UPPSALA
UNIVERSITET

Uppsala University
Department of Physics & Astronomy
Division of Materials Theory

# Quantum confinement effects in materials for solar cell applications

Dimitrios Koulentianos
Supervision: Assistant Professor Jan Rusz

# Quantum confinement effects in materials for solar cell applications

Dimitrios Koulentianos
Supervision: Assistant Professor Jan Rusz

## Contents



# 1 Introduction

Due to the continuous demand nowadays for renewable energy sources, solar energy has become a focal point. As a result a lot of effort has been made in order to construct solar cells with higher efficiencies and lower cost. The main purpose in order to accomplish this task is the design of new materials that fulfill specific requirements, which are necessary in order to get the desired results.

Our attention towards $CuInSe_2$ was turned after the potential applications on solar cell technology that can result from the study and understanding of it. $CuInSe_2$ is now considered to be one of the most promising materials for the construction of solar cells. In a paper published in 2012 by Wang *et al.* [1], it is mentioned that $CuInSe_2$ has chalcopyrite structure at room temperature and is a semiconductor characterized by direct band gap. Moreover, it has a high rate of absorbing sunlight and it is a low cost material.

In addition, a review article published in *"Nature materials"* by Curtarolo *et al.* in 2013 [2] describes a new method in order to synthesize new materials for new desired applications and identify the existing ones in the current applications, called *high-throughput* (HT). According to [2] a desired material for solar cell applications should be a semiconductor having strong optical absorption coefficient, low cost and the range of the band gap be at $\approx 1.3$ eV. In the `http://pveducation.org/pvcdrom/materials/CuInSe2`, all the parameters and the properties of $CuInSe_2$ are mentioned. One can then see that for $CuInSe_2$ the energy band gap is $1.02eV$ but it is also known that *quantum confinement effect* allows the manipulation of the band gap, so we can expand the band gap of $CuInSe_2$. There is also the demand for an energy

gap size around the the peak of sun-light intensity. So, we want to explore quantum confinement effects in $CuInSe_2$ nanoparticles, hoping to optimize the gap for a suitable size of nanoparticles. For the above reasons $CuInSe_2$ was chosen for the purposes of this project.

Our purpose in this project was to study the dependance between the energy gap and the diameter for $CuInSe_2$ nanoparticles (quantum dots). The theoretical background of the project is based on the quantum confinement effect and the *Density Functional Theory (DFT)*, whilst for the computational part a **FORTRAN 90** code was used for the construction of the nanoparticles, the *Jmol* software for the visualization of them and the main part of the calculations was made by the the **SIESTA** software.

The structure of this report is the following, first we start by introducing the quantum confinement effect, some fundamental aspects of the DFT theory (Kohn-Sham equation,etc.), the idea of a *pseudopotential* and by explaining *basis set*. Then we describe the software used for our task, we mention and explain the code used for the construction of nanoparticles and the way calculations in SIESTA software take place. Finally we conclude by mentioning the results we got for the $CuInSe_2$ nanoparticles.

# 2 Theory

As mentioned above, this section consists of four parts:

- The explanation of the quantum confinement effect
- Some fundamental aspects of DFT
- The pseudopotential idea
- The basis set

## 2.1 Quantum confinement effect

In general the quantum confinement can be observed when the dimensions of a material are of the same magnitude as the wavelengths of the electrons in the sample. When quantum confinement occurs a change in the electronic structure of the material is observed and the electronic and optical properties are different of those that characterize the bulk material.

When the dimensions are a lot greater than the electron's wavelength in the sample the electron can be considered as free and it is known from quantum mechanics that it will have a continuous spectrum. The decreasing of the dimensions will cause the electron's spectrum to become discrete and also an increase the material's band gap will be observed. A rough way to justify the previous statement is the equation for the energy states of the three dimensional particle in a box model:

$$E_{n_x,n_y,n_z} = \frac{\hbar^2\pi^2}{2m}\left[\left(\frac{n_x}{L_x}\right)^2 + \left(\frac{n_y}{L_y}\right)^2 + \left(\frac{n_z}{L_z}\right)^2\right] \tag{1}$$

From equation (1) it can be seen that a decrease in dimensions corresponds to an increase in the energy. Figure 1 shows the transition from the bulk material to a quantum dot. In the same figure the blue shift in the optical properties due to the increase of the energy can be observed.

Before moving forward we will try to describe the quantum confinement effect in a semiconductor. In a semiconductor it is known that with the increase of temperature an electron can acquire higher energy than the material's band gap and jump from the valence to the conduction band leaving a hole in the valence band. This electron-hole pair is called an exciton and it can be simulated by a hydrogen atom. A way to test if we are in the quantum confinement regime is to compare the radius of the material $R$ with the exciton's Bohr radius $\alpha_B^*$. We define the weak and the strong confinement regime as follows:

- When $R/\alpha_B^* \approx 1$, then we are in the weak confinement regime
- When $R/\alpha_B^* \ll 1$, then we are in the strong confinement regime

The exciton's Bohr radius is:

$$\alpha_B^* = \epsilon_r\left(\frac{m_e}{m^*}\right)\alpha_B, \tag{2}$$

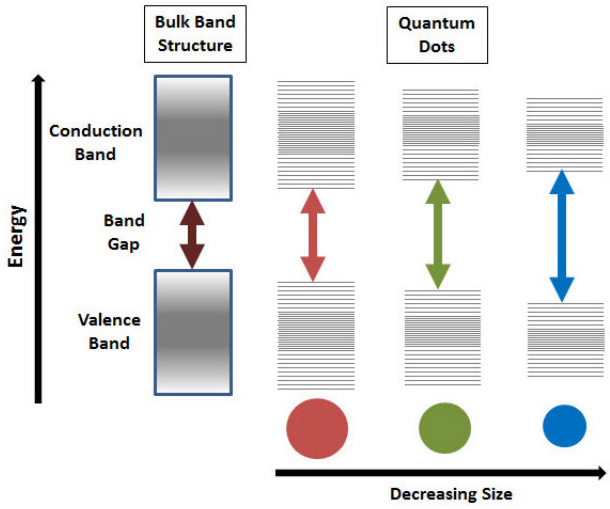



Figure 1: The decrease of dimensions results in the increase of the band gap. The energy difference results also the blue shift in optical properties. Figure taken from `http://www.sigmaaldrich.com`.

where $\alpha_B \approx 0.53$Å is the Bohr radius, $\epsilon_r$ is the dielectric constant, $m_e$ is the mass of the electron and $m^* = (m_e^* \cdot m_h^*)/(m_e^* + m_h^*)$ is the effective reduced mass of the electron-hole system, where $m_e^*, m_h^*$ are the effective masses of the electron and the hole respectively. Finally the confinement energy for the exciton is given by the formula:

$$E_{con} = \frac{\hbar^2 \pi^2}{2\alpha^2}\left(\frac{1}{m_e} + \frac{1}{m_h}\right), \tag{3}$$

where $\alpha$ is the exciton's radius and $m_e$ and $m_h$ the masses of the electron and the hole respectively.

## 2.2 Density Functional Theory

It is known from quantum mechanics that the wavefunction $\psi$ of a system contains all the information about the system and satisfies the Schrödinger equation:

$$\hat{H}\psi = \epsilon\psi, \tag{4}$$

where $\hat{H}$ is the Hamiltonian operator and $\epsilon$ the energy of the system. Exact solution for equation (4) can be found for the particle in a box, the harmonic oscillator, the hydrogen atom and maybe other physical systems, but in general getting the exact solution of equation (4) is very difficult. In addition, in the three previous systems we have one particle for the first and the second case and two for the third (the electron and the proton). In the case of an atom with many electrons or a molecule, equation (4) can't be solved exactly and certains approximations should be applied.

Suppose that we want to solve equation (4) for a system of $N$ electrons and $N'$ nuclei. The wavefunction will have the form[1] $\Psi = \Psi(\vec{r_1}, ..., \vec{r_N}, \vec{R_1}, ..., \vec{R_N})$ and the Schrödinger equation will be now:

$$\hat{H}\Psi = E\Psi, \tag{5}$$

where the Hamiltonian, under the Born-Oppenheimer approximation, will have the form:

$$\hat{H} = -\frac{\hbar^2}{2m}\sum_{i=1}^{N}\nabla_i^2 - \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{j=1}^{N'}\frac{Z_j e^2}{|\vec{R}_j - \vec{r}_i|} + \frac{1}{8\pi\epsilon_0}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N}\frac{e^2}{|\vec{r}_i - \vec{r}_j|}, \tag{6}$$

where the first term is the kinetic energy of the electrons, the second the electrostatic energy between electrons and nuclei and the third the electrostatic repulsion between electrons. In order for someone to calculate the total energy of the system the repulsion between nuclei should also be taken under

[1] We switch to upper case $\Psi$ to remark ourselves that we have a many electron system.

consideration. Exact solution of equation (5) with the Hamiltonian (6) is not possible and the first approximation method for that purpose was the *Hartree-Fock* approximation. Since the purpose of this project is not a complete description of DFT we will not mention the details of this method and we refer the reader to the relevant bibliography [3],[4].

The *Thomas-Fermi* model [4], which was stated in 1927, can be considered as a first step towards DFT. It has the great advantage that it avoids painful calculations for the solution of equation (5), as it was the case with the Hartree-Fock model. Before we describe the Thomas-Fermi model, we have to mention first a very important principle of Physics, called the *variational principle*.

