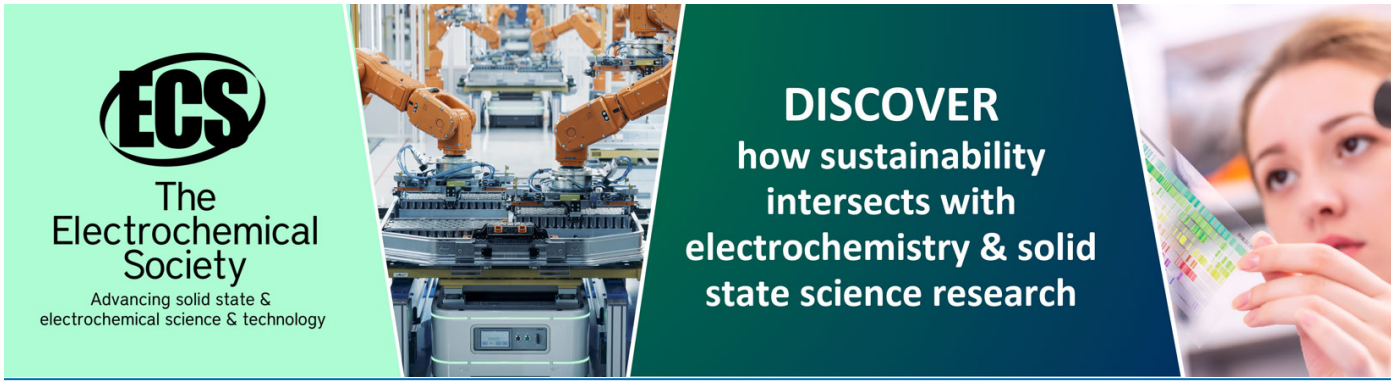
Implementation of density functional theory method on object-oriented programming (C++) to calculate energy band structure using the projector augmented wave (PAW)



IOP Conf. Series: Journal of Physics: Conf. Series **853** (2017) 012043 doi:10.1088/1742-6596/853/1/012043

Implementation of density functional theory method on object-oriented programming (C++) to calculate energy band structure using the projector augmented wave (PAW)

E Alfianto[1], F Rusydi[2*], N D Aisyah[2], R N Fadilla, H K Dipojono[3], M A Martoprawiro[4]

[1]Dep. of Computer Systems, Institut Teknologi Adhi Tama Surabaya, Jl Arif Rachman Hakim 100, Surabaya, Indonesia.
[2]Theoretical Physics Research Group, Dep. of Physics, Fac. of Science and Technology, Universitas Airlangga, Jl. Mulyorejo, Surabaya, Indonesia 60115
[3]Laboratory of Computational Material Design, Research Group of Engineering Physics, Institut Teknologi Bandung, Jl Ganesha 10, Bandung, Indonesia, 40132,
[4]Inorganic and Physical Chemistry Division, Dep. of Chemistry, Institut Teknologi Bandung, Jl Ganesha 10, Bandung, Indonesia, 40132

[*]Corresponding email address: rusydi@fst.unair.ac.id

Abstract. This study implemented DFT method into the C++ programming language with object-oriented programming rules (expressive software). The use of expressive software results in getting a simple programming structure, which is similar to mathematical formula. This will facilitate the scientific community to develop the software. We validate our software by calculating the energy band structure of Silica, Carbon, and Germanium with FCC structure using the Projector Augmented Wave (PAW) method then compare the results to Quantum Espresso calculation's results. This study shows that the accuracy of the software is 85% compared to Quantum Espresso.

1. Introduction

The research on material science increased rapidly for the last twenty years. It is caused by the increasing numbers of calculation and simulation method.[4]. They are Molecular Dynamic, Density Functional Theory, Hatree Fock, etc. All of these methods use the numerical method. Thus, we need the support of advanced computers with good software [6].

Generally, software is created using Fortran language [9]. Fortran language has advantages in numerical calculation. However, currently the Fortran language is less popular because of its complex usage [6]. This research presents a software which uses C++ programming language with Object-oriented programming rules. The created code is similar to mathematical formula (expressive software), hence the user will be able to understand each calculation process easily [8].

