How to Write Input File for "createAtoms"

X. W. Zhou*

Mechanics of Materials Department,

Sandia National Laboratories, Livermore, California 94550, USA

(Dated: June 10, 2008)

I. INTRODUCTION

An atomistic simulation requires an initial atomic configuration. Most simulation packages provide capabilities to create simple crystalline atomic configurations. More complicated configurations, however, often need to be created using external tools. "createAtoms" was designed to provide one of such tools.

"createAtoms" can be used to create a crystalline computational system containing regimes with different crystal structures, crystallographic orientations, compositions, and other defects. Current version of "createAtoms" is based upon a rectangular computational box cell, and requires that the crystals have three perpendicular crystallographic orientations. In addition to cubic and orthogonal crystal structures, hexagonal structure also satisfies this requirement. For example, the [0001], [$\bar{1}$100], and [11$\bar{2}$0] directions of a hexagonal crystal structure are mutually perpendicular. As a result, "createAtoms" can be used to create any cubic, orthogonal, and hexagonal elemental or compound crystal structures.

II. COMPILING AND EXECUTION

"createAtoms" is written in a single FORTRAN file "createAtoms.f" (with a separate header file "createAtoms.h"). The program can therefore be compiled using a single command "f77 createAtoms.f". On some machines (e.g., uniaxial), f90 compiler (instead of f77) needs to be used for the code to run properly. The compiled code can be executed using "executable < input_file", where "input_file" is the name of an input file that contains a series of assignment statements defining the crystalline system to be created. In the following, we document the use of these assignment statements. As an exercise, we also demonstrate how to create a hexagonal wurtzite GaN nanowire.

III. INPUT FILE ORGANIZATION

The assignment statements in the input file are grouped into various cards that starts with "&cardname" and ends with "&end". Each card performs a well-defined function. Cards must follow a designated sequence. The names of cards and their sequence are shown in Fig. 1. Note here that "latcard", "subcard", "defcard", and "hitcard" can be used multiple times whereas all the other cards are used only once. For convenience, the repeated

cards are represented by "..." in Fig. 1. "createAtoms" will continuously read in "latcard", "subcard", "defcard", and "hitcard" until an empty version of the card (marked by red frame in Fig. 1) is encountered. If such an empty card is not provided, the program will abort with an error message.

Each "latcard" is associated with one or more "subcard"s and one or more "defcard"s. The "subcard"s allow complicated crystal structures and compounds to be defined, and the "defcard"s allow the defects (vacancies, solute atoms, and surfaces etc.) to be created. The combination of latcard-subcard-defcard then allows a local region of the system to be defined in terms of phase, crystal structure, crystal orientation, composition, and defects, etc.

Each card is essentially composed of variables. Users assign values to these variables via the input file. Some variables will adopt default values if they are not assigned. We will explain these in the following card by card.

IV. MAINCARD

"maincard" contains six variables: "ntypes", "amass", "ielement", "ilatseed", "perlb", and "perub". Here "ntypes" is an integer representing the total number of different atom species used in the simulation, "perlb" and "perub" are two vectors representing respectively the higher and lower bounds of the system cell, ilatseed is a random number seed used to randomize atoms for force decomposed parallel molecular dynamics algorithm, amass and ielement are two lists representing respectively mass and atomic number of different atom species. To create a hexagonal wurtzite GaN nanowire, for example, we would use the following "maincard":

$\&maincard$

$ntypes = 2$

$perub = 903.02776, 54.576777, 53.557606$

$perlb = -0.5, -0.5, -0.5$

$ilatseed = 21$

$amass = 69.72, 14.00674, 1.0, 1.0, 1.0$

$ielement = 31, 7, 1, 1, 1$

$\&end$

Here ntypes = 2 because we only have two different atoms: Ga and N. In this example, we