We know that the expectation value of energy for a system being in the state $\Psi$ is given by:

$$E[\Psi] = \frac{\int \Psi^* \hat{H} \Psi \mathrm{d}\vec{r}}{\int \Psi^* \Psi \mathrm{d}\vec{r}}, \tag{7}$$

the variational principle states now that: *any other state $\Psi$ which is not the one that corresponds to the ground state energy will result an energy which is an upper bound to the ground state state energy $E_0$.* It can be shown that minimization of the *functional* $E[\Psi]$, with respect to the wavefunctions of all electrons, in equation (7) will give the $\Psi_0$ and $E_0$, the ground state and the ground state energy respectively. In the case now of the Thomas-Fermi model we will apply the variational principle, but the variable with respect to which different quantities have to be minimized will not be the wavefunction $\Psi$ but the *electron density* $\rho(\vec{r})$.

Though the wavefunction contains all the information for the system, the calculation of it can be very difficult. Thomas and Fermi, proposed a model based on the uniform electron gas, where the kinetic energy and the total energy of an electron are both functionals of the electron density and are given by:

$$\begin{aligned} T_{TF}[\rho(\vec{r})] &= \frac{3}{10}(3\pi^2)^{2/3} \int \rho^{5/3}(\vec{r}) \mathrm{d}\vec{r} \\ E_{TF}[\rho(\vec{r})] &= T_{TF}[\rho(\vec{r})] + \frac{1}{2} \iint \frac{\rho(\vec{r}_1)\rho(\vec{r}_2)}{r_{12}} \mathrm{d}\vec{r}_1 \mathrm{d}\vec{r}_2. \end{aligned} \tag{8}$$

Now the correct density which should be applied to the second of equations (8), can be found by the variational principle mentioned before. According to that model the density that gives the ground state energy can be found, fulfilling the condition:

$$\int \rho(\vec{r}) \mathrm{d}\vec{r} = N, \tag{9}$$

where N is the total number of electrons. That make sense because if we consider a non excited state of the system then by integrating with respect to the electron density we must have the total number of electrons, so in that way we have a good way to get the most proper density for equations (8).

Until that point we see that $\rho(\vec{r})$ plays a very important role in our effort to find a way to solve equation (5). This argument is also supported by two theorems which play a fundamental role in DFT and were stated in 1964 by P.Hohenberg and W.Kohn [4], [5]. In the following we will mention the two theorems and we will see their consequences. The proof of each theorem which is not given here can be found in the relevant paper [5].

The first Hohenberg-Kohn theorem states that: *for a system of interacting particles in an external potential $V_{ext}$, both the external potential and as a result the Hamiltonian are unique functionals of $\rho(\vec{r})$.* The energy of the system can be written as a sum of three terms, where each term will be a functional of $\rho(\vec{r})$ and the same will be for the total energy. We get:

$$E[\rho(\vec{r})] = T[\rho(\vec{r})] + V[\rho(\vec{r})] + U[\rho(\vec{r})], \tag{10}$$

where the first term in the right hand side of equation (10) is the kinetic energy, the second is the external potential and the third describes the electrostatic repulsions between the electrons. Now equation (10) can be written as:

$$E[\rho(\vec{r})] = \int \rho(\vec{r}) V_{ext} \mathrm{d}\vec{r} + F_{HK}[\rho(\vec{r})], \tag{11}$$

with $F_{HK}[\rho(\vec{r})]$ given by:

$$F_{HK}[\rho(\vec{r})] = T[\rho(\vec{r})] + U[\rho(\vec{r})]. \tag{12}$$

$F_{HK[\rho(\vec{r})]}$ is probably the most important quantity in DFT and if it could be exactly determined then equation (5) could be solved exactly as well. The advantage of introducing this method is its applicability to all the systems, from atoms to molecules and solids. Unfortunately we can not know the exact form of $F_{HK}[\rho(\vec{r})]$ but the second term in the right hand side of equation (12) can be defined. It will be the sum of the classical Coulomb repulsion and of non-classical terms, we get:

$$U[\rho(\vec{r})] = \frac{1}{2}\iint \frac{\rho(\vec{r}_1)\rho(\vec{r}_2)}{r_{12}}\mathrm{d}\vec{r}_1\mathrm{d}\vec{r}_2 + E_{ncl} = J[\rho(\vec{r})] + E_{ncl}, \tag{13}$$

the non-classical terms are the self interaction correction and the exchange and Coulomb correlation.

Up to that point we see that the only parameter in all the expressions for energy is the electron density $\rho(\vec{r})$ and if the ground state density is inserted into the energy expressions acquired above then we get the ground state energy. This is the second Hohenberg-Kohn theorem which states that *the functional $F_{HK}[\rho(\vec{r})]$ will give the ground state energy if and only if the density inserted in it is the ground state density.* This is actually an implementation of the variational principle and it can be written as:

$$E_0 \leq E[\rho(\vec{r})] = T[\rho(\vec{r})] + V[\rho(\vec{r})] + U[\rho(\vec{r})] \tag{14}$$

From equation (14) we see that any other density, except the ground state density, will give an energy which will be an upper bound to the ground state energy of the system.

So by using the DFT we avoid difficult calculations in order to get an approximated solution of equation (5), but still some functionals can not be fully determined. This is of course the $F_{HK}[\rho(\vec{r})]$ functional, which it will be determined if $T[\rho(\vec{r})]$ and $E_{ncl}$ could be determined. That can be seen by writing equation (12) in the form:

$$F_{HK}[\rho(\vec{r})] = T[\rho(\vec{r})] + J[\rho(\vec{r})] + E_{ncl}. \tag{15}$$

Now the Thomas-Fermi model is not the best approximation for the kinetic energy functional $T[\rho(\vec{r})]$, so W.Kohn and L.J.Sham in an article published in 1965 [4], [6] proposed a way to define the kinetic energy functional. We will describe briefly what they did and we refer the reader to relevant paper for a detailed description.

The main idea was to calculate the kinetic energy for a system of non-interacting particles which has the same density as the one of the interacting particles. In that case, the kinetic energy and the density will be given by:

$$T_{ni} = -\frac{\hbar^2}{2m}\sum_{i=1}^{N}\int \phi_i^*\nabla^2\phi_i\mathrm{d}\vec{r} \qquad \rho(\vec{r}) = \sum_{i=1}^{N}|\phi_i(\vec{r})|^2. \tag{16}$$

In addition, they wrote the $F_{HK}[\rho(\vec{r})]$ functional in the form:

$$F_{HK}[\rho(\vec{r})] = T_{ni} + J[\rho(\vec{r})] + E_{XC}[\rho(\vec{r})], \tag{17}$$

where the $E_{XC}[\rho(\vec{r})]$ term is called the *exchange-correlation energy* and can is defined as:

$$E_{XC}[\rho(\vec{r})] = (T[\rho(\vec{r})] - T_{ni}) + (U[\rho(\vec{r})] - J[\rho(\vec{r})]). \tag{18}$$

Now the total energy for the interacting particles can be written in terms of $T_{ni}$ and $E_{XC}[\rho(\vec{r})]$ as:

$$E[\rho(\vec{r})] = T_{ni} + J[\rho(\vec{r})] + E_{XC}[\rho(\vec{r})] + V[\rho(\vec{r})]. \tag{19}$$

In equation (19), $E_{XC}[\rho(\vec{r})]$ is the only term for which we don't have an analytical expression. If we replace the other terms by their analytical expressions given by equations (11), (13), (16) in equation (19) and apply the variational principle for the minimization of energy under the usual demand:

$$\int \phi_i^*(\vec{r})\phi_j(\vec{r})\mathrm{d}\vec{r} = \delta_{ij}, \tag{20}$$

then we obtain the *Kohn-Sham equation*:

$$\Big( -\frac{\hbar^2}{2m}\nabla^2 + V_S(\vec{r})\Big)\phi_i(\vec{r}) = \epsilon_i\phi_i(\vec{r}). \tag{21}$$

Let us know do some comments about equation (21). First we can say that it has the exact same form as Schrödinger equation, but now instead of the many-electron wavefunction $\Psi$ we have the single particle wave-function for our non-interacting particles system $\phi_i(\vec{r})$. The *Kohn-Sham potential* $V_S(\vec{r})$ is given by:

$$V_S(\vec{r}) = V_{ext}(\vec{r}) + V_{XC}(\vec{r}) + V_C(\vec{r}), \tag{22}$$

where $V_C(\vec{r})$ is the Coulomb potential and $V_{XC}(\vec{r})$ is the *exchange-correlation potential*, obtained as the variation of $E_{XC}[\rho(\vec{r})]$ with respect to $\rho(\vec{r})$. Once the Kohn-Sham equation has been solved, we can replace the $\phi_i(\vec{r})$ to the second of equations (16) and get the density of the non-interacting system which will be the same as the one of the interacting one. Finally we mention, without entering in more details, that the orbitals $\phi_i(\vec{r})$ have little or strictly speaking no physical meaning.

In order for the Kohn-Sham equation (21) to be solved, we need to obtain an expression for $E_{XC}[\rho(\vec{r})]$. Such an expression can be obtained only through approximation methods and here we will mention the two most common approximations, which are the *Local Density Approximation (LDA)*[4] and the *Generalized Gradient Approximation (GGA)* [4].

