Pisco-Med Publishing

Solve and Implement the main Equations of Photovoltaic Cell Parameters Using Visual Studio Program

Mohammed RASHEED[1], Mohammed Abdelhadi Sarhan[2 *]

[1] Applied Sciences Department, University of Technology-Baghdad- Iraq

[2] College of Science, Department of Mathematics, Al-Mustansiriyah University-Baghdad- Iraq

Abstract: In the present work, the main equations of Photovoltaic cell parameters are designed and implemented using visual Studio program. The important equations parameters of a solar cells is short-circuit current I_{sc}, open-voltage circuit V_{oc}, the voltage, the current at maximum power V_m and I_m respectively, incident power density, maximum power P_m, fill factor FF and the photovoltaic efficiency of the cell η.

Keywords: Important parameters, photovoltaic cell, visual studio, C sharp language

Received: 20th Jul. 2019 **Accepted:** 12th Aug. 2019 **Online:** 17th Aug. 2019

1. Introduction

C Sharp language is an extension of the language C and C++ and ease of Visual Basic and JavaScript was released in June 2000, it is still fledgling. Were created by Microsoft by team Microsoft led Anders Hejlsberg an engineer distinct Microsoft was produced products and other languages, contains Borland Delphi and Borland Turbo C++ and focused engineer Elsie sharp to take the strengths that characterize the other languages with add enhancements to do this a better language. C sharp is strong and flexible like the rest of the languages we can create many different applications[1]. Valsa sharp depends only on your imagination. Do not put this language restrictions on what you are doing have been used in projects with multiple forms as dynamic web sites and development tools and translation software and many others. The C language was created sharp programming language using objects. Many believe that there is no need for new programming languages, languages found Kgava and the C++, Delphi and visual basic enough to meet all needs[2]. Although the CIA sharp derived from that C and C++ has been created from scratch without relying on those languages and included several new features making it easier ans similar to Java in some similar characteristics[3-4]. Solar cells are the procedure of converting intensity of light immediately to electric charge using photovoltaic effect. Many kinds of PV can be fabricated based on the materials used and the method of fabrication in order to obtain economic solar cells with high efficiencies such as; indium tin oxide (ITO), niobium doped titanium dioxide (TiO_2:Nb), zinc oxide (ZnO), , nickel oxide (NiO), Bismuth trioxide (Bi_2O_3) and aluminum doped zinc oxide (AZO) which are used in rigid and flexible electronics[5-9]. Many kinds of photovoltaic cell such as silicon, inorganic and organic solar cells[10-14]. In addition several good materials can be utilized to enhanced the efficiency of solar cells because of their excellent optical properties such as zinc tungstate, sodium tungstate, sodium cadmium orthophosphate, copper doped zinc oxide[15-19]. For other applications of solar cells in space based on Kepler equation; in addition using linear programming and fuzzy logic and set theory[20-31].

In this paper, the main equations of solar cell physical parameters have been demonstrated and implemented using Visual Studio program. The advantages of using C sharp are simple, modern, a strong and elastic language, using the programming language objects.

2. Experimental Method

In this study, the method including Visual Studio C# program a design method to calculate PV physical parameters and conversion efficiency evolutionary algorithms were briefly described.

3. Results and Discussion

The design of the main equations to solve and implement the output of the solar cell parameters for any values of the input parameters using C sharp program; the major physical factors are utilized to describe the rendering of PV cell are the maximum power P_{max}, current density J_{sc}, V_{oc}, and FF.

4. The Open Circuit Voltage V_{oc}[32]

Eq. for V_{oc} is obtained by precision $I = 0$; this equation indicate the linearity behaviour of V_{oc} that decreases with temperature in (Kelven).

$$.V_{oc} = \frac{KT}{q} \ln \frac{I_L}{I_o} \qquad 1$$

where $T\,(K) = 273.15 + T^o(C)$

The V_{oc} can too be calculated based on a carrier concentration

$$V_{oc} = \frac{KT}{q} \ln \frac{(N_A + \Delta n) \times \Delta n}{{n_i}^2} \qquad 2$$

where $\frac{KT}{q}$: thermal voltage, N_A : doping concentration, Δn: carrier concentration (excessing) and n_i: carrier concentration (intrinsic). The calculation of V_{oc} from the n_i.

Eqs. 1 and 2 is designed and implemented to calculate the value of V_{oc} shown in Figs. 7, 10.