2. Computational method and implementation

2.1. Algorithm

Figure 1 shows a complete flowchart of eDFT, the name of this software. The process start from reading the input file and continue to define the basic variable. The basic variable contains basic information of calculation: the atomic number, the atomic mass, the number of electrons etc. After getting the basic variable information, we are able to calculate the sampling points and reciprocal lattice vector. The next process is to calculate potential and ewald energy.

In order to solve the Kohn Sham (KS) equation [3], we need to give an initial density. Initial density is needed to construct an effective potential. After solving KS equation, we will be able to get a new density. Then, the program will compare the new density to the initial density [2]. The comparison process is called evaluation. If the deviation is less than 0.000001, the program continue to calculate the total energy and eigenvalue.

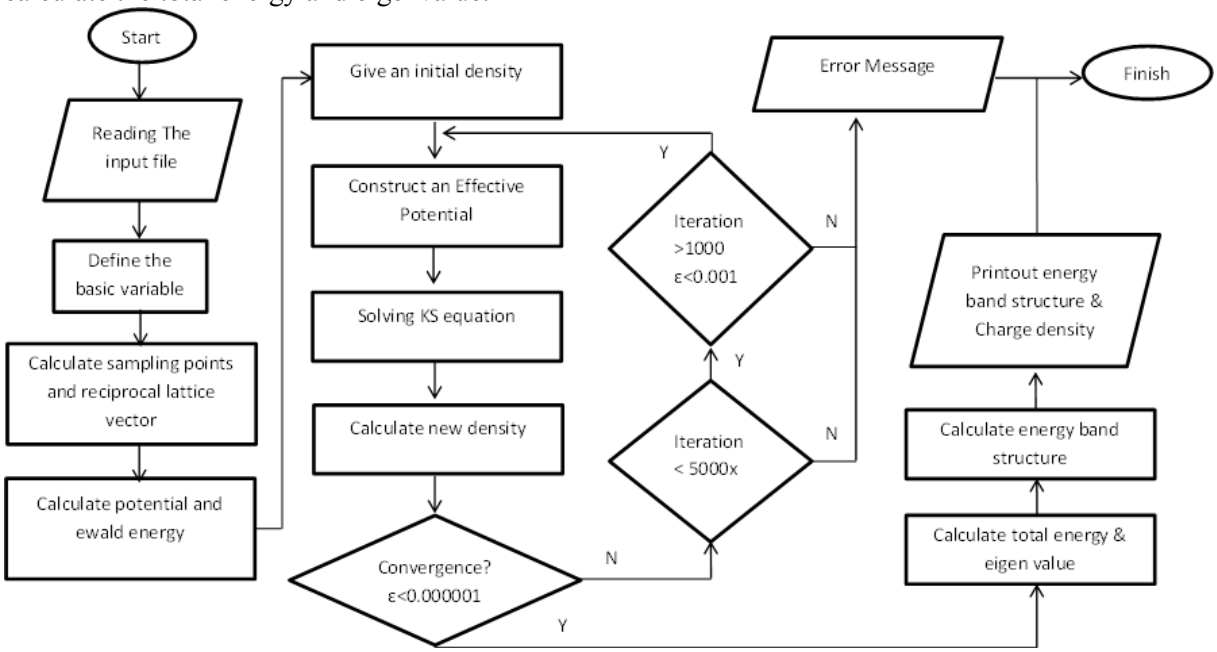


Figure 1. Algorithm of the eDFT

In order to find an energy band structure, we applied the Projector Augmented Wave (PAW) method for the calculation process. After the calculation is finished, the calculation results were printed in an output file. The output file contains the energy band structure and charge density data.

If the first evaluation process showed that the deviation is more than 0.000001, the second evaluation would be applied. The second evaluation wasto evaluate the iteration process. In this process, the iteration must be less than 5000 times. If the iteration wasmore than 5000 times, the calculation would stop and give an error message. Besides the evaluation, there was deviation checking. The deviation checking gives information about convergence: If the iteration was more than 1000 times and the deviation was more than 0.001, the calculation process would stop and give an error message. The aim of giving restriction on the iteration and the deviation is to avoid producing divergent values as the continuous calculation result.