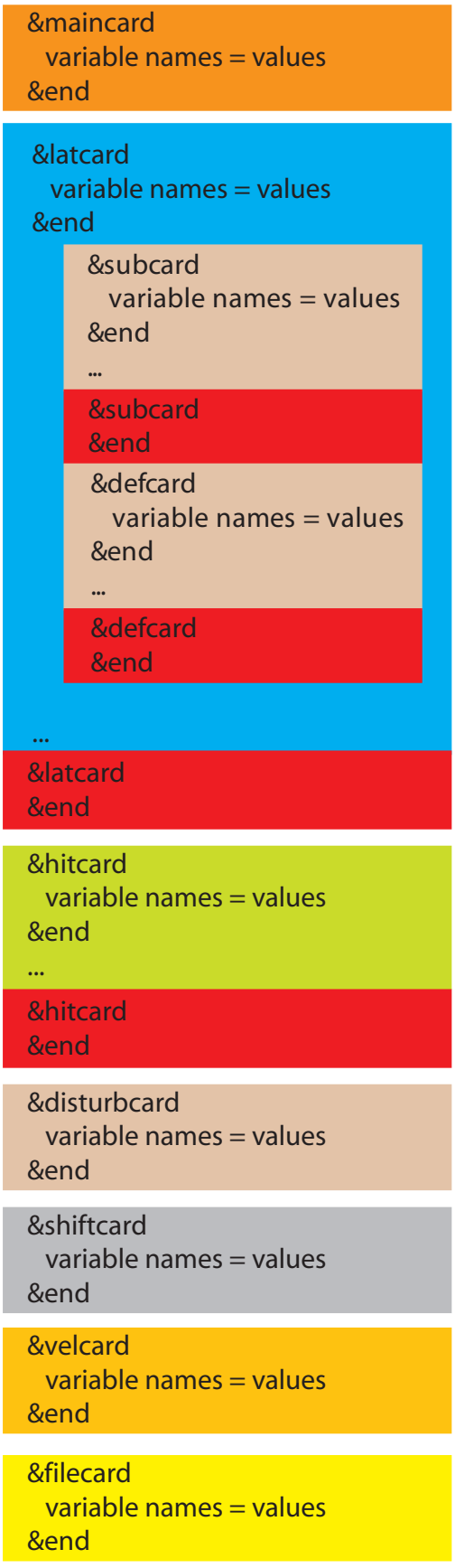

FIG. 1: structure of createAtoms input ¯le.

assume that the nanowire axis, the [0001] crystallographic direction, is along x-coordinate of our system. The $[\bar{1}100]$ and $[11\bar{2}0]$ crystallographic directions are assumed to align with y- and z- axes respectively. The stacking planes along x-, y-, and z- directions are then respectively (0001), $(\bar{1}100)$, and $(\bar{1}\bar{1}20)$. Our simulation would employ a Tersoff type of interatomic potential: J. Nord, K. Albe, P. Erhart, and K. Nordlund, J. Phys.: Condens. Matter, 15, 5649(2003). According to this potential, the hexagonal wurtzite lattice constants are a = 3.1798592 Å and c = 5.1926883 Å. This translates to lattice spacing of 5.1926883, 5.5076777 and 3.1798592 Å respectively for the (0001), $(\bar{1}100)$, and $(\bar{1}\bar{1}20)$ planes. If periodic boundary conditions are used, the length of the computational cell, perub - perlb, must be an integer multiplication of the plane spacing. Here we choose perub - perlb = 174×5.1926883, 10×5.5076777, $17 \times 3.1798592 = 903.52776$, 55.076777, 54.057606 Å. We arbitrarily choose a small offset (from zero) for perlb = -0.5, -0.5, -0.5 Å so that the cell boundaries do not coincide with atom positions. We can therefore get perub = 903.02776, 54.576777, 53.557606 Å. The choice of crystallographic orientations in the coordinate system and lattice constant of the crystal will be defined in "latcard" in the following. Returning to the case above, the random number seed ilatseed = 21 is arbitrarily chosen, amass = 69.72, 14.00674, 1.0, 1.0, 1.0 and ielement = 31, 7, 1, 1, 1 define mass and atomic number for the first element Ga and the second element N. In createAtoms, the maximum number of mass and ielement values that can be input is assumed to be five. Since ntypes = 2, only the first two amass and ielement values are used and others are ignored.

V. LATCARD

"latcard" contains eleven variables: "lattype", "alat", "xrot", "yrot", "zrot", "periodicity", "xbound", "ybound", "zbound", "strain", and delx". Here, "lattype" is a character string representing the lattice type and can be assigned to one of the three intrinsic lattices in createAtoms: sc (simple-cell), bcc (body-centered-cell), and fcc (face-centered-cell). Note that the sc, bcc, and fcc used here differ slightly from the commonly used terms simple-cubic, body-centered-cubic, and face-centered-cubic. This is because while we assume orthogonal crystal cell, we do not require lattice constants a = b = c, Fig. 2. The lattice constants a, b, and c are assigned to the vector variable alat. This means that the 'sc', 'bcc', and 'fcc' used in createAtoms are identical to simple-cubic, body-centered-cubic, and face-centered-cubic