5. Solar Cell Efficiency η[33]

PV quantum efficiency is calculated from a part of incident energy which is transmitted to electric current

$$\eta_1 = \frac{I_m \times V_m}{V_{oc} \times I_{sc}} \times 100\% \qquad 3$$

$$\eta_2 = \frac{V_{oc} \times I_{sc} \times FF}{E \times A} \times 100\% \qquad 4$$

$$\eta_3 = \frac{V_{oc} \times I_{sc} \times FF}{P_{in}} \times 100\% \qquad 5$$

$$\eta_4 = \frac{P_m}{P_{in}} \times 100\% \qquad 6$$

$$P_m = I_m \times V_m \qquad 7$$

and

$$P_{in} = E \times A \qquad 8$$

$$\eta = \frac{P_m}{P_{new}} \times 100\% \qquad 9$$

$$\eta = \frac{I_m \times V_m}{P_{in}} \times 100\% \qquad 10$$

where

η : PV efficiency, E: energy photon, A: area of the cell.

Eqs. 3, 4, 5, 6, 7, 8, 9, 10 are used to calculate the value of fill factor is illustrated in **Figure 3**, **Figure 5-Figure 9**.

6. Maximum Power[34-35]

The equations below indicate the relation between the maximum power and the maximum current and voltage

$$P_m = I_m \times V_m, \qquad 11$$

$$= J_m \times V_m \qquad 12$$

$$= V_{oc} \times I_{sc} \times FF \qquad 13$$

$$P_{new} = P_{old} \times A \qquad 14$$

The Eqs. 11, 12, 13, 14 are utilize to calculate an efficiency of a PV cell and designed as shown in **Figure 4- Figure 8**.

7. Fill Factor FF[34-35]

FF is described based on the ratio of maximum output power divided on V_{oc} multiplied by I_{sc} of the PV cell, so that

$$FF = \frac{P_m}{V_{oc} \times I_{sc}} \qquad 15$$

.

$$= \frac{I_m \times V_m}{V_{oc} \times I_{sc}} \qquad 16$$

when

$$\frac{d(IV)}{dV} = 0$$

$$V_{Mp} = V_{oc} - \frac{n \times K \times T}{q} - \ln\left(\frac{q \times V_{Mp}}{n \times K \times T} + 1\right) \qquad 17$$

where ν_{oc}: normalized V_{oc} and the above equation needs Lambert functions to solve but a simpler approach is to use iteration to calculate V_{Mp}.

$$FF = \frac{\nu_{oc} - \ln(\nu_{oc} + 0.72)}{\nu_{oc} + 1} \qquad 18$$

ν_{oc}: normalized V_{oc}

$$\nu_{oc} = \frac{q}{nKT} \times V_{oc} \qquad 19$$

The Eqs. 15, 16, 17, 18, 19 are utilize to calculate the efficiency of a PV shown in Figs. 6, 1, 11, 7, 2,

All the figures with the programs are listed below

```
private void button1_Click(object sender, EventArgs e)
        {


            Voc = Convert.ToDouble(textBox1.Text);
            n = Convert.ToDouble(textBox2.Text);
            k = Convert.ToDouble(textBox3.Text);
            t = Convert.ToDouble(textBox4.Text);
            q = Convert.ToDouble(textBox5.Text);
            Vm = Convert.ToDouble(textBox6.Text);

            k1 = Convert.ToDouble(textBox8.Text);
            q1 = Convert.ToDouble(textBox9.Text);

            p1 = k * Math.Pow(10, k1);
            p2 = q * Math.Pow(10, q1);


        Vmp = Voc - ((n * p1 * t) / p2) * Math.Log((Vm / ((n * p1 * t) / p2)) + 1);

            textBox7.Text = Vmp.ToString();
        }
  private void button2_Click(object sender, EventArgs e)
        {

            k = Convert.ToDouble(textBox3.Text);
            k1 = Convert.ToDouble(textBox8.Text);
            p1 = k * Math.Pow(10, k1);
            textBox10.Text = p1.ToString();
        }
```

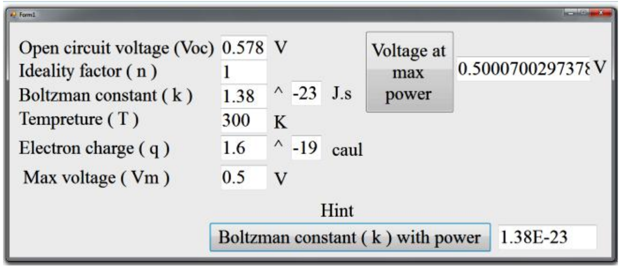

Figure 1. the value of the open-circuit voltage.