2.2. Code Structure

eDFT is designed using object-oriented programming rule. The program is divided into many parts. The first part is master program and the second part is mathematical modules and installation instruction using Makefile. The mathematical module is divided into 6 files. Each file contains mathematical instructions and method to solve specific mathematical problem.

2.2.1. Master Program

Master Program is a code that handles all processes in the calculation. The master program is called eDFT. eDFT is divided into 3 part, eDFT.cpp, eDFT.h and eDFT.x. eDFT.cpp is a code based on C++ programming language which contains all calculation processes.

The expressive software is used to write the code. For example, we use the code of "std::vector<std::vector<int>> R=pac_Diag(Latice)" to define lattice vector. It means that we create vector into vector which contains integer variable "R". The letter R means diagonal value of Lattice matrix. Another example is we use "poisson(r,S,R,G2)" to calculate Poisson solver.

The control setting for output is also available in the master program. We are able to show or hide the calculation process when the job is run. This part is designed to be a simple tree. A simple tree means simple code structure. The aim of designing the simple tree is to facilitate the user to modify the program easily.

eDFT.x is a compiled file or binary file which is the result of compiling all codes. eDFT.h contains lists of header used in all calculation process. The header lists are from the standard C++ and library from the third party. The header lists are:

```
#include <vector>
#include <iostream>
#include <stdlib.h>
#include <math.h>
#include <complex.h>
#include <fftw3.h>
#include <string>
#include <fstream>
#include <sstream>
#include "OpenBlas/include/cblas.h"
```

2.2.2. Mathematical Module

Consisted of 7 files which were specifically designed for the calculation using DFT method. The first was"opmat.cpp". opmat stands for "Operasi Matriks (Matrix operation)". This file contained module for matrix operation, such as opmat_invers, opmat_kali, opmat_transpose, opmat_vmin etc. All the functions had opmat_function-name as the format.

The package for defining basic variable, sampling points and reciprocal lattice vector was poisson.cpp. The calculation process in the poisson.cpp used Matrix and Vector. We used opmat.h package to support calculation in poisson.cpp part. The use of opmat package would be more efficient if we compare it to the writing of every calculation code.

The DFT calculation process is contained in the ewald.cpp. The calculation process began with finding r at every point. In this calculation, we used matrix operation because all of the data were matrix and vector. Fast Fourier Transform was also used in this calculation to find "n" from many vector positions. Unfortunately, this calculation used complex numbers, so FFTW package from BLAS was used to calculate complex numbers[7].

The other support file was package.cpp.This part contains a package to calculate the value in the matrix "Prod", finding diagonal matrix "Diag" etc. This package is only aimed for support the DDFT calculation, but it is rarely used.

To get the print out of calculation result, out.cpp is created. This package uses standard output file in C++ (iostream). We modified the standard output (integer and double) to be a matrix and vector output. The illustration of file structure is shown in Figure 2

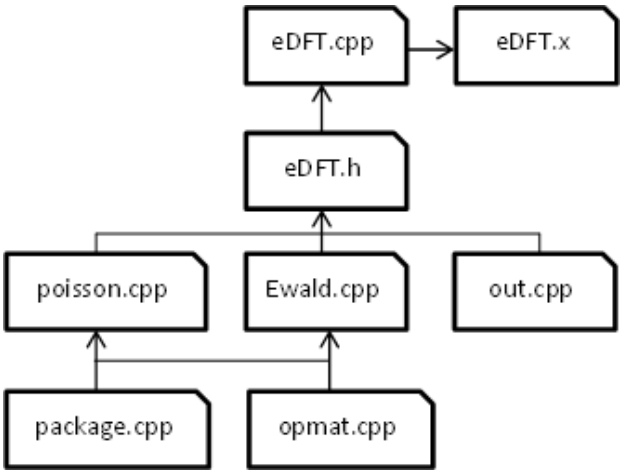


Figure 2. The eDFT File Structure

2.3. Finding sampling points and Reciprocal Lattice Vector

We have to find sampling points to calculate Reciprocal Lattice Vector [5]. The algorithm to define the sampling points and to calculate Reciprocal Lattice Vector is shown in Table 1.