only when the three values of alat are set to equal numbers. In createAtoms, users have the option to rotate the crystals so that the coordinate system does not necessarily coincide with the unit crystal cell axes. The three vectors xrot, yrot, and zrot, are then used to represent the Miller indices of the crystallographic planes (of the rotated crystal) that are normal to the coordinate axes x, y, and z. The vector variable periodicity is used to account for the periodic stacking of crystallographic planes in the three coordinate axes. For instance, the stacking sequence of (111) planes of a face-centered-cubic crystal is ABCABC..., and therefore periodicity in the (111) stacking direction should be assigned 3.0. The variables xbound, ybound, and zbound each holds two values representing the lower and higher bounds in the three coordinate directions within which atoms are filled according to the prescribed crystal structure. When the box defined by the xbound, ybound, and zbound are smaller than that defined by the perlb and perub bounds, free surfaces are created unless the gap is filled by other "latcard"s. Default xbound, ybound, and zbound values are to fill the entire computational cell defined by perlb and perub. The variable strain is a vector defining the three normal strains in the three directions so that when a lattice of one size is joined with a lattice of another size, a misfit strain can be pre-applied. Finally, delx is a vector defining a shift of the lattice so that one lattice can be spatially adjusted with respect to another. When not set, strain and delx are both default to zeros.

To continue with the creation of the hexagonal wurtzite GaN nanowire, we would use the following "latcard":

$\&latcard$

$lattype = `sc'$

$alat = 5.1926883, 5.5076777, 3.1798592$

$xrot = 1.0, 0.0, 0.0$

$yrot = 0.0, 1.0, 0.0$

$zrot = 0.0, 0.0, 1.0$

$periodicity = 1.0, 1.0, 1.0$

$strain = 0.0, 0.0, 0.0$

$delx = 0.0, 0.0, 0.0$

$\&end$

As described above, createAtoms can be used to directly create simple-cubic, body-centered-cubic, and face-centered-cubic structures as well as their orthogonal variants. The

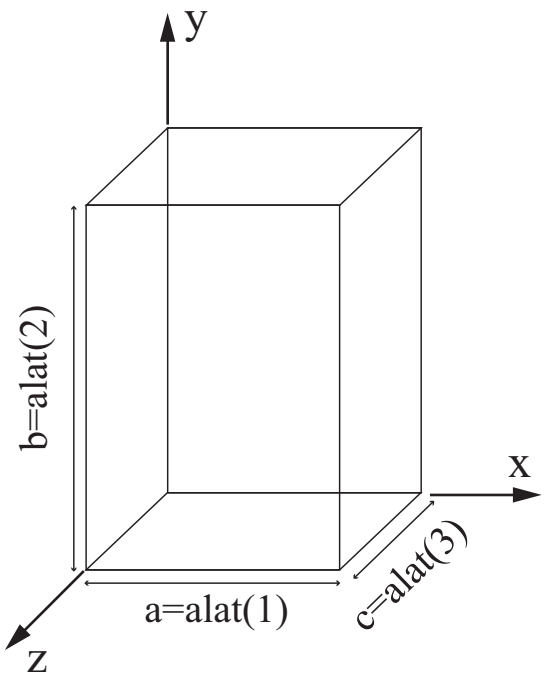

FIG. 2: unit crystal cell.

wurtzite structure is hexagonal and hence cannot be created directly. Because we use orthogonal crystallographic directions as our coordinate axes, each atom is periodically translational in the three coordinate axes. We can therefore use lattype = 'sc' coupled with multiple "subcard"s to create a hexagonal structure.

Fig. 3(a) shows the top view of atom arrangements of a hexagonal-closely-packed (hcp) crystal structure, where the blue circles are the atoms in the bottom plane of the crystal cell and the red circles are the atoms in the plane immediately above. From Fig. 3(a), we can define an orthogonal simple-cell (sc) as shown by the dash lines. This is magnified in Fig. 3(b). According to the Tersoff potential that we use, the lattice constants a = 3.1798592 Å and c = 5.1926883 Å. From the geometry and coordinate axes chosen in Fig. 3, we can identified the lattice constants for the orthogonal cell as alat = 5.1926883, 5.5076777, 3.1798592. Note that when using un-rotated crystal cell orientations, xrot = 1.0, 0.0, 0.0, yrot = 0.0, 1.0, 0.0, zrot = 0.0, 0.0, 1.0, and periodicity = 1.0, 1.0, 1.0. Finally, we do not need strain and shift so both strain and delx are set to zero.