\>>

```
 private void button1_Click(object sender, EventArgs e)
        {
            double q, n, k, t, Vold, Vnew, k1, q1, p1, p2;

            q = Convert.ToDouble(textBox1.Text);
            n = Convert.ToDouble(textBox2.Text);
            k = Convert.ToDouble(textBox3.Text);
            t = Convert.ToDouble(textBox4.Text);
            Vold = Convert.ToDouble(textBox5.Text);

            k1 = Convert.ToDouble(textBox8.Text);
            q1 = Convert.ToDouble(textBox7.Text);

            p1 = k * Math.Pow(10, k1);
            p2 = q * Math.Pow(10, q1);


            Vnew = (p2 / (n * p1 * t)) * Vold;

            textBox6.Text = Vnew.ToString();

        }
```

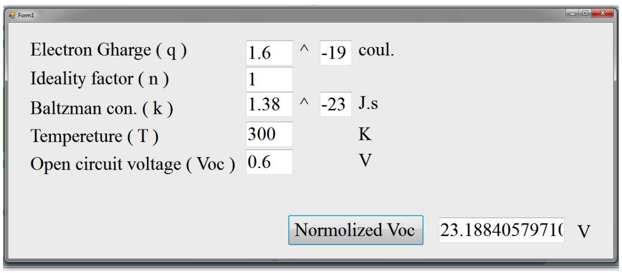

Figure 2. The value of open-circuit voltage.

\>>

```
 private void button1_Click(object sender, EventArgs e)
        {
            double Imax, Vmax, Pin, eff;
```

```
            Imax =
Convert.ToDouble(textBox1.Text);
            Vmax =
Convert.ToDouble(textBox2.Text);
            Pin =
Convert.ToDouble(textBox3.Text);

            eff = (((Imax * Vmax) / Pin) *
100);

            textBox4.Text =
eff.ToString();
        }
```

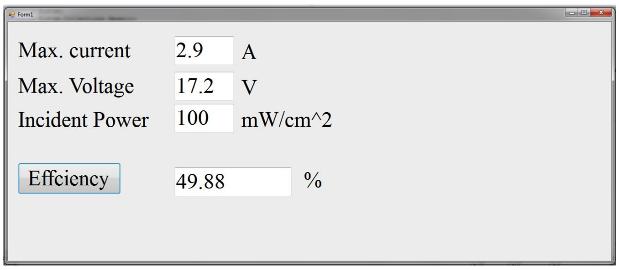

Figure 3. The value of efficiency.

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
private void button1_Click(object sender,
EventArgs e)
        {
            double Imax, Vmax;

            Imax =
Convert.ToDouble(textBox1.Text);
            Vmax =
Convert.ToDouble(textBox2.Text);

            textBox8.Text = (Imax *
Vmax).ToString();
        }

        private
void button2_Click(object sender,
EventArgs e)
        {
            double Jmax, Vmax2;

            Jmax =
Convert.ToDouble(textBox3.Text);
            Vmax2 =
Convert.ToDouble(textBox4.Text);

            textBox9.Text = (Jmax *
Vmax2).ToString();
        }

        private
void button3_Click(object sender,
EventArgs e)
        {
            double Isc, Voc, FF;

            Isc =
Convert.ToDouble(textBox5.Text);
            Voc =
Convert.ToDouble(textBox6.Text);
            FF =
Convert.ToDouble(textBox7.Text);