Table 1. Algorithm to define Sampling points and to calculate Reciprocal lattice vector

Real-space Sampling points and Reciprocal lattice vector (RLV)
• Read vector S as an input file • Calculate capacity of vector in package.cpp (pac_prod(S)) • Find matrix M & N as a 3 dimensional representation of system • Calculate value of Sampling points (SP) as a matrix "r" and RLV as "G" matrix • Build matrix dimension from vector S as representation of input system r=M*pac_Diag(S) • Calculate RLV using G=2*PI*N/R

We need to calculate the charge density "n" to get information about charge distribution [5]. The solution of the charge density is

$$n = g_1(r) - g_2(r) = \frac{e^{-\frac{r^2}{2\sigma_2^2}}}{(2\pi\sigma_2^2)^{\frac{3}{2}}} - \frac{e^{\frac{r^2}{2\sigma_1^2}}}{(2\pi\sigma_1^2)^{\frac{3}{2}}}\sigma, \tag{1}$$

$$n = g_1(r) - g_2(r) = \frac{e^{-\frac{r^2}{2\sigma_2^2}}}{(2\pi\sigma_2^2)^{\frac{3}{2}}} - \frac{e^{\frac{r^2}{2\sigma_1^2}}}{(2\pi\sigma_1^2)^{\frac{3}{2}}}\sigma,$$

where r is space between two atoms from the proton, $g_1(r)$ and $g_2(r)$ are normalization from 3 dimensional Gaussian, and σ is radius of the system. The algorithm to calculate poisson solver is shown in Table 2.

Table 2. Algorithm to calculate Poisson solver

Poisson solver
• Matrix allocation "pac_prod(S)
• Create matrix "one". The value of all element is 1.
• Find value of "Sr" by times the matrix with 0.5.
• Times matrix "one" with the "Sr"
• Find Gaussian normalization "G"
• Calculate Phi in reciprocal using FFTW
• Find real number of Phi from reciprocal

2.4. Adding PAW

We added "eband.cpp" to find energy band structure. eband.cpp is a code based on Plane Augmented Wave (PAW). The calculation begins with solving the Schrodinger-like equation that have been finished in "ewald.cpp" and "poisson.cpp". The wave function with plane wave is given by

$$\psi_{nk} = \sum_G c_{nk}(G) e^{i(G+k)r}, \tag{2}$$

$$\psi_{nk} = \sum_G c_{nk}(G) e^{i(G+k)r},$$

where ψ_{nk} is wave function, n is band number, k is point in Brillion Zone, G is reciprocal Lattice Vector.

Wave function in real space is given by

$$\rho(r) = \sum_n \sum_k \psi_{nk}(r)^* \psi_{nk}(r), \tag{3}$$

$$\rho(r) = \sum_n \sum_k \psi_{nk}(r)^* \psi_{nk}(r),$$

where $\rho(r)$ is wave function in real space. Then, we calculated Hamiltonian from the plane wave matrix element, which is notated in dirac as follows:

$$\langle k+G'|H|k+G\rangle = |k+G|^2 + \langle k+G'|\hat{V}_H + \hat{V}_{xc} + \hat{V}_{pp}|k+G\rangle. \tag{4}$$

$$\langle k+G'|H|k+G\rangle = |k+G|^2 + \langle k+G'|\hat{V}_H + \hat{V}_{xc} + \hat{V}_{pp}|k+G\rangle. \tag{5}$$

$$\langle k+G'|H|k+G\rangle = |k+G|^2 + \langle k+G'|\hat{V}_H + \hat{V}_{xc} + \hat{V}_{pp}|k+G\rangle.$$

The symbol $\hat{V}_H$ is Hartree Potential, $\hat{V}_{xc}$ is potential from exchange and correlation, and $\hat{V}_{pp}$ is empirical pseudopotential. Then the Hartree potential in reciprocal space is

$$\langle k + G'|\hat{V}_H|k + G\rangle = \frac{8\pi\rho(G' - G)}{|G' - G|^2} \quad (6)$$

$$\langle k + G'|\hat{V}_H|k + G\rangle = \frac{8\pi\rho(G' - G)}{|G' - G|^2}$$

The algorithm to get the energy band structure is shown in Table 3.