In the maincard, we have indicated that the perub - perlb needs to match the lattice size in order to use periodic boundary conditions. When using unit crystal cell orientations without straining, the lattice-size-matching condition requires that perub - perlb = integer

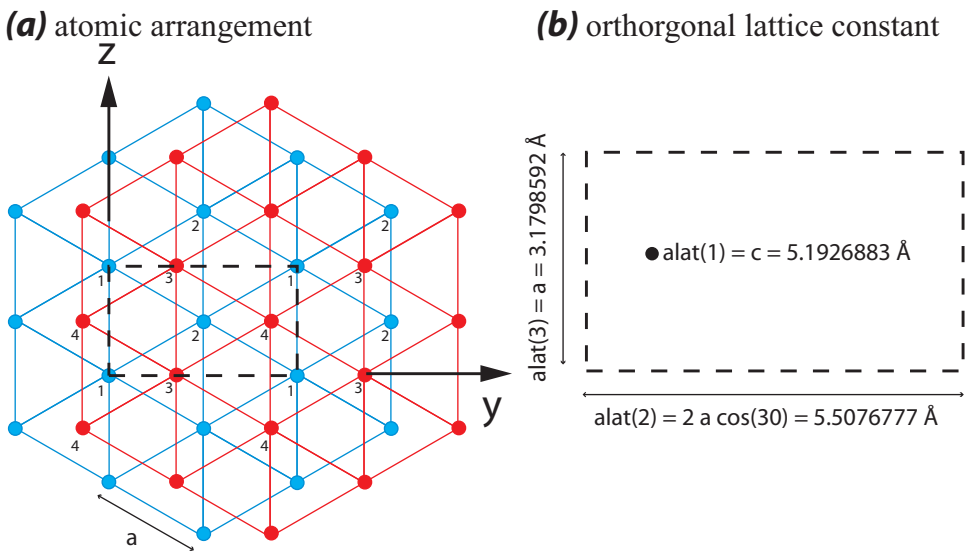

FIG. 3: top view of an hcp structure.

· alat. If the xrot, yrot and zrot are not in the unit crystal cell orientations, or when strain is not zero, then the general lattice-size-matching condition is expressed as perub - perlb = integer · $d_{(hkl)}$ · periodicity · (1 + strain), where $d_{(hkl)}$ represents the spacing between the planes normal to the corresponding coordinate axis. Defining the lattice constants as a = alat(1), b = alat(2), and c = alat(3), and the Miller index of the corresponding plane as (hkl) (i.e., the three components of xrot, yrot, or zrot are notated as h, k, and l), then the plane spacing can be calculated as

$$d_{(hkl)} = \frac{1}{\sqrt{(h/a)^2 + (k/b)^2 + (l/c)^2}} \tag{1}$$

Note that because multiple "latcard"s with different lattice sizes can be used, the periodic boundary lattice-size-matching condition can only be applied between the "maincard" and one of the "latcard"s.

VI. SUBCARD

The task a "subcard" performs is to fill atoms at sites of the lattice defined by the preceding "latcard". Separate "subcard"s are useful in several occasions: (a) some lattices can be decomposed into multiple sublattices (e.g., both fcc and hcp lattices can be decomposed into four sc sublattices); (b) complicated crystals may be composed of multiple sublattices (e.g., a diamond-cubic structure is composed of two interlacing face-centered-cubic sublattices);

and (c) compound structures may have different elements occupying different sublattices. "subcard" contains two variables: rcell and ccell. Here rcell is a vector defining the position of the sublattice origin, and ccell is a list of compositions of atoms occupying the sublattice (in the same atom sequence as in amass or ielement lists). Wurtzite GaN structure is composed of two interlacing hcp sublattices occupied by Ga and N atoms respectively. Examination of Fig. 3 indicates that an hcp lattice can be represented by four sc sublattices marked by "1", "2", "3", and "4" respectively. Corresponding to these four sublattices, we can use the following four "subcard"s to create a Ga hcp lattice:

$\&subcard$

$rcell = 0.0, 0.0, 0.0$

$ccell = 1.0, 0.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.0, 0.5, 0.5$

$ccell = 1.0, 0.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.5, 0.33333333, 0.0$

$ccell = 1.0, 0.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.5, 0.83333333, 0.5$

$ccell = 1.0, 0.0, 0.0, 0.0, 0.0$

$\&end$

Here rcell defines the origin of the four sublattices shown in Fig. 3 in unit of the orthogonal lattice constant, and ccell = 1.0, 0.0, 0.0, 0.0, 0.0 refers to pure Ga. The N sublattice is also an hcp structure but is shifted with respect to the Ga sublattice in the x-direction by a 0.375 lattice constant so that Ga and N atoms form tetrahedral bonds with respect to each other (i.e., wurtzite is a hexagonal version of the diamond structure). As a result, the following four "subcard"s can be used to define the N sublattice (with an additional empty subcard to terminate the list):

$\&subcard$

$rcell = 0.375, 0.0, 0.0$

$ccell = 0.0, 1.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.375, 0.5, 0.5$

$ccell = 0.0, 1.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.8755, 0.33333333, 0.0$

$ccell = 0.0, 1.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$rcell = 0.875, 0.83333333, 0.5$

$ccell = 0.0, 1.0, 0.0, 0.0, 0.0$

$\&end$

$\&subcard$

$\&end$

Here ccell = 0.0, 1.0, 0.0, 0.0, 0.0 refers to pure N.