The LDA is based on the *uniform electron gas* model, where the electrons are moving in a space of positive charge and the system as a whole is neutral. The exchange-correlation energy is given by:

$$E_{XC}^{LDA}[\rho(\vec{r})] = \int \epsilon_{XC}(\rho(\vec{r}))\rho(\vec{r})\mathrm{d}\vec{r}, \tag{23}$$

where $\epsilon_{XC}(\rho(\vec{r}))$ is the exchange-correlation energy density. It can be considered as the sum of two terms:

$$\epsilon_{XC}(\rho(\vec{r})) = \epsilon_X(\rho(\vec{r})) + \epsilon_C(\rho(\vec{r})), \tag{24}$$

where $\epsilon_X(\rho(\vec{r}))$ denotes the exchange contributions and $\epsilon_C(\rho(\vec{r}))$ denotes the correlation contributions. Bloch and Dirac gave an expression for $\epsilon_X(\rho(\vec{r}))$ [4] which is:

$$\epsilon_X(\rho(\vec{r})) = -\frac{3}{4}\left(\frac{3\rho(\vec{r})}{\pi}\right)^{1/3}. \tag{25}$$

There is no expression for $\epsilon_C(\rho(\vec{r}))$ but numerical quantum Monte-Carlo simulation can be found [7].

The GGA method is taking into account not only the density $\rho(\vec{r})$ in a point $\vec{r}$ but also the gradient of the density $\vec{\nabla}\rho(\vec{r})$ so that non-homogeneity of distribution can be included. The exchange-correlation energy in GGA is given by:

$$E_{XC}^{GGA}[\rho(\vec{r})] = \int \epsilon_{XC}(\rho(\vec{r}), \vec{\nabla}\rho(\vec{r}))\rho(\vec{r})\mathrm{d}\vec{r} \tag{26}$$

The main reason the LDA and GGA methods mentioned is, as we will see later, that they need to be declared in the input file for Siesta DFT, the program used for the calculations that took place for this project.

## 2.3 The pseudopotential

We saw in the previous the Kohn-Sham potential given by equation (22), which is necessary in order to solve the Kohn-Sham equation (21). Since in the way the potential was defined made the calculations easier, a general idea was to apply in equation (5) a pseudopotential, so obtain a solution by avoiding difficult calculations, which of course describes the properties of a system.

In our case, since we deal with many interacting atoms where only the valence electrons determine the interactions, a good potential (pseudopotential) to replace the real complicated one, could be a potential where the the effects emerging from the core (non-valence) electrons and the nuclei have been removed and only the chemically active valence electrons have been taken under consideration.

We have to specify at this point, that the pseudopotential should have the same form and give the same wavefunction with the real potential outside a cutoff radius $r_c$. That can be seen in figure 2. Pseudopotentials characterized by large cutoff radii are called softer, i.e., converge rapidly but are also less transferable, i.e., do not reproduce realistic results under different conditions.

For the calculations took place with SIESTA, `http://departments.icmab.es/leem/siesta/`, we have used pseudopotentials. There are codes made in oder to generate pseudopotentials e.g. *Opium*, `http://opium.sourceforge.net/`, since we had the pseudopotentials pre-generated we won't enter into more details and mention here the mathematical forms of different pseudopotentials.

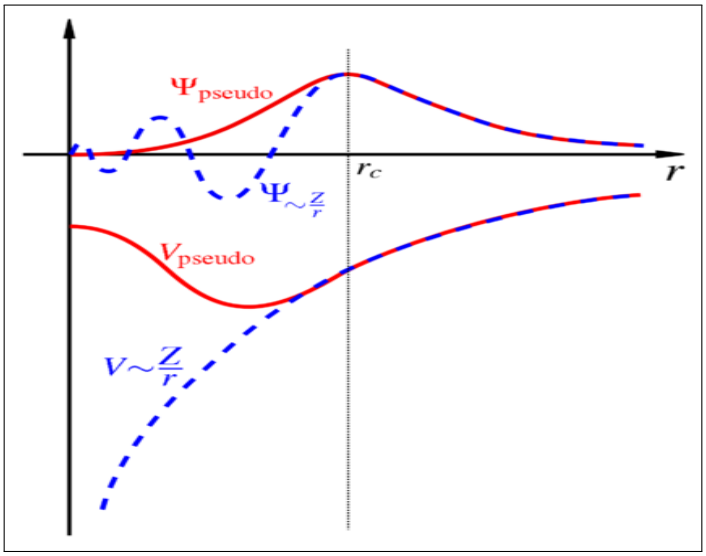



Figure 2: After $r_c$ both the real potential and the pseudopotential are giving the same wavefunction.

### 2.4 Basis set

It is known from the theory of *linear differential equations*, that the linear combination of the solutions of these equations is also a solution and that apply for both the ordinary and the partial linear differential equations. The Schrödinger equation (4) is in general a partial linear differential equation, so the solutions of (4) form a *vector space* and as it is also known from Linear Algebra, that each vector space has a *basis* i.e. a set of linearly independent vectors, a linear combination of which can give any vector of the space.

In the above paragraph, the term vector has not been used to describe physical quantities that can be represented by vectors like force, velocity etc. The functions that are solutions to (4) and called atomic orbitals, are now vectors and form a vector space. So as a basis set now we can have a set of atomic orbitals, which, when written as a linear combination, give molecular orbitals, or a set of plane wave functions as is often the case in materials physics.

Now in our case, for the calculations took place with SIESTA, due to the fact that the pseudopotential takes only the valence electrons into consideration, it is convenient to have a basis set where a basis function will correspond to each valence atomic orbital. A basis set like that is called a *split-valence basis set*. When multiple basis functions correspond to each valence orbital then we say that we have a double, triple or quadruple - zeta ($\zeta$). In a more formal description that has to do with the radial functions per angular momentum channel. So a single $\zeta$ means one radial function per angular momentum channel etc. We will come back to that when we will see the structure of an input file in SIESTA.

## 3 Computational Setup

In this section we mention and explain all the software used for the purposes of this project. In the following we will describe:

- The FORTRAN code made for the construction of the nanoparticles

- Their visualization with Jmol software
- The calculations with Siesta DFT

## 3.1 Construction of nanoparticle models

In order for someone to write a code for the construction of $CuInSe_2$ nanoparticles, the unit cell of the crystal has to be known. After the unit cell is known, one has to write down the coordinates of each atom in the unit cell and expand this structures for different radii of nanoparticles. The unit cell can be found at `http://gurka.fysik.uu.se/ESP/` and it contains sixteen atoms. It can also be seen that the lattice parameters of the unit cell are: $a = b = 5.8$Å and $c = 11.7$Å, whilst for the angles we have $\alpha = \beta = \gamma = 90°$. The positions of the atoms in the code have been expressed in fractional coordinates (thus, each of the $x, y, z$ coordinates is a fraction of $a, b, c$ respectively). According to all the previous, we have the following code for the construction of nanoparticles:

```
program make_nano_CuInSe122

implicit none

double precision radius, alat, clat
double precision basis (3,16)

integer h,k,l,hmax,kmax,lmax,ib,nb,nat
double precision r(3)

character (len=2) atbasis(16)

basis=reshape( (/0.,    0.,    0.,          &
                      1./2., 1./2.,    0.,  &
                      1./2.,    0., 1./4.,  &
                         0., 1./2., 1./4.,  &
                      1./4., 1./4., 1./8.,  &
                      3./4., 3./4., 1./8.,  &
                      3./4., 1./4., 3./8.,  &
                      1./4., 3./4., 3./8.,  &
                         0.,  0.,   1./2.,  &
                      1./2., 1./2., 1./2.,  &
                      1./2.,  0.,   3./4.,  &
                         0., 1./2., 3./4.,  &
                      1./4., 1./4., 5./8.,  &
                      3./4., 3./4., 5./8.,  &
                      3./4., 1./4., 7./8.,  &
                      1./4., 3./4., 7./8./), shape(basis))

atbasis= reshape( (/'Cu', 'In', 'In', 'Cu', 'Se', 'Se', 'Se', 'Se', 'In', 'Cu', 'Cu
                     'In', 'Se', 'Se', 'Se', 'Se'/), shape(atbasis))

alat=5.800   !Angstrom
clat=11.700  !Angstrom
radius= !desired length between 5-12 Angstrom
nb= 16
hmax = ceiling(radius/alat)+1
kmax = ceiling(radius/alat)+1
lmax = ceiling(radius/alat)+1

nat = 0
```

```
  do h=-hmax, hmax
    do k=-kmax, kmax
      do l=-lmax, lmax
        do ib = 1, nb
          r(1) = alat*(dble(h)+basis(1,ib))
          r(2) = alat*(dble(k)+basis(2,ib))
          r(3) = clat*(dble(l)+basis(3,ib))
          if(sqrt(dot_product(r,r))<radius) then
            write(30,'(a2,3f14.6)') atbasis(ib), r
            nat = nat + 1
 write(40,'(3f14.6,i5,a25,i5)') ((/dble(h),dble(k),dble(l)/)+basis(:,ib)), 1,
           'Nanoparticle no. of atoms', nat
          endif
        enddo
      enddo
    enddo
  enddo

  print*, nat

end
```

Since the reader might not be familiar with programming, let us know explain what each line in the previous code means. For a detailed explanation of FORTRAN language itself, we refer the reader to the relevant bibliography [8]. The very first line starts with program, the code which should be executed must be within the program block. At the end of the code there is also end which shows the end of the block. In the second line of the code there is the implicit none statement. This statement is used in order for the programmer to declare all the variables, which will be used. The reason someone uses this statement is to avoid possible mistakes which are related with the way FORTRAN codes were made when the language was first developed. Due to lack of memory the first codes were made as short as possible and the type of a variable had to do with the first letter of the variable, so I,J,K,L,M,N were denoting an integer and all the other letters a real number. This way of making programs was commonly resulting in mistakes so the safer implicit none statement is used.