Table 3. Algorithm to calculate Energy band structure

Energy Band Structure

• Calculate G vector
• Determine k-points
• Insert the initial charge density (Using Gaussian function)
• Calculate Reciprocal space using Fourier transform
• Calculate using self-consistency method (Calculate Hamiltonian)
  o Calculate Hartree potential
  o Calculate exchange and correlation potential
  o Calculate pseudopotential
• Calculate band energy in every k-points.

.

3. **Result and Discussion**

The first calculation process to create G Vector which was shown in Table 1. The aim of this calculation is to plot the value in three dimension coordinate. The complete test of G vector is shown in Figure 3. The value test in Figure 3 used the length of cell of 6 Bohr and matrix input S = [20;25;30]. The calculation result shows that G vector has three-dimensional value.

The next test is the test for Poisson solver calculation package. The algorithm of this calculation is shown in Table 2. The result of Poisson test is shown in Table 4.

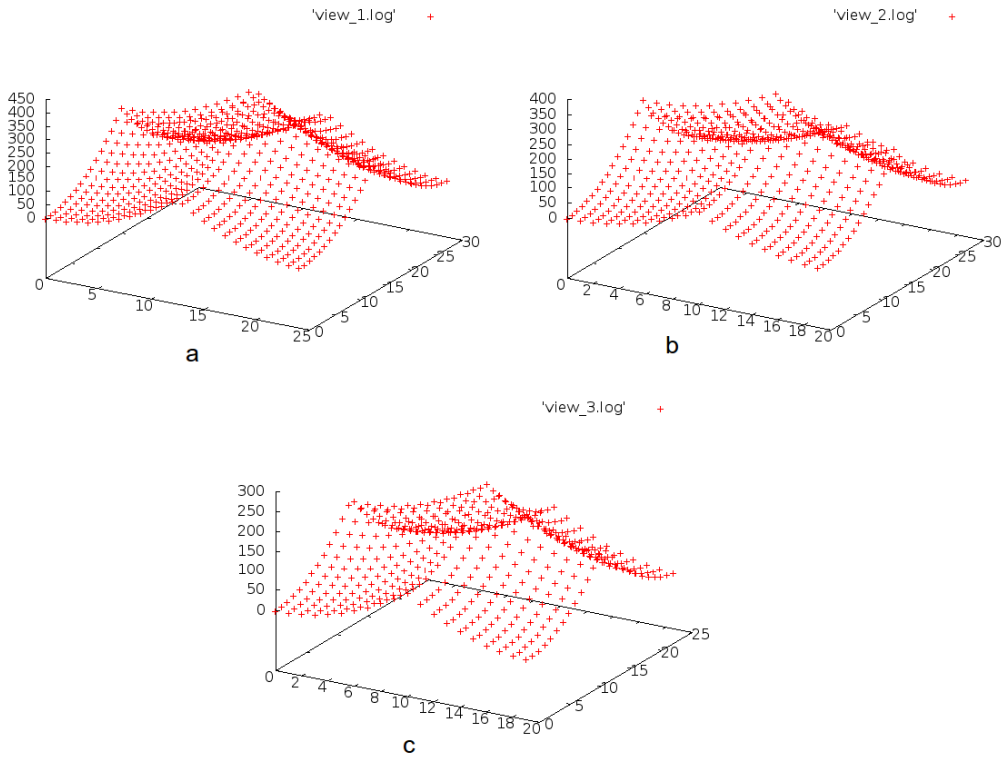


Figure 3. (a). G vector in slice x. (b) G vector in slice y. (c) G vector in slice Z

Table 4. Poisson calculation test results

	Analytic	eDFT
Gaussian Normalization	1	0.999781
Coulomb Potential energy	0.0551425	0.0551405

The final calculation was to find the energy band structure of Silicon as a system tests. The calculation result is shown as energy band structure graph at every k-points. The Silicon energy band structure graph as the result of eDFT calculation is similar to the energy band structure graph as the result of Quantum Espresso calculation. The similarities between them reaches 85%. Figure 4 shows us the energy band structure as the result Quantum Espresso calculation and eDFT. The trend of the graph is similar.