VII. DEFCARD

"defcard" contains variables xmin, xmax, ymin, ymax, zmin, zmax, oldtype, newtype, prob, cent, plane, dist. Current implementation of createAtoms allows the change of atom type, thereby enable the creation of solute atoms, vacancies, and surfaces. Variables oldtype, newtype, and prob altogether mean that any atoms with atom type oldtype will be changed to atom type newtype at a probability of prob. When newtype $\leq$ 0, the atom will be changed to vacancy. In our GaN lattice, where Ga is represented by atom type 1 and N is represented by atom type 2, oldtype = 1, newtype = 0, and prob = 0.1 would create a GaN structure with about 10% Ga vacancies. When oldtype = 0 or not defined, the conversion will be made to all atoms. For instance, oldtype = 0, newtype = 0, and prob = 0.1 create a structure with about 10% vacancies in both Ga and N sublattices. There are two options to define regions within which defects are to be created: box option and plane option. Box

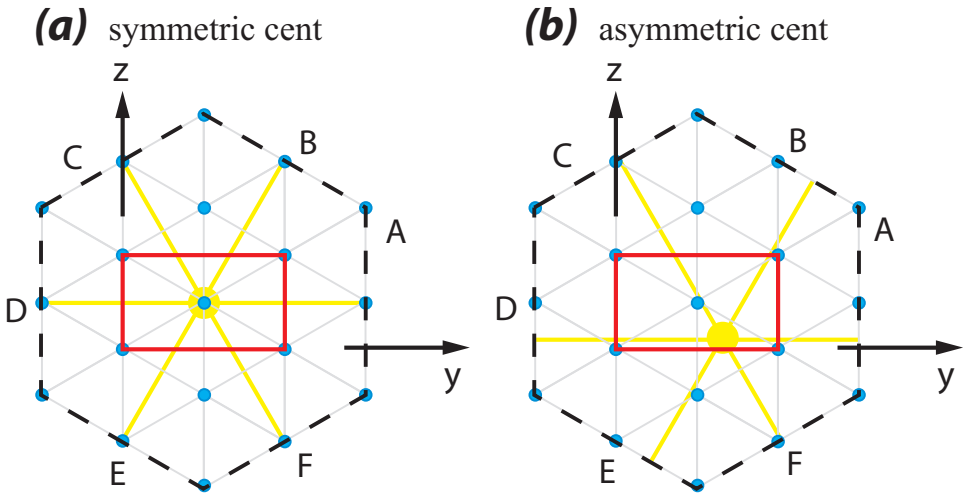

FIG. 4: Miller index of hexagonal facets.

option allows defects to be created within a specified box region whose boundaries in the three coordinate directions can be specified by the variables xmin, xmax, ymin, ymax, zmin, zmax. Plane option allows defects to be created beyond a specified plane. The latter case is specified by the variables cent, plane, and dist, where cent is a vector defining the position of a central point, plane is a vector defining the Miller index of a crystallographic plane, and dist is a distance value. Measured from the "cent" point, if the projected distance of any atoms along the plane normal direction exceeds the distance "dist", the defect creation will be applied. In this case, prob is always equal to 1.0. Note that only one option is used in each defcard. If both options are defined, then the box option overwrites the plane option when dist < 0 and the plane option overwrites the box option when dist ≥ 0.

Fig. 4 shows the cross section of a hexagonal structure whose facets are shown using dash lines. The three-index representation of the six facet planes can be written as A: (010), B: (011), C: $(0\bar{1}1)$, D: $(0\bar{1}0)$, E: $(0\bar{1}\bar{1})$, and F: $(01\bar{1})$.