In FORTRAN we have six types of data which are: real, double precision, complex, logical, integer, character. The double precision statement used for the variables **radius, alat, clat** and **r(3)** has to be explained now. During the calculations done, there will be real numbers which may have many decimal digits. Now the computer can't store in its memory all the digits, which in certain cases might be infinite (e.g. $e$, $\pi$). When the double precision statement has been declared before the variable, then the computer will use a certain number of bytes (usually 8 bytes) for the representation of the number and that will result in a certain number of decimal digits which will be kept and is usually 15 digits. The integer variables in this code are the indexes $h, k, l$ and their max values $h_{\max}, k_{\max}, l_{\max}$ and the **ib** which denotes a random atom, **nb** which denotes the number of atoms in the unit cell, thus sixteen, and the **nat** which is the total number of atoms. The character type is used in order to introduce the the chemical symbol of the atoms. This statement does what the name suggests, inserting characters as variables. The parentheses after the statement are used to declare the number of characters, which in our case is two and then we denote that it refers to the atoms of the unit cell with the **atbasis(16)**.

The reshape function is very important part of the code. The general structure of this function is reshape(source,shape), where the source is used in order for the data to be inserted in array order and the shape is used for the shape of of the array which will be produced. At this point we have to mention that an array in FORTRAN can be made by an array constructor which is a list of constants (real, integer) between the symbols **(/.../)**. For example the array $(/x, y, z/)$, where $x, y, z$ are real numbers, is a vector in a Cartesian coordinate system. So in our code we define the **basis** variable, which is nothing more than the unit cell, with the reshape function. The source part contains all the data in an array form and these are the position vectors of each atom in the unit cell. For example, for the first atom is in the $(0, 0, 0)$ position the vector will be of course $(/0., 0., 0./)$. The dot after each number is used in order to get a real number and not an integer when we must express the coordinate as a fraction.

For example, FORTRAN will give the value 0 to 1/2, so in order to get 0.5 one has to write 1./2. as a coordinate. The shape part could have been written as $(/3, 16/)$. Instead we write shape(basis) and we have already defined the dimensions of the basis to be (3,16) in the beginning with **basis(3,16)**. The same procedure has been followed and with the **atbasis** variable. The reshape function has been used and now the source part consists of the chemical symbolisms of the atoms in an array form and the shape instead of been written in the form $(/1, 16/)$, has been written as shape(atbasis) where the dimensions have been defined one more time in the beginning with the **atbasis(16)**. Now one should also notice the correspondence between the two arrays, where a position vector in the first corresponds to a specific atom in the second. We also mention that the ampersands (&) are used in FORTRAN in order to break a line.

Next we have to assign a value both to the real and the integer variables. The values of lattice parameters of the unit cell are known and in our case real variables. Moreover because $a = b$ we just give the value of $a$ assigned as **alat** and equating it with 5.800. The $c$ value has been assigned as **clat** and it is equal with 11.700. The lines following the exclamation marks (appeared with green color) are comments and do not taken into account during the compilation of the code. In our case we comment that all the values are measured in Ångström (Å). The next real variable, and probably the most important for our purpose, is the radius of the nanoparticle. It is also expressed in Å and the values used, since they do not appear in the code, were 5Å, 7.6Å and 12Å. After that we have assigned a value to an integer variable which is the number of the atoms in the basis (unit cell), written as **nb** and be equal with 16. Then we have to define the integer variables **hmax, kmax, lmax**. These three variables are the maximum values of the Miller indexes and have to be integer. For that reason we use the ceiling function, which has the general form ceiling(M) which gives the smallest integer which is greater than or equal to the argument M. So the maximum value of each index should be the integer given by the ceiling function when it's argument is **(radius/alat)** and then add one. For reasons that will become obvious shortly, the last integer variable which is the total number of atoms in the nanoparticle, **nat**, has been set equal to zero.

Now that we have set the maximum values of $h, k, l$ we want the code to assign to each index all the values from the negative of the maximum value until the maximum value. In order to accomplish that in FORTRAN one has to use the do statement. So by typing do **h=-hmax, hmax**, one says the code to assign in $h$ all the values between the selected range. This procedure is called a loop. A loop is taking place in oder for someone to get an output, shortly we will see which is the output we want and how we will get it. In addition, the loop has to start with the do statement and finish with the enddo statement. We repeat the same procedure we did for the $h$ index, for the $k$ and $l$ indexes and then we have do **ib=1, nb**, so that we include all the atoms in the unit cell.

After the last do statement, what we have done is to define a translation vector $\vec{r}$. In our case the components have been defined as $r(1), r(2), r(3)$ instead of the usual $x, y, z$ notation. At that point we have to justify for the way each component has been defined. We will do that for $r(1)$ and the same argumentation stands for $r(2)$ and $r(3)$. In the code it can be seen that the component has been defined as **r(1)=alat*(dble(h)+basis(1,ib))**. What we have actually done is to extend the $x$-component of the position vector of each atom in the basis (unit cell) by the lattice parameter $a$ multiplied by the index $h$. This is why we typed **basis(1,ib)**, it means the first column in the **basis** variable ($x$ component) and **ib** will take all the values from one to sixteen. This time he have used the dble function in order to convert the integer $h$ to a double precision real type. The general form is dble(M) and the argument M should be integer, real or complex. As mentioned before, the double precision has to do with the memory storage in the computer (8 bytes) and the number of decimal digits kept ( 15). By doing the same for $r(2), r(3)$ we can expand the unit cell and get the crystal structure.

Let us now summarize the previous. Up to now we have introduced certain variables in the code, we have set the position vector for each atom in the unit cell, as well as their chemical symbol and we have extend the unit cell in all three dimensions to get the crystal structure. Two points have still to be explained, why we set the total number of atoms (**nat**) equal to zero and why we introduced the loops for the $h, k, l$ indices. These two points will become clear after we explain what follows the definition of the translation vectors in the code.

After the $r(1), r(2), r(3)$ have been defined, the condition for the construction of nanoparticles should be introduced. In order to introduce the condition we use the if...then statement. What this statement does is to give as an output only these results that satisfy the condition introduced immediately after the if part. In our case the condition is the following **(sqrt(dot_product(r,r))<radius)**, which says

that we want to keep only atoms that are inside a given radius. The distance of an atom from the $(0,0,0)$ point is of course given by $r = \sqrt{\vec{r} \cdot \vec{r}}$ and we demand that to be less than the radius set in the previous part of the code and defined by the variable **radius**. So now we can see why the loops were inserted. Since we have to run the code for different radii and the maximum values of the indexes depend on the radius selected, the code will have to calculate the specific range between the minimum and the maximum value for each radius. In addition since the indices are inserted in the translation vector $\vec{r}$, the code has to calculate all the translation vectors for all the values of the $h, k, l$ indexes and then keep only the ones that satisfy the condition, that will be the wanted output mentioned before.

The part after the then statement is for the output of the code and it begins with the write statement. Without entering into details about the parentheses part we mention that we will get an output file in which the chemical symbol and the position vectors of each atom that fulfills the condition will be appeared. That can be seen by the fact that after the parentheses is written **atbasis(ib),r**, that means that we will get the chemical symbol given in the **atbasis** variable and a position vector for each of them. And now we can also justify why set the total number of atoms equal to zero in the beginning. In the code, it is written also after the write statement **nat=nat+1**, in that way the code understands that the *final* number of atoms will be the initial number of atoms which is zero plus the atoms that fulfill the condition, so just the latter atoms. After that there is one more write which is for a different type of output file. Since we did not use that file for the visualization of the nanoparticles in Jmol software, we won't explain that part here.

Finally, the print* statement says the code what to give as an output, which in our case is the number of atoms that fulfill the condition. In addition, due to the write statement we will also have a file with their spatial components. We compiled the above code for different radii and we got three different nanoparticles, each having a different number of atoms. In what follows we will see how the visualization with Jmol software took place.

## 3.2 Visualization with Jmol software

The Jmol is an open source JAVA viewer for chemical structures in 3D. Though it is mainly used in the fields of Chemistry and Biochemistry for the visualization of proteins, it is also very useful in order for someone to view the crystal structure of a material and in our case to visualize the nanoparticles. The program can be downloaded free from `http://www.jmol.org/`.

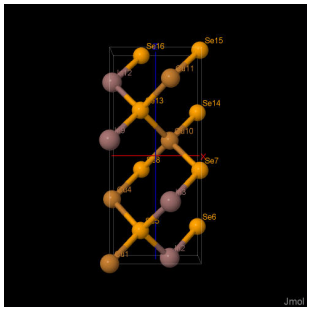



Figure 3: The $CuInSe_2$ unit cell.