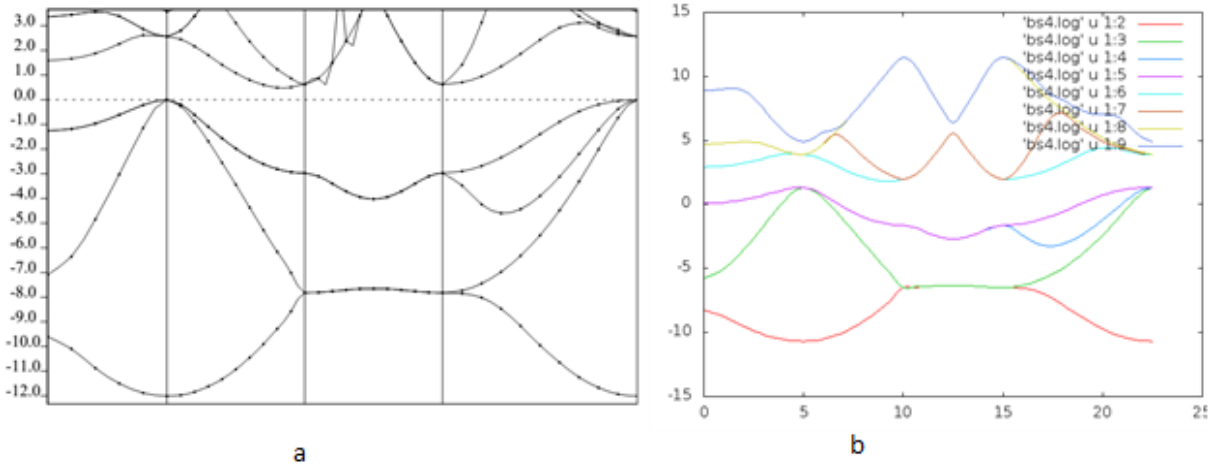


Figure 4. (a) Energy band structure of silicon calculate by Quantum Espresso, (b) Energy band structure of Silicon calculate by eDFT

4. Conclusion

We have implemented DFT which uses PAW method to calculate energy band structures. Silicon is chosen as the sample for the calculation. We have written the software using expressive software paradigm which can be easily understood by users. The result shows that our eDFT calculation result is 85% similar to Quantum Espresso calculation result.

Acknowledgement

This calculation of this work is supported by computer research facility of Computational Material Design laboratory, Institut Teknologi Bandung, Indonesia.

References

[1] Born M and Huang K 1988 *Dynamical Theory of Crystal Lattices* Oxford Classic Texts in the Physical Sciences Clarendon Press Oxford.
[2] Hohenberg and W Kohn 1964 *Inhomogenous electron gas* Phys Rev 136(3B):B864â871
[3] Kohn and L J Sham 1965 Self-consistent equations including exchange and correlation effects Phys Rev. 140 4A:A1333â1138
[4] SS David 2009 *Density Functional Theory A practical introduction* John Willey and sons Inc.
[5] S Ismail-Beigi and T A Arias Comp. Phys. Comm. 128 1.
[6] R Carry, Stevlana and G Shasharina 196 Comparison of C++ and Fortran 90 for Object-Oriented scientific programming *Computer Physics Communications* 105 (1997) 20-36
[7] Fathurahman F, Alfianto E, Dipojono, H K, and Martoprawiro, M A (2014, September). Acceleration of norm-conserving Pseudopotential Plane-Wave-Based DFT Calculation on GPU using CUDA. In *Proceeding of 3rd International Conference on Computation for Science and Technology* Vol. 2
[8] Alfianto, E. 2015 Implementasi Metode Teori Fungsional Kerapatan pada bahasa C untuk menemukan energi keadaan dasar berbagai atom. *Jurnal Arus Elektro Indonesia*, *1*(3).
[9] Ratcliff, L E, Grisanti, L, Genovese, L, Deutsch, T, Neumann, T, Danilov, D and Cornil, J 2015 Toward Fast and Accurate Evaluation of Charge On-Site Energies and Transfer Integrals in Supramolecular Architectures Using Linear Constrained Density Functional Theory (CDFT)-Based Methods. *Journal of chemical theory and computation* 11(5)