt: string: text to be displayed on the message dialog.
```

```
question: bool: if True -> yes/no buttons are added to the dialog. default: False. only ok button is added to the dialog.
```

Overrides: Tkinter.BaseWidget.__init__

```python
ok(self)
```

if ok button is clicked, the prompt message box exits.
yes(self)
Method of messageDialog class. If delete question prompt confirmed: item is removed from ListBox (look at deleteEntry method)

1.5  Class InitialBulks

object    
MSE.InitialBulks

Class to generate initial guesses for the crystal structures and run single point dftb calculation on the structure starting with 1X1X1 k-points and increasing the number of k-points until convergence is reached. >> total energy difference between two consecutive calculations should be lower than 0.001 eV in order for it to be considered converged.

1.5.1  Methods

__init__(self, atm1, atm2)
atm1 and atm2 are strings for elements selected in mainWindow instance.
Overrides: object.__init__

wurtzite(self)
method to create initial guess for wurtzite bulk structure. Initial parameters are:
a= 3.2494, c= 5.2042, u= 0.3819 Angstrom, alpha= beta= 90, gamma= 120 degrees (from lit.)
ase.spacegroup.crystal class is used with spacegroup 186.

rocksalt(self)
method to create initial guess for rocksalt bulk structure. Initial parameters are:
a= 4.3484 Angstrom alpha= beta= gamma= 90 degrees (from lit.)
ase.spacegroup.crystal class is used with spacegroup 225.

zincblende(self)
method to create initial guess for zincblende bulk structure. Initial parameters are:
a= 4.5446 Angstrom, alpha= beta= gamma= 90 degrees (from lit.)
ase.spacegroup.crystal class is used with spacegroup 216.
CsCl(self)
method to create initial guess for CsCl bulk structure. initial parameters are:
\(a = 2.7106\) alpha= gamma= 90 degrees (from lit.)
ase.spacegroup.crystal class is used with spacegroup 221.

graphitic(self)
method to create initial guess for graphitic bulk structure. initial parameters are:
\(a = 3.3259, c = 4.868\) Angstrom alpha= beta= 90, gamma= 90 degrees (from lit.)
ase.spacegroup.crystal class is used with spacegroup 194.

BCT(self)
method to create initial guess for BCT bulk structure. initial parameters are:
\(a = 5.5052, c = 3.2448, u = 0.3674\) Angstrom alpha= beta= gamma= 90 (from lit.)
ase.spacegroup.crystal class is used with spacegroup 136.

cubane(self)
method to create initial guess for cubane bulk structure. initial parameters are:
\(a = 6.29, u = 0.326\) Angstrom alpha= beta= gamma= 90 (from lit.)
ase.spacegroup.crystal class is used with spacegroup 217.

calcKP(self, system, structure)
method to calculate optimal number of k-points iteratively.

>>system: Atoms object, initial crystal
>>structure: str, crystal type name

1.5.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calc</td>
<td>single point DFTB calculator.</td>
</tr>
<tr>
<td></td>
<td>Value: Dftb(label= 'dftb', write_dftb= True,</td>
</tr>
<tr>
<td></td>
<td>scc= True, latticeo...)</td>
</tr>
</tbody>
</table>

1.6 Class calculateMain

object

MSE.calculateMain
Class to run SCC-DFTB calculations to optimize structures, do volume-energy scan, angular scan and isolated atoms calculations.

1.6.1 Methods

```python
__init__(self, atm1, atm2)
atm1 and atm2 are read from the mainWindow interface and spin-polarized SCC-DFTB is performed on them.

Environment variables DFTB_PREFIX and DFTB_COMMAND are set to the corresponding folders where parameter sets and dftb+ program are located, respectively. Overrides: object.__init__(inherited documentation)
```

```python
setRef(self, elems)
Method to run SCC-DFTB on selected elements.
>> e.g. for wurtzite Zn2O2, isolated O and Zn atoms are calculated.
```

```python
optimize(self, structure, system, optK)
Runs SCC-DFTB optimization of lattice and geometry to acquire the equilibrium bulk structure. Uses ase Dftb calculator object with 225 minimization steps, fermi-temperature=0 k.

>>>optK: int, k-point mesh from calcKP method

>>>structure: str, name of the selected bulk to create folders and store information in atoms.info dictionary

>>>system: Atoms, the initial crystal structure created in InitialBulks class
```

```python
volumeScan(self, structure, crystal)
Runs single-point SCC-DFTB on crystals by expanding/contracting the equilibrium unit-cell +/-5%.

>>>crystal: optimized Atoms object

>>>structure: str, bulk name
```

```python
angleScan(self, atoms, structure, calc)
Skews the bulk structures to see the bond strength. This method changes cell coordinates and angles with all combinations of the three unit-cell angles (alpha, beta and gamma).