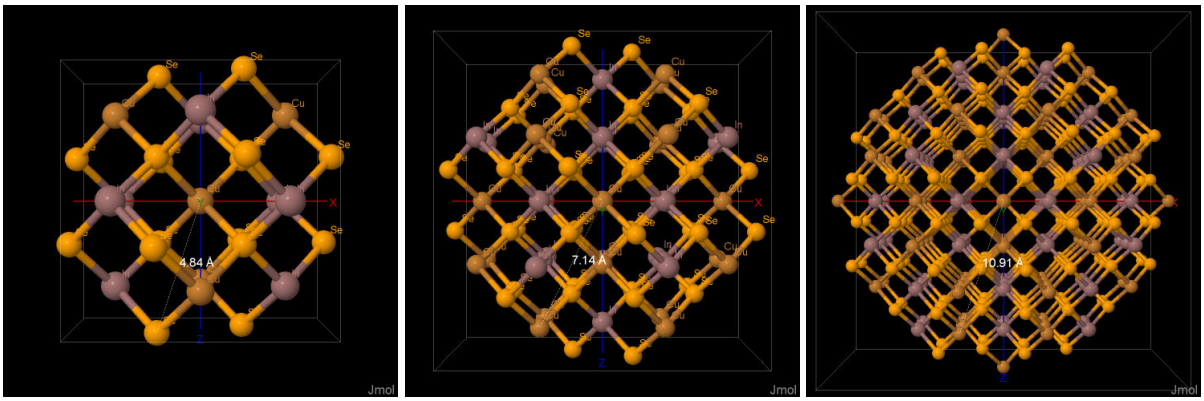
In order for Jmol to visualize a structure, one has to open with Jmol the xyz file that describes it. The xyz file is nothing more than a simple plain text file, where in the very first line there is the total number of atoms, then there is an empty line and in all the other lines there is the chemical symbol of an atom and its coordinates. To make the previous more clear we give the form of the xyz file describing methane molecule. If the file below is executed with Jmol we get the structure of methane molecule.

```
5

C    0.000000    0.000000    0.000000
H    0.000000    0.000000    1.089000
H    1.026719    0.000000   -0.363000
H   -0.513360   -0.889165   -0.363000
H   -0.513360    0.889165   -0.363000
```

Now is becoming obvious the usefulness of the code described above. What the code does in practice is to give everything we need for the xyz file. We have the number of atoms for each structure and the coordinates of the atoms. Due to the fact that the unit cell has 16 atoms, the shortest radius (5Å) structure has 29 atoms and the others even more, we can't write down the xyz files for the unit cell and the structures as we did for the methane molecule. The unit cell and the structures for the three different radii that the code was compiled, which are 5Å, 7.6Å, 12Å can be seen in figures (3), (4a), (4b), (4c), respectively. In these figures we have the nanoparticles made by the code described before. In order to proceed with calculations with the

Siesta DFT software, one has to hydrogenate the structures. A hydrogenated structure, also referred as hydrogen-terminated structure lacks unpassivated surface atoms and dangling bonds. The latter form when there are atoms with unfilled valence shells, so they form covalent bonds with other atoms. When dangling bonds occur, calculations are often unstable. In nature, dangling bonds are quickly saturated by forming bonds with nearby atoms or molecules, which can modify some properties of nanoparticles. In order to avoid dangling bonds in our calculations, the surface atoms are bonded with hydrogen atoms. That can be done using a different code, but that is outside the scope of this project and we won't describe that code in detail. In figures (5a), (5b), (5c) we have the hydrogenated structures for the 5Å, 7.6Åand 12Å respectively. There is one last thing which will become clear in the next subsection. If someone looks in figure (5a), then it can be seen that there are not only H atoms but also He atoms. That has to do with the calculations in Siesta, but still we referred to the structures of figure (5) as hydrogenated nanoparticles.

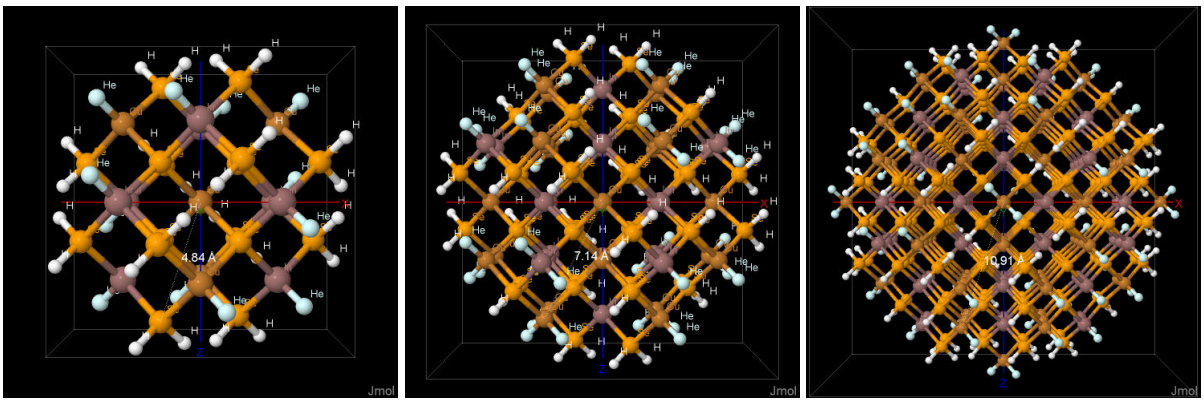



(a) Nanoparticle of 5Å radius, containing 29 atoms.

(b) Nanoparticle of 7.6Å radius, containing 87 atoms.

(c) Nanoparticle of 12Å radius, containing 293 atoms.

Figure 4: Different radii non-hydrogenated nanoparticles. Which color represents each atom can be seen in 4a and 4b.




(a) Hydrogenated nanoparticle of 5Å radius, containing 65 atoms.

(b) Hydrogenated nanoparticle of 7.6Å radius, containing163 atoms.

(c) Hydrogenated nanoparticle of 12Å radius, containing 465 atoms.

Figure 5: The hydrogenated nanoparticles of figures 4a, 4b, 4c. In 5a one can see which color represents each atom.

## 3.3 The SIESTA calculations

Up to now, we have constructed the nanoparticles and we have visualized the structures. What comes next is to take some insight about their physical properties, especially the electronic structure and the

energy gap. We remind in this point that our expectation is an increase of the energy gap when there is a decrease in the radius.

As we have seen in the theoretical part, solving equation (5) and getting the electronic structure and the energy eigenvalues, for such a system, is not possible. Approximations should be applied, but still the calculations are very complicated, so we have to make use of a code one more time. The code that will be used this is time is called SIESTA and it is made in order to do calculations about the physical properties of a system based on DFT.

SIESTA is a computer code, using DFT in order to calculate ground state properties of materials. Strictly speaking, we can not get any info by SIESTA about the excited states of the system. Moreover, SIESTA is using pseudopotentials in order to run calculations, that means as we saw in the theory part that only valence electrons are taken into account and when there are *semicore states* i.e. the overlap of the wavefunctions between the core and the valence electrons is non-negligible, they should be explicitly declared in the input file for SIESTA. As we will shortly see, that happened in our case with the Se atoms and we had to specify these semicore states.

In what follows, we will try to explain the main parts of a SIESTA input file. The reader can find a detailed description of SIESTA in the SIESTA tutorial [9]. The input file for SIESTA has an `.fdf` suffix and should be saved in the same directory with the files for the pseudopotentials (5 files in our case) which are usually in `.psf` format. The `.fdf` input file for the 5Å structure is the following:

```
WriteXML no
WriteCoorXmol yes
WriteEigenvalues yes

# System name (every output file will have that name) and system label
SystemName     template_nc_5_input
SystemLabel     CuInSe_5A

NumberOfSpecies 5
NumberOfAtoms 65

# Atoms present in the system
%block ChemicalSpeciesLabel
1 29 Cu
2 49 In
4 201 H_1.5
5 202 H_0.5
%endblock ChemicalSpeciesLabel

%block PAO.Basis
Se 2 nodes
n=4 1 2 P 1 S 0.15
5.0 5.0
n=4 0 2 S 0.15
5.0 5.0
%endblock PAO.Basis

# Basis set and parameters for the basis set
PAO.EnergyShift 200 meV
%block PAO.BasisSizes
Cu DZP
In DZP
Se DZP
H_1.5 DZP
H_0.5 DZP
%endblock PAO.BasisSizes
```

```
# Charge of the pseudohydrogen
%block SyntheticAtoms
4
1 2 3 4
1.500 0.000000 0.000000 0.000000
5
1 2 3 4
0.500 0.000000 0.000000 0.000000
%endblock SyntheticAtoms

# kgrid defined for non-periodic systems
kgrid_cutoff 0.0 Ang

# Used functional
xc.functional LDA
xc.authors CA

SpinPolarized .false.

# SCF convergence parameter
MeshCutoff 160.  Ry
MaxSCFIterations 500
DM.MixingWeight 0.01
DM.NumberPulay 5
DM.Tolerance 1.d-4
WriteDM .true.

SolutionMethod diagon
ElectronicTemperature 25 meV

# Relaxation parameters
MD.TypeOfRun cg
MD.NumCGsteps 1000
MD.MaxCGDispl 0.01 Ang
MD.MaxForceTol 0.04 eV/Ang

# Structure, in angstroms (Ang)
AtomicCoordinatesFormat Ang
%block AtomicCoordinatesAndAtomicSpecies
    0.000000    0.000000    0.000000    1# Cu 1
   -2.900000   -2.900000    0.000000    2# In 2
   -3.821106   -3.821106    0.929047    4# He 3
       ⋮                       ⋮
    0.580171    5.215838   -0.600106   5 # H 65
%endblock AtomicCoordinatesAndAtomicSpecies
```

Let us now explain what all the above mean. First we mention that lines starting with a "#" in a SIESTA input file are comments (like lines starting with a "!" in FORTRAN) and won't be taken under consideration during the compilation of the code. These lines are there to explain specifically what each part means.