            textBox10.Text = (Isc * Voc *
FF).ToString();
        }
```

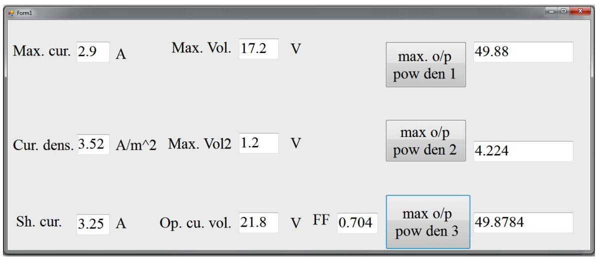

Figure 4. The value of maximum power density.

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
  using System;
  using System.Collections.Generic;
  using System.ComponentModel;
  using System.Data;
  using System.Drawing;
  using System.Linq;
  using System.Text;
  using System.Windows.Forms;

  namespace WindowsFormsApplication1
  {
      public partial class Form1 : Form
      {
          public Form1()
          {
              InitializeComponent();
          }
          double Im, Vm, Isc, Voc, FF, E,
A, Pin, Pm;

          private
void button1_Click(object sender,
EventArgs e)
          {
              Im =
Convert.ToDouble(textBox1.Text);
              Vm =
Convert.ToDouble(textBox2.Text);
              Isc =
Convert.ToDouble(textBox3.Text);
              Voc =
Convert.ToDouble(textBox4.Text);

              textBox10.Text = ((((Im * Vm)
/ (Isc * Voc)) * 100) ).ToString();
          }

          private
void button2_Click(object sender,
EventArgs e)
          {
              Isc =
Convert.ToDouble(textBox3.Text);
```

```
            Voc =
Convert.ToDouble(textBox4.Text);
            FF =
Convert.ToDouble(textBox5.Text);
            E =
Convert.ToDouble(textBox6.Text);
            A =
Convert.ToDouble(textBox7.Text);

            textBox11.Text = ((((FF *
Isc * Voc) / (E * A)) * 100) ).ToString();

        }

        private
void button3_Click(object sender,
EventArgs e)
        {
            Isc =
Convert.ToDouble(textBox3.Text);
            Voc =
Convert.ToDouble(textBox4.Text);
            FF =
Convert.ToDouble(textBox5.Text);
            Pin =
Convert.ToDouble(textBox8.Text);

            textBox12.Text = ((((FF *
Isc * Voc) / (Pin)) * 100) ).ToString();
        }

        private
void button4_Click(object sender,
EventArgs e)
        {
            Pin =
Convert.ToDouble(textBox8.Text);
            Pm =
Convert.ToDouble(textBox9.Text);

            textBox13.Text = ((((Pm) /
(Pin)) * 100) ).ToString();
        }

        private void Form1_Load(object
sender, EventArgs e)
        {

        }

        private
void button5_Click(object sender,
EventArgs e)
        {

            Im =
Convert.ToDouble(textBox1.Text);
            Vm =
Convert.ToDouble(textBox2.Text);
            Isc =
Convert.ToDouble(textBox3.Text);
            Voc =
Convert.ToDouble(textBox4.Text);

            textBox5.Text = ((Im * Vm) /
(Isc * Voc)).ToString();
        }

        private
void button6_Click(object sender,
EventArgs e)
        {
            Im =
Convert.ToDouble(textBox1.Text);
            Vm =
Convert.ToDouble(textBox2.Text);

            textBox9.Text = (Im *
Vm).ToString();
        }

        private
void button7_Click(object sender,
EventArgs e)
        {
            E =
Convert.ToDouble(textBox6.Text);
            A =
Convert.ToDouble(textBox7.Text);

            textBox8.Text = (E /
A).ToString();
        }
    }
  }
```

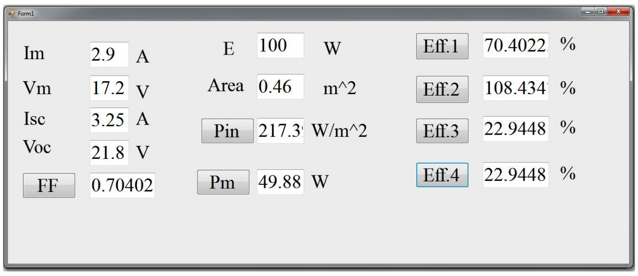

Figure 5. The value of the efficiency.

>>

```
  private void button1_Click(object
sender, EventArgs e)
        {
            double Voc, Isc, FF, Pmax;

            Voc =
Convert.ToDouble(textBox1.Text);
            Isc =
Convert.ToDouble(textBox2.Text);
            FF =
Convert.ToDouble(textBox3.Text);
```

```
            Pmax = Voc * Isc * FF;

            textBox4.Text =
Pmax.ToString();

        }
```

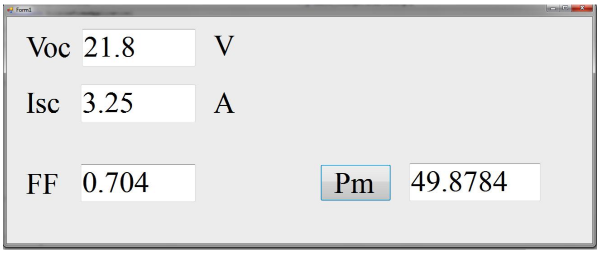

Figure 6. The value of maximum power density.