>>>atoms: Atoms, optimized Atoms object

>>>structure: str, bulk phase name ase in the main window

>>>calc: ase Dftb calculator object
```
cartesian(self, arrays, out=None)

Returns an array of all combinations of arrays. It is used for angular scanning. Out is a two
dimensional ndarray containing combinations of all members of arrays.

Example: cartesian([[2,3],[1,5]]) -> out=array([[2, 1], [2, 4], [3, 1], [3, 4]])

read_trajs(self, whichStruct, whichAngle, whichCharge)

Returns a dictionary with a defined format to be written to ReaxFF input files. The items
in whichStruct, whichAngle and whichCharge are included in the final dictionary (trajs).

>>> trajs = {structureName:[[descr,remark,Atoms],...],...} Final dictionary (trajs) values are
comprised of a list for each Atoms object. The list items are: descrp: [structureName,
number is sequence (e.g. in volume-scan), calcType, empirical formula]

>>> e.g. ['RS', '3', 'B', 'ZnO'] remark: string of chemical formula, structure Name, index
(e.g. in volume-scan),calculator name, hashed Atoms.

>>> e.g. Zn4O4 rocksalt 3/11 bulk from singlepoint calculation. 524798740937483' Atoms
object. This formatting is old and later I started using Atoms.info dictionary to transfer
data on the object.

calcTypes : calcTypes = {3:('B', 'bulk'), 2:('S', 'surface'), 1:('R', 'rod'), 0:('M', 'Molecule')}
defined by periodic boundary conditions:

>>> e.g. periodic structure in x and y (2) is considered surface etc.

1.6.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>structures</td>
<td>dictionary of bulk structure abbreviations and names.</td>
</tr>
</tbody>
</table>
2 Module ReaxCalculator

Extension to ASE as a calculator for ReaxFF. It can read and write structures in GEO format and also write structures in the trainset.in format. In the MSE environment it is called in the GUI software to prepare the training-set for ReaxFF parameterization from the SCC-DFTB results. However, it can be called separately to perform the same task outside the MSE environment.

Refer to ReaxFF manual for GEO and trainset.in descriptions. Multiple snapshots are supported.

Can be called from ase.io.geo.py:

if atoms are not written by MSE software, descrp, remark and rutype list should be created to include descriptions of each Atoms.

Otherwise, if not passed the function creates them automatically (not recommended).

e.g.: structures = read(‘data.traj@0:’)

write(‘geo’,structures, format=’geo’,trainset=True, descrp=descrpList, remark=remarkList,rutype=rutypeList,mode=’minima’)

To read the geo or fort.90 files back into ASE readable Atoms:

e.g.:
geoStructs = read(’fort.90@0:’,format=’geo’)

2.1 Functions

<table>
<thead>
<tr>
<th>read_geo(fileobj, index=-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Atoms in the fileobj (geo format) in the range chosen. Reads structure in GEO format (refer to ReaxFF manual). Also reads fort.90 and assign Reax energies to atoms objects information dictionary in eV. Multiple snapshot are allowed seperated by END keyword at the end of each snapshot. This method can either be called directly via ase.io.geo.read_geo or indirectly via ase.io.read. In the latter case the format=’geo’ attribute should be passed when calling the method</td>
</tr>
</tbody>
</table>

| >>fileobj: if str, reads file |
| >>index: int, default=-1 only reads the last snapshot in geo file. use @x:y after filename to determine which objects to read from the input file. @0: reads all Atoms Objects in the input file. |

<table>
<thead>
<tr>
<th>fractional_coordinates(atoms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reads the atoms object with reax_coordinates and converts them to fractional_coordinates. This allows for hexagonal (and maybe other type) cell to be correctly read from geo file. It uses Atoms object’s has a set_scaled_position method to set the fractional coordinates.</td>
</tr>
</tbody>
</table>

| >>atoms: Atoms, read from geo file with reax coordinates. |
write_geo(fileobj, images, descrp=None, rutype=None, remark=None, trainset=False, mode='atomic')

Write structure in GEO format (refer to ReaxFF manual). Multiple snapshots are allowed. If images is a dictionary then format is:

```
>> images =
{
    'struct_0':
        [ [descrp_0_0],[remark_0_0],Atoms_0_0]
        ,...,
        [descrp_0_N],[remark_0_N],Atoms_0_N]
    ,...,
    'struct_N':
        [ [descrp_N_0],[remark_N_0],Atoms_N_0]
        ,...,
        [descrp_N_N],[remark_N_N],Atoms_N_N]
}
```
else if images is a list then it is a list of ASE Atoms objects.