With plane indexes determined above, we can use the following six "defcard"s to create a hexagonal nanowire (with an additional empty defcard to terminate the list):

$\&defcard$

$plane = 0.0, 1.0, 0.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

$plane = 0.0, 1.0, 1.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

$plane = 0.0, -1.0, 1.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

$plane = 0.0, -1.0, 0.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

$plane = 0.0, -1.0, -1.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

$plane = 0.0, 1.0, -1.0$

$cent = 0.0, 28.456, 27.029$

$dist = 17.0$

$newtype = -1$

$\&end$

$\&defcard$

&end

Here cent was chosen to be a central symmetric point so that an equal distance value of "dist" in all "defcard"s will give a symmetric hexagonal cross section, as shown by the yellow lines in Fig. 4(a). Note that the x-component of cent does not affect the result as all planes are parallel to x-axis. In principle, cent can be chosen to be anywhere inside the cross section. However, if cent is not at an asymmetric point, then different dist values have to be used in different "defcard"s in order to create a symmetric cross section, as shown by the yellow lines in Fig. 4(b). The variable dist is arbitrarily set to be dist = 17.0, which affects the cross section size of the wire. Note that we have used newtype = -1 rather than newtype = 0 to indicate vacancy. Both newtype = -1 and newtype = 0 cause the removal of atoms. However, newtype = -1 performs an additional function. In createAtoms, it is assumed that the cross section area of a bulk system is always determined by cell sizes perub - perlb in the cross section dimensions. For nanowire application, however, the cross section area is reduced due to the cut off of the planes. The reduction of the area can be quantified by the number of atoms removed compared with the total number of atoms prior to any removal of atoms. Setting newtype = -1 rather than 0 will activate the counting of the number of atom removed for the calculation of the cross section area. The calculated cross section area will be written as part of the header of the crystal file.

Finally, if there is no additional "latcard-subcard-defcard" combos, an empty "latcard" is inserted here to signal the end of the "latcard" list.

VIII. HITCARD

IX. DISTURBCARD

"disturbcard" contains two variables dismax and strain. After the entire system is created, createAtoms provides an additional capability to randomly disturb the position of each atom and apply normal strain to the system. "dismax" is a value relating to the maximum displacement atoms can have so that the actual displacement of each atom in each of the coordinate directions is randomly chosen between -0.5·dismax and 0.5·dismax. The vector strain defines the three normal strains. Default values of dismax and strain are both zero. Here for GaN nanowire, we use an empty "disturbcard":

&*disturbcard*

&*end*

X. SHIFTCARD

To facilitates analysis, createAtoms implements a "shiftcard" to provide a capability to shift the computational cell position. "shiftcard" contains one integer variable: mode. Default value of mode is mode = 0, which means no shift. When mode = 1 and hit is not defined, the center of the perlb(2) and perub(2) will be aligned at y = 0. When mode = 1 and hit is defined, then the interface across two designated hit values will be aligned at y = 0. When mode = 2, the cell boundaries perlb(1) and perub(1) will be reset at the middle point between two widely-separated crystallographic planes. To illustrate this, we show in Fig. 5 the atomic configuration of wurtzite GaN (0001) planes stacked along the x-direction. It can be seen that the planes are stacked with Ga/N-Ga/N ... sequence in the x-direction, where / means a close separation as indicated by the arrows "A", whereas - means a wide separation, as indicated by the arrows "B". Using mode = 2 would then creates cell boundaries indicated by the brown region rather than the gray region. This ensures neat, universal configurations for thermal transport simulations where the hot and cold boundaries will also be inserted at similar middle points between two widely-separated crystallographic planes (see, isoT.f).

According to the description, we can use the following "shiftcard" for our GaN nanowire:

&*shiftcard*

mode = 2

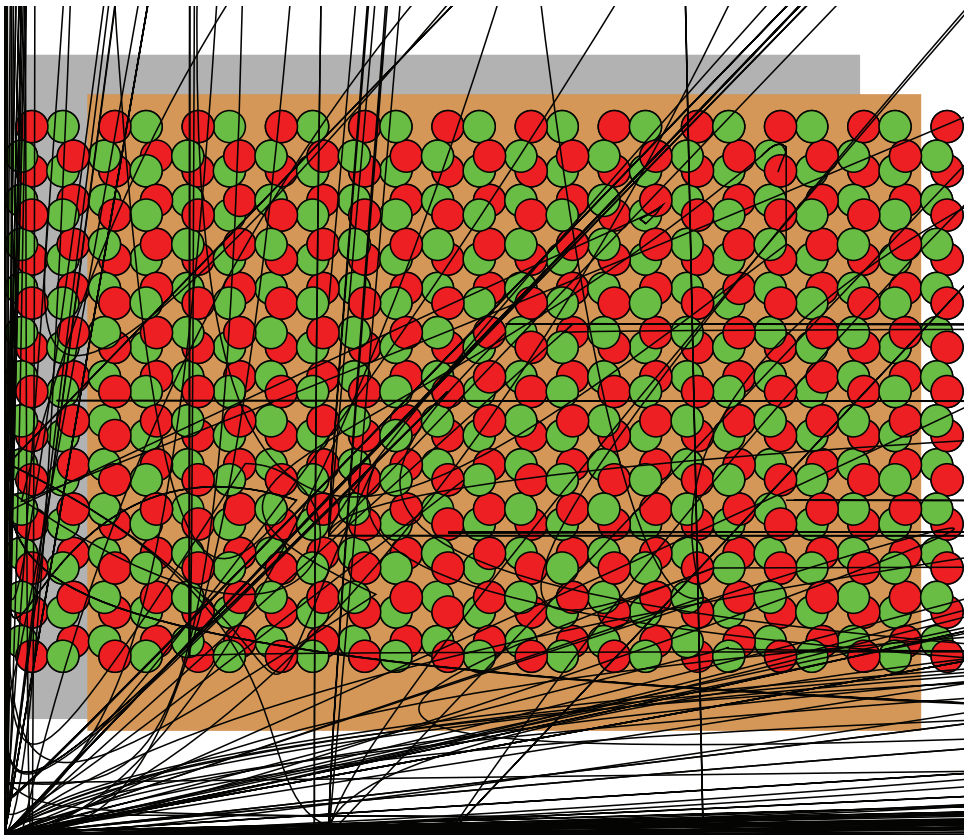

FIG. 5: Wurtzite GaN (0001) planes in x-direction.

&end

XI. VELCARD

"velcard" can be used to initialize temperature for the system by assigning velocities to atoms. It contains six variables: Tbnd, Tmid, naxis, nmesh, equal, and ensureTave. Definition of the four variables, Tbnd, Tmid, naxis, and nmesh, is shown in Fig. 6. The system is divided into nmesh mesh grids in the naxis-th dimension (naxis = 1, 2, 3 corresponding respectively to the x-, y-, and z- axes). The atom velocities are assigned according to Boltzmann distribution and a linear temperature distribution where the temperature at the lower and higher boundaries of the computational cell equals Tbnd and the temperature in the middle of the cell equals Tmid. Due to equation partition between kinetic energy and potential energy, the temperature initialized by assigning velocities alone will drop approximately by half when the kinetic energy is equilibrated with the potential energy during molecular

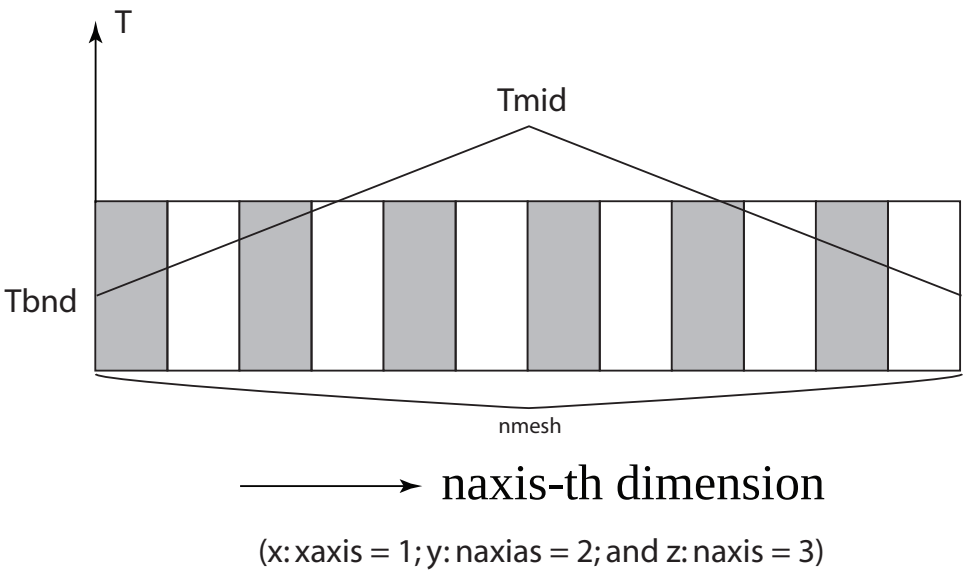

FIG. 6: temperature pro¯le for initializing atom velocities.

dynamics simulations. The multiplication factor equal can be used so that Tbnd becomes equal·Tbnd, and Tmid becomes equal·Tmid. A value of equal = 2.0 will therefore give a correct temperature distribution after equilibrium between kinetic and potential energies is reached. Finally, the initialized average temperature of the entire system is close to Tave = (Tbnd+Tmid)/2. However, they may not be exactly equal. Setting ensureTave = 1.0 will cause the atom velocities to be scaled so that the average system temperature is exactly Tave. If ensureTave is set to 0.0, such as in the default case, there is no velocity rescaling for Tave.