>>

```
  double T, k, t, q, IL, Io, Io1, p3, p1,
p2, ki, q1;


        private
void    button2_Click(object    sender,
EventArgs e)
        {
            double k1;
            k =
Convert.ToDouble(textBox2.Text);
            T =
Convert.ToDouble(textBox1.Text);
            q =
Convert.ToDouble(textBox4.Text);
            IL =
Convert.ToDouble(textBox5.Text);
            Io =
Convert.ToDouble(textBox6.Text);

            k1 =
Convert.ToDouble(textBox9.Text);
            q1 =
Convert.ToDouble(textBox11.Text);
            Io1 =
Convert.ToDouble(textBox10.Text);

            p1 = k * Math.Pow(10, k1);
            p2 = q * Math.Pow(10, q1);
            p3 = Io * Math.Pow(10, Io1);


            textBox8.Text = (((p1 * T) /
p2) * Math.Log10((IL / p3))).ToString();
        }

        private void Form1_Load(object
sender, EventArgs e)
        {

        }
```

```
        private
void    button3_Click(object    sender,
EventArgs e)
        {
            textBox1.Text=(273.15      +
Convert.ToDouble(textBox3.Text)).ToStr
ing();
        }
```

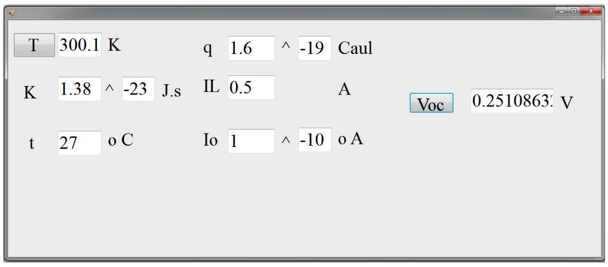

Figure 7. The value of open-circuit voltage.

>>

```
  double      Voc,      Isc,      FF,
Pmax,Pold,A,Pnew,eff;

        private
void    button1_Click(object    sender,
EventArgs e)
        {
            Voc =
Convert.ToDouble(textBox1.Text);
            Isc =
Convert.ToDouble(textBox2.Text);
            FF =
Convert.ToDouble(textBox3.Text);

            Pmax = Voc * Isc * FF;

            textBox4.Text =
Pmax.ToString();
            textBox5.Text =
textBox4.Text;
        }

        private
void    button2_Click(object    sender,
EventArgs e)
        {
            Pold =
Convert.ToDouble(textBox5.Text);
            A =
Convert.ToDouble(textBox6.Text);

            Pnew = Pold * A;

            textBox7.Text =
Pnew.ToString();


        }

        private
void    button3_Click(object    sender,
EventArgs e)
        {
```

```
            eff = (Pmax / Pnew) * 100;
                   textBox8.Text          =
eff.ToString();


        }
```

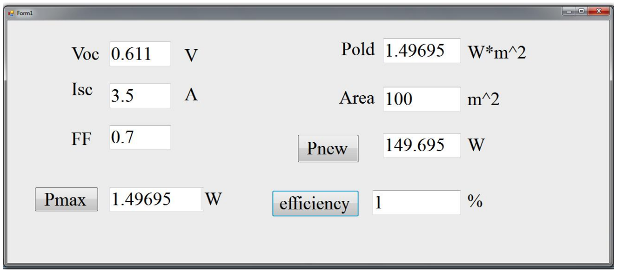

Figure 8. The value of maximum power and efficiency.