The first three lines of the input file now have to do with the type and the form of the output file produced by SIESTA. Though they are not that important for our purpose we will a short explanation for each. The very first line is `WriteXML no` and says to the code not to write the output file as an XML file. The second line should result in one more file containing the coordinates of the atoms of the system in Ångström, since there is the option `yes`, `WriteCoorXmol yes`. That file should be read by $\mathrm{XM_{OL}}$.

The third line is similar in structure with the previous two, `WriteEigenvalues yes` and the result will be the appearance of the Hamiltonian's eigenvalues for the sampling $\vec{k}$ points in the output file. The reason we chose `yes` is because we specified the `Solution Method` to be `diagonal` as we will see shortly, also we need eigenvalues to determine the size of the energy band gap. Otherwise the option should have been `no`.

Now we can proceed to main part of the input file. As the first comment suggests, we should have a system name and a label for the system also. So after the options `SystemName` and `SystemLabel`, we have defined that we have the template of the input file for the nanoparticle of 5Å radius, written as `template_nc_5_input` and be the system name and the label `CuInSe_5A`. Then we have to define the number of different atoms in the system and the total number of atoms in the system after the options `NumberOfSpecies` and `NumberOfAtoms` respectively. In our case we have 5 different atom species and 65 atoms in total.

After that, we have a block where we have to define, which are the atoms present in the system and assign them a label. A block in a SIESTA input file starts with "%block" and the name of the block and ends with "%endblock" and one more time the name of the block. Then we have assigned a number from 1 to 5 for each element, next to the number there is the atomic number of the element and then the chemical symbol. Now there are two things that should be specified. First we see that we have two atomic numbers that are 201 and 202. Atomic numbers greater than 200 declare synthetic elements, in our case we have two types of pseudohydrogen (`H_1.5`,`H_0.5`). The subscripts denote the charge of each pseudohydrogen, which is $(3/2)e$ and $(1/2)e$ respectively. The reason why those pseudohydrogens are used is to compensate the lack of charge of the cations and the excess of charge of anions at the surface of the nanoparticle. If we see the structures shown in figure 5, we will notice that after the hydrogenation process He atoms are bonded to cations which are Cu and In atoms and H atoms are bonded to anions which are Se atoms. So the `H_1.5` represents the He and the `H_0.5` represents the H.

Then we have the block with the pseudo-atomic orbital basis (`PAO.Basis`). There we have defined the semicore states for the Se atom. SIESTA demands that we explicitly describe any semicore states, since the pseudopotentials used take into account only valence electrons as mentioned before. The way semicore states are defined is as follows. In the beginning we have to write down the chemical symbol of the atom for which semicore states appear, in our case Se. After that we have to give the number of the orbitals we use to describe the semicore states, in our case 2.

We have the line: `n=4 1 2 P 1 S 0.15`. Here we describe the orbital. The principal quantum number $n$ is 4 and the angular momentum quantum number $l$, denoted immediately after $n$ is equal to 1. Then we have the number 2, which is the number of $\zeta$ for the shell and the `P` denotes a shell of polarization functions will be constructed from the first zeta orbital of angular momentum $l$ (SIESTA tutorial [9]). Then the number 1 is the $\zeta$ number for the polarization shell created. Finally the `S` sets the split-norm parameter at 0.15, the number which follows `S`.

Next we have to describe the second orbital. This time the principal quantum number $n$ is again 4, but this time the quantum number of angular momentum $l$ is 0. The number of $\zeta$ is again 2 and now we have to polarization shell, the split-norm parameter is again equal to 0.15. The numbers under the description of each shell are the radii and due to the fact that we have 2 $\zeta$ we have each number two times under each shell. So in our case the radii are 5 Bohr.

The next part of the input file that we will describe now is the block `PAO.BasisSizes`. By size we mean the number of orbitals per atom. As we can see we have two radial functions per angular momentum channel double $\zeta$ denoted by `DZP`. The P at the end means that we can also have polarization shells. So for each atom we have a `DZP`.

Then we have to start a block describing the synthetic atoms. In this block block there are the numbers we have assigned to the synthetic atoms, in our case 4 and 5. Under these numbers there are the numbers 1,2,3,4, which show the number of valence electrons with given $l = 0, 1, 2, 3$ and under the latter there are the occupancy numbers. So we have one electron with $l = 0$ and zero electrons with $l = 1, 2, 3$, for both types of pseudo-hydrogens.

Finally it can be seen that the exchange correlation functional used is LDA with the parametrization CA (Ceperley-Alder). At this point we have to specify that we have to provide SIESTA a file with the pseudopotential used for each atom. There are certain programs doing that but for the purposes of this project we had the pseudopotentials already made. We won't explain in detail the rest of the input file, but as mentioned before we refer the reader to the SIESTA tutorial [9], where everything is explained.

# 4 Results and discussion

After we have seen how the input file looks like, we can proceed and present our results. After the calculations with SIESTA have been run we get several output files. The most important of them for our case is the one in which the Fermi energy of the system is written as well as the eigenvalues of energy. From the latter file we can get the density of states (DOS)[2] for each structure, using the `eig2dos` code, a `FORTRAN` code that comes with SIESTA.

After compiling the code for the output file showing the energy eigenvalues we get a list showing the number of states per energy eigenvalue. By plotting the DOS vs the energy and by knowing the Fermi energy, the energy band gap can be identified and its range can then be calculated. At this point we have to mark that though the SIESTA calculations are giving us the value of the Fermi energy for each structure, `eig2dos` code is shifting the Fermi energy to zero, so the energy band gap should be expected around zero in a DOS vs energy plot.

Due to the quantum confinement effect explained in the theory part, one should expect to see an increase in the energy band gap of the nanoparticle, as its radius reduces. From our calculations that was indeed the case and in the following DOS vs energy plots quantum confinement can be observed. Before presenting these plots, there is one more crucial point that has to be explained in order for the energy band gap to be identified. As mentioned before, in order to proceed to calculations with SIESTA the structures should have been hydrogen terminated first so no dangling bonds appear. These are nothing more than covalent bonds between atoms with unpaired electrons.

If now hydrogenation has not occurred properly, then there will be free electrons leading to dangling bonds. Moreover, these free electrons of the surface atoms will account for a state (wave-function) that has to be taken under consideration in the calculation and which will appear as an impurity in the DOS. That can result in some states even inside the energy band gap as it was the case in this project. In what follows we will present the DOS vs energy plots for the bulk material and the 5Å, 7.6Å and 12Å radius structures. Then we will calculate the range of the energy band gap and finally we discuss how the impurities in the energy band gap can be eliminated in future calculations.

The DOS vs energy plot for the bulk material can be seen in figure 6, where we see that the energy band gap range is $\approx 0.8$ eV.

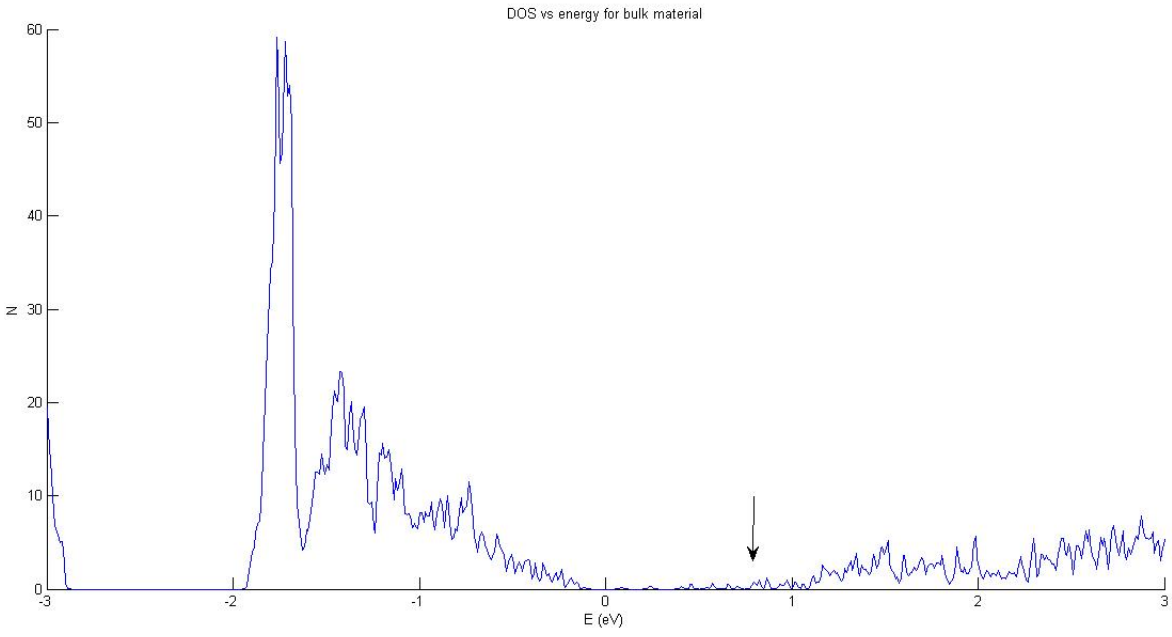



Figure 6: DOS vs energy for the bulk material. The range of the energy band gap is $\approx 0.8$ eV.