To generate a constant temperature T, one can use Tbnd = T, Tmid = T, naxis = 1, and nmesh = 1. Default is not to initialize temperature. In our case of the GaN nanowire, we do not initialize temperature so we can use an empty "velcard":

&velcard

&end

XII. FILECARD

"filecard" defines the file name for outputting the created configuration. It contains four character string variables: dynamo, Paradyn, lammps, and xyz, respectively for file format readable to dynamo, Paradyn, lammps, and jmol (or vmd). Default values for these variables

are 'none'. If assigned with 'none', then the corresponding file will not be created. We would use the following "filecard" to create a Paradyn file and a lammps file:

$\&filecard$

$dynamo = `none'$

$paradyn = `atoms.para'$

$lammps = `GaN\_NanoW5.lmp'$

$xyz = `none'$

$\&end$

XIII. RESULT

By running createAtoms with the input file described above, we can create a hexagonal wurtzite GaN nanowire. The cross section of the created nanowire is shown in Fig. 7.

---

* xzhou@sandia.gov

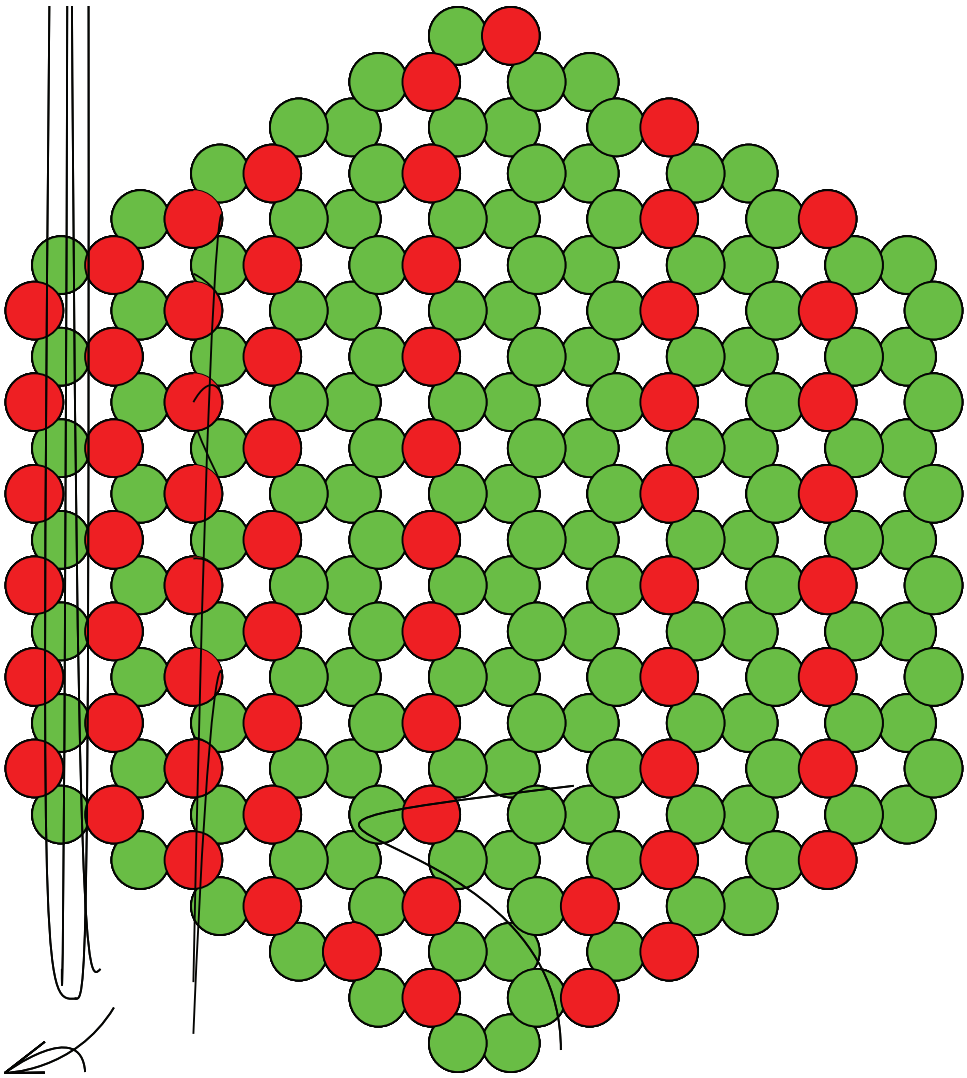

FIG. 7: cross section of the created hexagonal GaN nanowire.