\>>>

```
  private   void   button1_Click(object
sender, EventArgs e)
         {
            double Imax, Vmax, Pin, eff;

            Imax                          =
Convert.ToDouble(textBox1.Text);
            Vmax                          =
Convert.ToDouble(textBox2.Text);
            Pin                           =
Convert.ToDouble(textBox3.Text);

            eff = (((Imax * Vmax) / Pin)
* 100);

            textBox4.Text                 =
eff.ToString();
         }
```

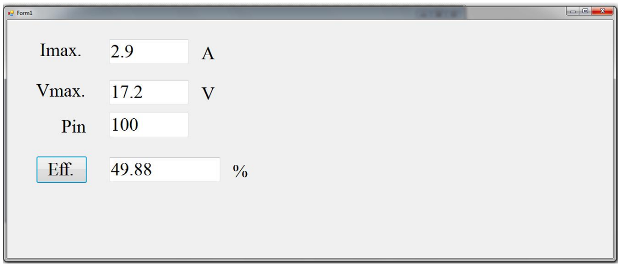

Figure 9. The value of the efficiency.

\>>>

```
  private   void   button1_Click(object
sender, EventArgs e)
         {
            double k, t, q, NA, dn, ni,
Voc, k1, q1, NA1, dn1, ni1, p1, p2, p3,
p4, p5;

            k                             =
Convert.ToDouble(textBox1.Text);
            t                             =
Convert.ToDouble(textBox2.Text);
            q                             =
Convert.ToDouble(textBox3.Text);
            NA                            =
Convert.ToDouble(textBox4.Text);
            dn                            =
Convert.ToDouble(textBox5.Text);
            ni                            =
Convert.ToDouble(textBox6.Text);

            k1                            =
Convert.ToDouble(textBox8.Text);
            q1                            =
Convert.ToDouble(textBox9.Text);
            NA1                           =
Convert.ToDouble(textBox10.Text);
            dn1                           =
Convert.ToDouble(textBox11.Text);
            ni1                           =
Convert.ToDouble(textBox12.Text);

            p1 = k * Math.Pow(10, k1);
            p2 = q * Math.Pow(10, q1);
            p3 = NA * Math.Pow(10, NA1);
            p4 = dn * Math.Pow(10, dn1);
            p5 = ni * Math.Pow(10, ni1);

            Voc  =  ((p1  *  t)  /  p2)  *
Math.Log(((p3 + p4) * p4) / (p5 * p5));

            textBox7.Text                 =
Voc.ToString();
         }
```

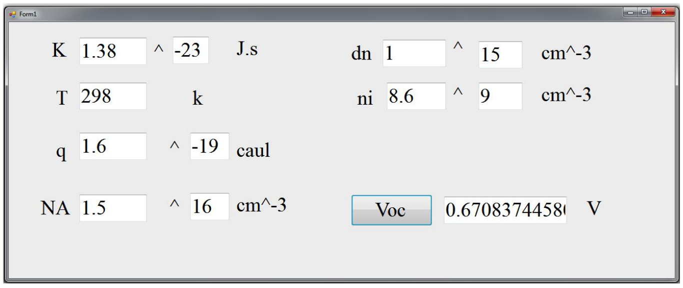

Figure 10. The value of open-circuit voltage.

\>>>

```
  private   void   button1_Click(object
sender, EventArgs e)
         {
            double Voc, FF;
            Voc                           =
Convert.ToDouble(textBox1.Text);

            FF = (Voc - (Math.Log((Voc +
0.72)))) / (Voc + 1);

            textBox2.Text                 =
FF.ToString();
         }
```

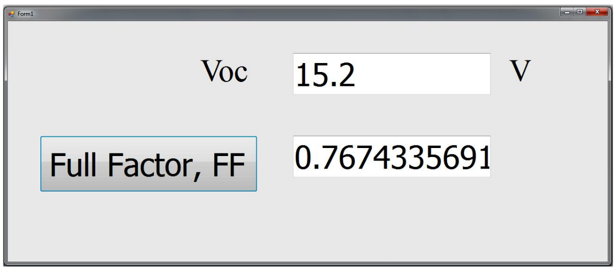

Figure 11. The value of fill factor.

>>>

8. Conclusion

The solar cell parameters is presented by Microsoft Visual Studio C # program is utilized to investigate the η of the solar cell. Elsie sharp came to eliminate the problems of Java and C++ languages, Faqamt cancel macro templates and multiple inheritance this caused confusion with the developers of C++ as well as the emergence of problems. In this research, the programs have been written in C sharp to design some of the equations for the calculation parameters basic task for solar cells, so that given any value for the input parameters produce direct value output of the parameters essential for any solar cell. An example of this if we want to calculate the efficiency of silicon solar cell, give input values is beyond capacity and falling evaluates efficiency value directly without the need to account through personal calculator and therefore; this method facilitates the process of calculating the parameters for any given input values.