[2]We remind here that with the term DOS we refer to the number of states per energy unit, for every energy level, that are available to be occupied by electrons.

Because we observe spurious impurity states in all calculations, we take a pragmatic approach to estimate the band gap. Later we discuss possible ways how to improve these calculations. Here consider the end of the energy band gap at the point where the DOS starts to have a non-negligible variation again. Some negligible variations inside the energy band gap are considered as impurities. The previous can become obvious in the DOS vs energy plots for the 5Å ($E_g = 1.6$ eV), 7.6Å ($E_g = 1.4$ eV) and 12Å ($E_g = 1.2$ eV) structures, shown in figures 7, 8 and 9 respectively:

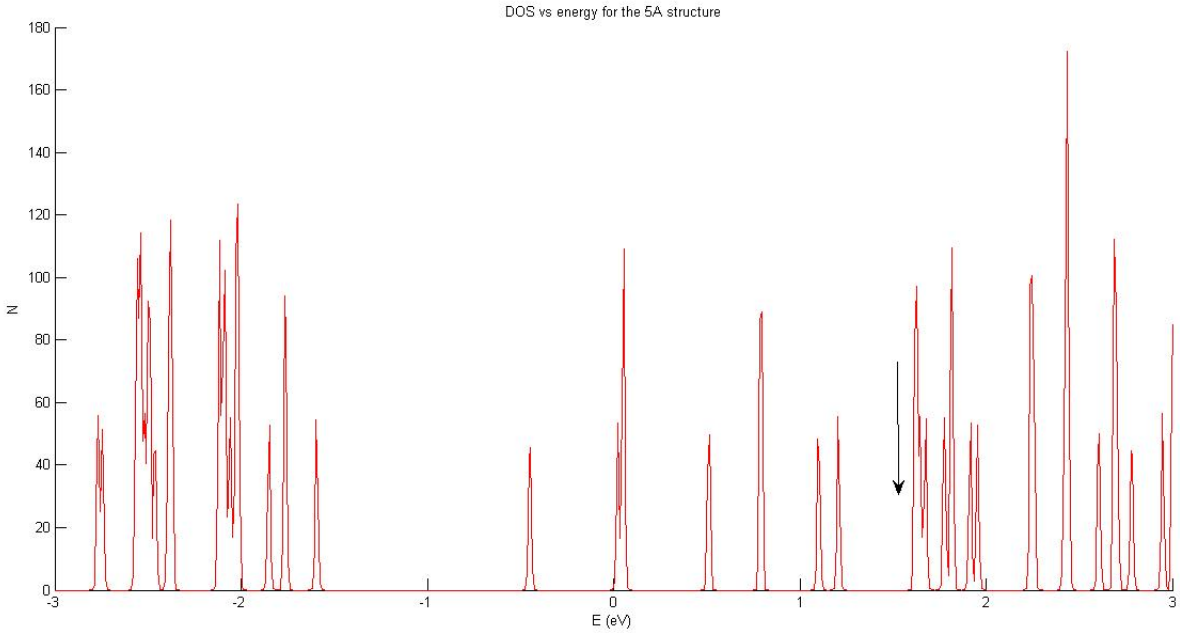



Figure 7: DOS vs energy for the 5Å structure. The range of the energy band gap is $\approx 1.6$ eV.

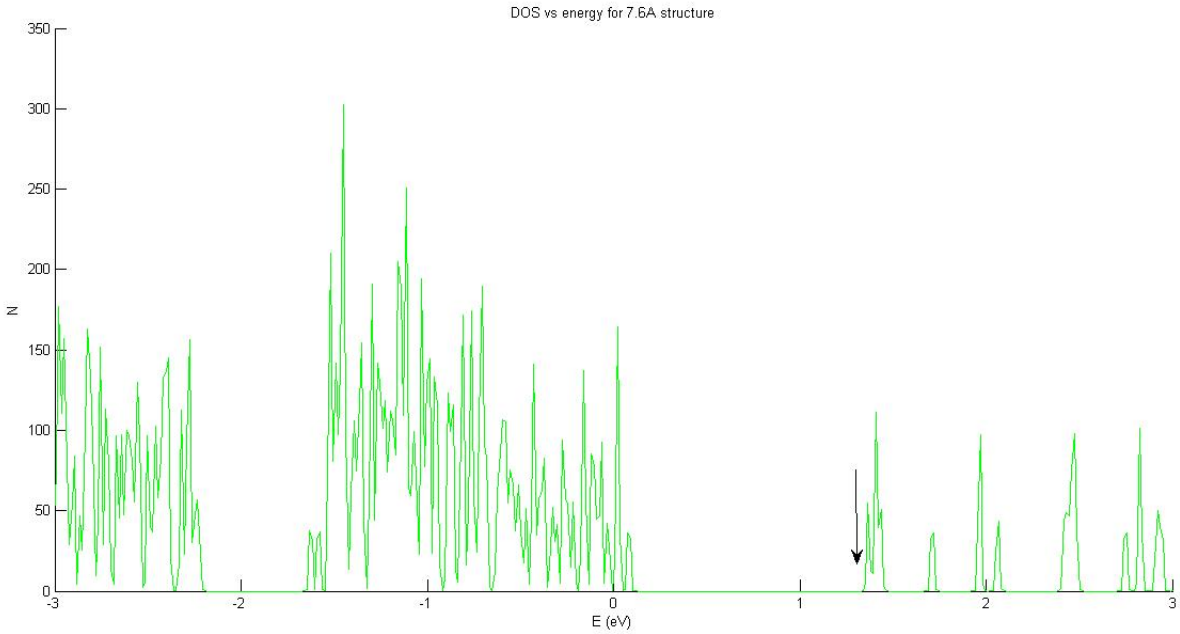



Figure 8: DOS vs energy for the 7.6Å structure. The range of the energy band gap is $\approx 1.4$ eV.

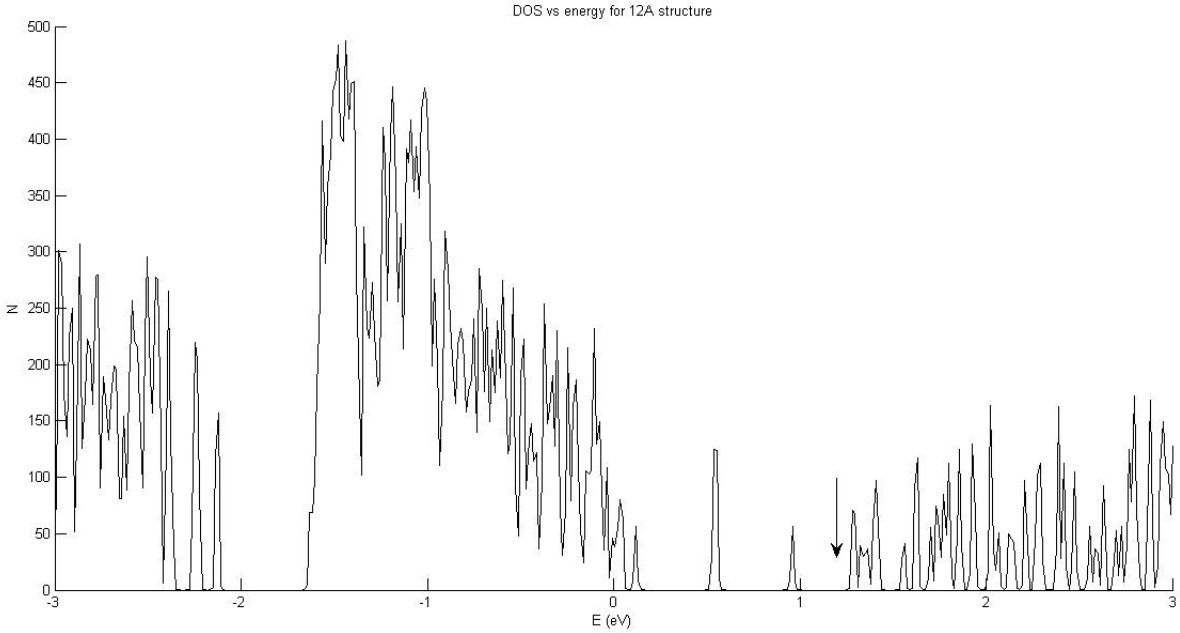



Figure 9: DOS vs energy for the 12Å structure. The range of the energy band gap is $\approx 1.2$ eV.

From the plots presented above, one should notice that quantum confinement effect can indeed be verified, as the range of the energy band gap increases with the decrease of the radius. That can become clear from plots 10 and 11, where in the first we see the DOS vs energy for all the structures and in the latter we have zoomed in the energy band gap region, showing where the energy band gap for each structure ends.