```
>> descrp: list of str, if not None, verbatimly sets the DESCRP for each Atoms object as specified in this list
>> rutype: list of str, if not None, sets RUTYPE for each of the Atoms in the geo file
>> remark: list of str, if not None, sets REMARK for each of the Atoms in the geo file
>> trainset: bool, if True: writes trainset.in file
>> mode: str,
    default is atomic: writes the energies vs element.
    minima: writes the energies vs global minimum energy
```

write_trainset(images, mode='atomic')

Writes the trainset.in file for ReaxFF parameterization from dictionary of Atoms objects.

```
>> images: dictionary of ReaxFF to be written into the trainset.in file
>> mode: as explained for the read_geo method
```
### Functions Module ReaxCalculator

<table>
<thead>
<tr>
<th><strong>reax_hash</strong>(<em>atoms</em>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the first 15 characters of a hash of the pickled string representing an atoms object. Used to generate unique keys for an Atoms object. There is some concern that there could be collisions in just the first 15 characters of the hash, but 15 characters is the limit of what the Reax code can handle. This I only used as part of the remark for each Atoms since it is susceptible to changes as the pickling method changes. This means that I think it cannot be trusted to be name DESCRP of the Atoms object.</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>For the DESCRP I implemented another strategy which is more human read-able in both geo and trainset.in. However, I think it is best to store the keys (descrp, rtype, remark) in Atoms.info dictionary for all the structures and not only the imported ones (as implemented now). This way I can make the code more straight forward and modular (naming can be done in another function in MSE).</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>reax_coordinates</strong>(<em>atoms</em>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns an array of reax coordinates. For this I used a source on the internet. Converting the fractional coordinates that Atoms.get_scaled_positions() function returns to the reax_coordinates is important for non-cubic lattice types (e.g. hexagonal wurtzite).</td>
</tr>
</tbody>
</table>

---

84
Index

MSE (module), 70
MSE.calculateMain (class), 79
  MSE.calculateMain.angleScan (method), 80
  MSE.calculateMain.cartesian (method), 80
  MSE.calculateMain.optimize (method), 80
  MSE.calculateMain.read_trajs (method), 81
  MSE.calculateMain.setRef (method), 80
  MSE.calculateMain.volumeScan (method), 80
MSE.importFrame (class), 73–77
  MSE.importFrame.addEntry (method), 74
  MSE.importFrame.addFile (method), 75
  MSE.importFrame.alphanum_key (method), 74
  MSE.importFrame.deleteEntry (method), 75
  MSE.importFrame.loadEntry (method), 75
  MSE.importFrame.makeWindow (method), 75
  MSE.importFrame.onClose (method), 74
  MSE.importFrame.readNewStructure (method), 75
  MSE.importFrame.setSelect (method), 75
  MSE.importFrame.sort_nicely (method), 74
  MSE.importFrame.tryint (method), 74
  MSE.importFrame.updateEntry (method), 75
  MSE.importFrame.validateName (method), 75
  MSE.importFrame.whichSelected (method), 74
  MSE.importFrame.write_info (method), 75
  MSE.importFrame.write_traj (method), 76
MSE.InitialBulks (class), 78–79
  MSE.InitialBulks.BCT (method), 79
  MSE.InitialBulks.calcKP (method), 79
  MSE.InitialBulks.CsCl (method), 78
  MSE.InitialBulks.cubane (method), 79
  MSE.InitialBulks.graphitic (method), 79
  MSE.InitialBulks.rocksalt (method), 78
  MSE.InitialBulks.wurtzite (method), 78
  MSE.InitialBulks.zincblende (method), 78
MSE.mainWindow (class), 70–73
  MSE.mainWindow.about (method), 71
  MSE.mainWindow.addMenu (method), 70
  MSE.mainWindow.btnCalcClick (method), 72
  MSE.mainWindow.bulkModulus (method), 71
  MSE.mainWindow.calcBMpressure (method), 71
  MSE.mainWindow.setWeights (method), 72
  MSE.mainWindow.show (method), 73
  MSE.mainWindow.update_line (method), 72
  MSE.mainWindow.userGuide (method), 71
  MSE.mainWindow.validateWeightEnt (method), 72
  MSE.messageDialog (class), 77
    MSE.messageDialog.ok (method), 77
    MSE.messageDialog.yes (method), 77
ReaxCalculator (module), 82–84
  ReaxCalculator.fractional_coordinates (function), 82
  ReaxCalculator.read_geo (function), 82
  ReaxCalculator.reax_coordinates (function), 84
  ReaxCalculator.reax_hash (function), 83
  ReaxCalculator.write_geo (function), 82
  ReaxCalculator.write_trainset (function), 83

85