References

1. S. S. Kulkarni, C.Y. Thean, A.W. Kong, A novel PC based solar electric panel simulator, IEEE, (2003). X. Gao, Y. Cui, J. Hu, G. Xu, Z. Wang, J. Qu, H. Wang, Parameter extraction of solar cell models using improved shuffled complex evolution algorithm, Energy Conversion and Management, 157 (2018) 460-479.
2. A. Sarwar, M. Hasan, A. Q. Ansari, Five parameter Modeling and Simulation of Solar PV Cell, IEEE, (2015).
3. J. F. Campos, L. F. Ruiz, J. L. Mata-Machuca, System for the Electrical Characterization of Solar Cells Based on One Data Acquisition Board and C#, IEEE International Autumn Meeting on Power, Electronics and Computing, (2017).
4. Omer A. Sultan, Khaleel I. Hassoon, Mohammed S. Rasheed, Deterioration of Silicon Solar Cell Parameter with Ambient Temperature, Al-Mustansiriyah Journal of Science, 14 (1) (2003) 25-31.
5. F. Sh. Tahir, M. S. Rasheed, I. A. Hameed, Analysis the Performance of Silicon Solar Cell Parameters with the Ambient Temperature using Fuzzy Logic, Journal of the College of Basic Education 18 (75) (2012) 173-183.
6. F. Sh. Tahir, M. S. Rasheed, Decline in the Performance of Silicon Solar Cell Parameters with the Ambient Temperature in Baghdad, Journal of the College of Basic Education, 18 (75) (2012) 95-111.
7. W. Saidi, N. Hfaidh, M. Rasheed, M. Girtan, A. Megriche, M. E. Maaoui, Effect of B_2O_3 addition on optical and structural properties of TiO_2 as a new blocking layer for multiple dye sensitive solar cell application (DSSC), RSC Advances, 6 (73) (2016) 68819-68826.
8. A. AUKŠTOLIS, M. Girtan, G. A. Mousdis, R. Mallet, M. Socol, M. Rasheed, A. Stanculescu, Measurement of charge carrier mobility in perovskite nanowire films by photo-CELIV method, Proceedings of the Romanian Academy Series a-Mathematics Physics Technical Sciences Information Science, 18 (1) (2017) 34-41.
9. M. Rasheed, R. Barillé, Room temperature deposition of ZnO and Al: ZnO ultrathin films on glass and PET substrates by DC sputtering technique, Optical and Quantum Electronics, 49 (5) (2017) 1-14.
10. M. Rasheed, R. Barillé, Optical constants of DC sputtering derived ITO, TiO_2 and TiO_2: Nb thin films characterized by spectrophotometry and spectroscopic ellipsometry for optoelectronic devices, Journal of Non-Crystalline Solids, 476 (2017) 1-14.
11. M. Rasheed, R. Barillé, Comparison the optical properties for Bi_2O_3 and NiO ultrathin films deposited on different substrates by DC sputtering technique for transparent electronics, Journal of Alloys and Compounds, 728 (2017) 1186-1198.
12. T. Saidani, M. Zaabat, M. S. Aida, R. Barille, M. Rasheed, Y. Almohamed, Influence of precursor source on sol–gel deposited ZnO thin films properties, Journal of Materials Science: Materials in Electronics, 28 (13) (2017) 9252-9257.
13. K. Guergouria A. Boumezoued, R. Barille, D. Rechemc, M. Rasheed M. Zaabata, ZnO nanopowders doped with bismuth oxide, from synthesis to electrical application, Journal of Alloys and Compounds, 791 (2019) 550-558.
14. F. Dkhilalli, S. Megdiche, K. Guidara, M. Rasheed, R. Barillé, M. Megdiche, AC conductivity evolution in bulk and grain boundary response of sodium tungstate Na_2WO_4, Ionics, 24 (1) (2018) 169-180.
15. F. Dkhilalli, S. M. Borchani, M. Rasheed, R. Barille, K. Guidara, M. Megdiche, Structural, dielectric, and optical properties of the zinc tungstate $ZnWO_4$ compound, Journal of Materials Science: Materials in Electronics, 29 (8) (2018) 6297-6307.
16. F. Dkhilalli, S. M. Borchani, M. Rasheed, R. Barille, S. Shihab, K. Guidara, M. Megdiche, Characterizations and morphology of sodium tungstate particles, Royal Society open science, 5 (8) (2018) 1-12.