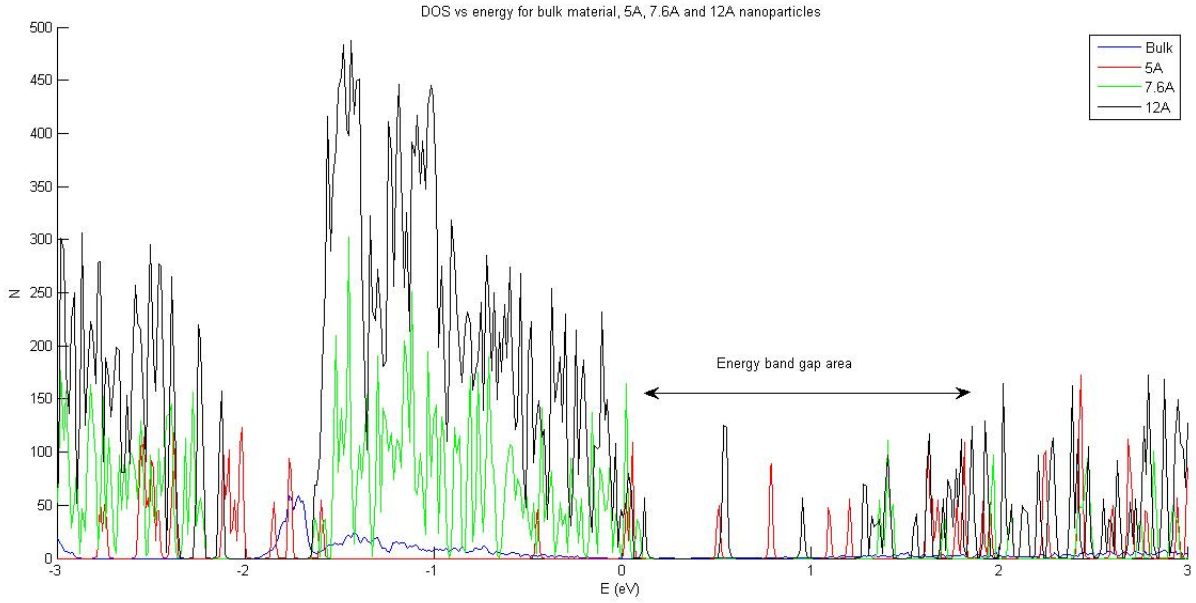



Figure 10: In the DOS vs energy plot showing all the structures, one can distinguish the different energy band gap for each structure and verify the quantum confinement effect.

In Table 1 we present for each structure, the calculated Fermi energy $E_F$ as it was given by SIESTA,

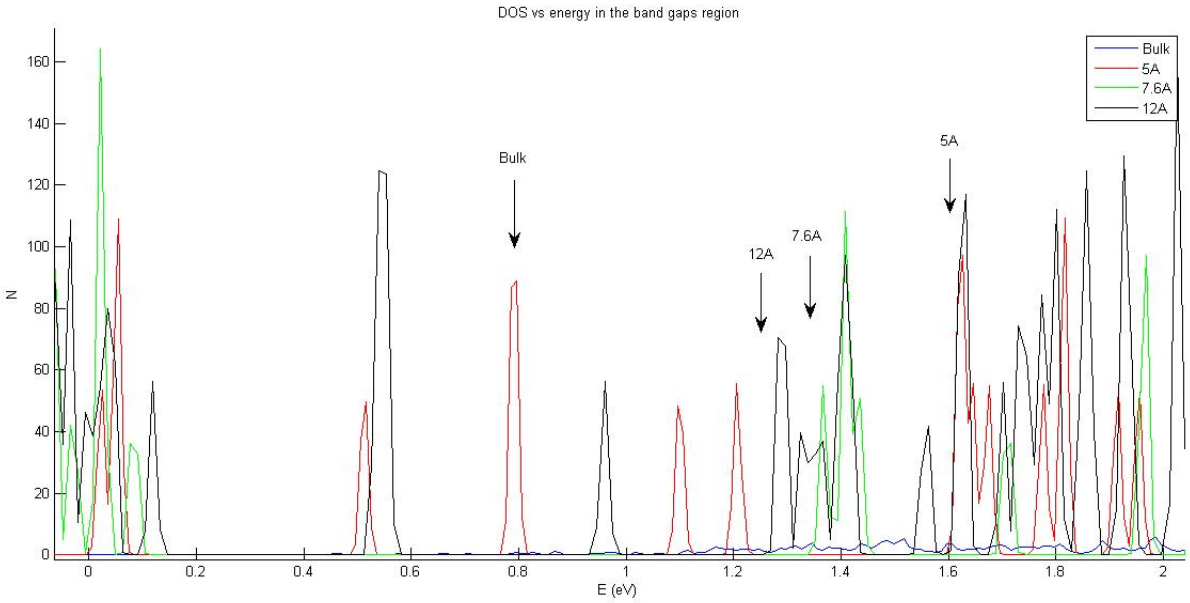



Figure 11: Zoomed DOS vs energy plot in the band gap region, showing where is the end of the energy band gap for each structure.

as well as the calculated energy band gap $E_g$. After that, in figure 12 we plot the radius vs the energy band gap in order to graphically verify quantum confinement.

Table 1: SIESTA results and calculated energy band gaps ($E_g$).

| Radius (Å) | $E_F$ (eV) | $E_g$ (eV) |
|---|---|---|
| Bulk | -3.28 | 0.8 |
| 12 | -4.37 | 1.2 |
| 7.6 | -4.73 | 1.4 |
| 5 | -2.77 | 1.6 |

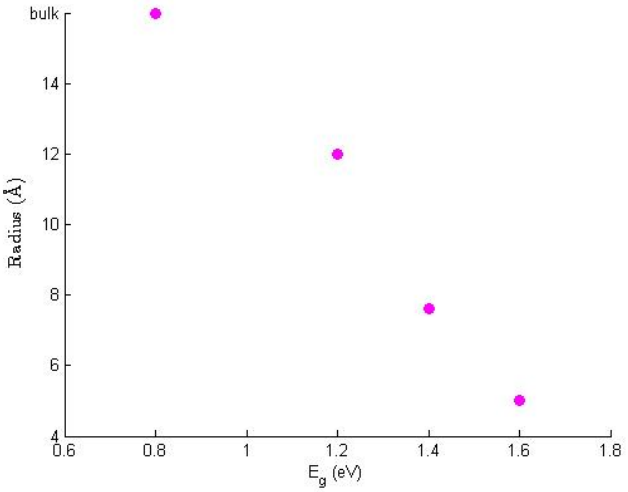



Figure 12: By plotting radius vs energy band gap, quantum confinement can be observed.

As mentioned before, quantum confinement can be verified. On the issue of the impurities, it is known that they are caused by free electrons belonging to surface atoms. This can be justified by the fact that they barely appear in the DOS vs energy plot of the bulk material (cf. figure 6). The contribution of the electrons of surface atoms for this structure is negligible compared the one of the bulk, so we get almost no impurities inside the energy band gap. On the other, one can observe many impurities in the DOS vs energy plot of the 5Å structure (cf. figure 7). We remind the reader that this structure contains just 29 atoms so the contributions from the electrons of the unpaired surface atoms not only can not be considered as negligible but might be the dominant ones.

In order for someone to get better results in future calculations, the following might be necessary. A more accurate code for hydrogenation of the structures so that there are no unpaired surface atoms causing dangling bonds. Furthermore, based on the current results, one should prepare the SIESTA input files in such a way that more accurate calculations will take place around the Fermi energy of each structure, so one could have a better insight regarding the DOS and the energy band gap. The previous tasks were beyond the scope of the current project, but still from the final results it can be concluded that quantum confinement effect can make it possible to construct in the future solar cells, the material of which they are made will have an energy band gap around the frequency with the maximal intensity of the sun light. That is proposed to have as a result an efficiency much greater than the bulk material.

## Acknowledgments

I would like to express my deep gratitude to Assistant Professor Jan Rusz, my main supervisor, for his patience guidance, useful critique of my work and enthusiastic encouragement throughout the duration of that project work.

I would also like to offer my special thanks to the Ph.D student Vancho Kocevski, for his valuable help in the computational part of that project, especially in the calculations with SIESTA and for being there to answer every question that I had.

## References

[1] Xuewen Wang, Yuan Zhang, Zhouhu Deng, Yujue Wang, Zidong Wang, I. Shih, *Journal of Crystallization Process and Technology*, **2**, 142-145, 2012

[2] S.Curtarolo, G.L.W.Hart, M.Buongiorno Nardelli, N.Mingo, S.Sanvito, O.Levy, *Nature Materials*, **12**, 191-201, 2013

[3] Robert.G.Parr, *Ann. Rev. Phys. Chem.*, **34**, 631-656, 1983

[4] J.C.Cuervas, *Introduction to Density Functional Theory* notes, `www-tfp.physik.uni-karlsruhe.de/~cuevas`

[5] P.Hohenberg, W.Kohn, *Physical Review B*, **136**, 864-870, 1964

[6] W.Kohn and L.J.Sham, *Physical Review A*, **140**, 1133-1138, 1965

[7] D.M. Ceperley, B.J. Alder, *Phys. Rev. Lett.*, **45**, 556, 1980

[8] Jeanne C.Adams, Walter S.Brainerd, Jeanne T. Martin, Brian T. Smith, Jerrold L.Wagner, *Fortran 90 Handbook*, Intertext Publications, McGraw-Hill Book Company

[9] Emilio Artacho *et al.*, SIESTA 3.2, `http://www.uam.es/siesta`