17. M. Enneffati, B. Louati, K. Guidara, M. Rasheed, R. Barillé, Crystal structure characterization and AC electrical conduction behavior of sodium cadmium orthophosphate, Journal of Materials Science: Materials in Electronics, 29 (1) (2018) 171-179.
18. D. Bouras, A. Mecif, R. Barillé, A. Harabi, M. Rasheed, A. Mahdjoub, M. Zaabat, Cu: ZnO deposited on porous ceramic substrates by a simple thermal method for photocatalytic application, Ceramics International, 44 (17) (2018) 21546-21555.
19. M. RASHEED, M. A. Sarhan, Characteristics of Solar Cell Outdoor Measurements Using Fuzzy Logic Method, Insight-Mathematics 1 (1) 2019 1-8.
20. M. RASHEED, Investigation of Solar Cell Factors using Fuzzy Set Technique, Insight-Electronic 1 (1) 2019 11-17.
21. M. RASHEED, M. A. Sarhan, Measuring the Solar Cell Parameters Using Fuzzy Set Technique, Insight-Electronic 1 (1) 2019 1-9.
22. M. RASHEED, Linear Programming for Solving Solar Cell Parameters, Insight-Electronic 1 (1) 2019 10-16.
23. M. S. Rasheed, Approximate Solutions of Barker Equation in Parabolic Orbits, Engineering & Technology Journal, 28 (3) (2010) 492-499.
24. M. S. Rasheed, An Improved Algorithm For The Solution of Kepler‘s Equation For An Elliptical Orbit, Engineering & Technology Journal, 28 (7) 2010 1316-1320.
25. M. S. Rasheed, Acceleration of Predictor Corrector Halley Method in Astrophysics Application, International Journal of Emerging Technologies in Computational and Applied Sciences, 1 (2) 2012 91-94.
26. M. S. Rasheed, Fast Procedure for Solving Two-Body Problem in Celestial Mechanic, International Journal of Engineering, Business and Enterprise Applications, 1 (2) 2012 60-63.
27. M. S. Rasheed, Solve the Position to Time Equation for an Object Travelling on a Parabolic Orbit in Celestial Mechanics, DIYALA JOURNAL FOR PURE SCIENCES, 9 (4) 2013 31-38.
28. M. S. Rasheed, Comparison of Starting Values for Implicit Iterative Solutions to Hyperbolic Orbits Equation, International Journal of Software and Web Sciences (IJSWS), 1 (2) 2013 65-71.
29. M. S Rasheed, On Solving Hyperbolic Trajectory Using New Predictor-Corrector Quadrature Algorithms, Baghdad Science Journal, 11 (1) 2014 186-192.
30. M. S. Rasheed, Modification of Three Order Methods for Solving Satellite Orbital Equation in Elliptical Motion, Journal of university of Anbar for Pure science, 2019 in press.
31. E. Kadri, M. Krichen, R. Mohammed, A. Zouari, K. Khirouni, Electrical transport mechanisms in amorphous silicon/crystalline silicon germanium heterojunction solar cell: impact of passivation layer in conversion efficiency, Optical and Quantum Electronics, 48 (12) (2016) 1-15.
32. E. Kadri, O. Messaoudi, M. Krichen, K. Dhahri, M. Rasheed, E. Dhahri, A. Zouari, K. Khirouni, R. Barillé, Optical and electrical properties of SiGe/Si solar cell heterostructures: Ellipsometric study, Journal of Alloys and Compounds, 721 (2017) 779-783.
33. E. Kadri, K. Dhahri, A. Zaafouri, M. Krichen, M. Rasheed, K. Khirouni, R. Barillé, Ac conductivity and dielectric behavior of a−Si: H/c−Si1−yGey/p−Si thin films synthesized by molecular beam epitaxial method, Journal of Alloys and Compounds, 705 (2017) 708-713.
34. M. A. Green, Solar cell fill factors: General graph and empirical expressions, Solid-State Electronics, 24 (1981) 788 - 789.
35. A. Jain, Exact analytical solutions of the parameters of real solar cells using Lambert W-function, Solar Energy Materials and Solar Cells, 81 (2) (2004) 269 – 277.