Университет ИТМО
Физико-технический мегафакультет
Физический факультет

Группа М3201 К работе допущен_____

Студенты Ткачук С.А. и Чуб Д.О. Работа выполнена_____

Преподаватель Шоев В.И. Отчет принят_____

Отчет по моделированию № 2.3

Частица в конденсаторе

1. Теоретическая часть

В данной работе исследуется движение электрона в конденсаторе под действием электрического поля, создаваемого разностью потенциалов между пластинами конденсатора. Используются уравнения движения в электрическом поле. Используется модель движения заряженной частицы в конденсаторе. Решение основано на численном методе, а именно, методе бинарного поиска, для определения минимальной разности потенциалов между пластинами конденсатора, при которой электрон не покидает конденсатор.

2. Используемые формулы:

$a_y = \frac{qU}{md\ln\left(\frac{R}{r}\right)}$ (1) - ускорение по оси y, зависящее от заряда частицы, ее массы, расстоянии d частицы от оси симметрии цилиндров, радиусов цилиндров и разности потенциалов обкладок

Формула (1) получена из 3-х формул:

$a = \frac{eE}{m}$ - формула ускорения электрона в электрическом поле

$E = \frac{q}{2\pi\varepsilon\varepsilon_0 ld}$ – формула напряженности электрического поля между обкладками цилиндрического конденсатора, на расстоянии d от оси цилиндров

$U = \frac{q}{2\pi\varepsilon\varepsilon_0 l} ln\frac{R}{r}$ – формула разности потенциалов между обкладками цилиндрического конденсатора

Выражая E через U получаем, что $E = \frac{U}{d\, ln\left(\frac{R}{r}\right)}$

$v_y = v_0 + a_y t$ (2) – изменение скорости по оси y

$y = y_0 + v_y t + \frac{a_y t^2}{2}$ (3) – изменение координаты y

3. Численный алгоритм для решения уравнений

Используется бинарный поиск для поиска минимальной разности потенциалов между обкладками конденсатора

4. Программный код на языке python:

```
import matplotlib.pyplot
import math
import numpy

r = 0.075
R = 0.16
v_x = 2 * (10 ** 6)
L = 0.24
q = -1.6021766208 * (10 ** -19)
m = 9.1093837015 * (10 ** -31)


def get_next_v_y(v_y_pred, a_y):
    return v_y_pred + a_y * delta_t


def get_next_y(y_pred, v_y, a_y):
    return y_pred + v_y * delta_t + a_y * delta_t * delta_t / 2


def get_a_y(u, y):
    return (u * q) / (m * (y + r) * math.log(R / r))


total_flight_time = L / v_x

delta_t = 10 ** -9
u_left, u_right = 0, 1000
u_mid = 0
```

```
while u_right - u_left > (10 ** -15):

    u_mid = (u_right + u_left) / 2

    y = (R - r) / 2
    v_y = 0
    a_y = get_a_y(u_mid, y)

    is_out = False

    for i in range(1, int(total_flight_time * (10 ** 9))):
        y = get_next_y(y, v_y, a_y)
        v_y = get_next_v_y(v_y, a_y)
        a_y = get_a_y(u_mid, y)
        if y < 0 or y > R - r:
            is_out = True
            break

    if is_out:
        u_right = u_mid
    else:
        u_left = u_mid

y = (R - r) / 2
v_y = 0

y_values = []
v_y_values = []
a_values = []
t_values = []
x_values = numpy.linspace(0, L, int(total_flight_time * (10 ** 9)))

for i in range(0, int(total_flight_time * (10 ** 9))):
    a_y = get_a_y(u_left, y)
    y = get_next_y(y, v_y, a_y)
    v_y = get_next_v_y(v_y, a_y)

    y_values.append(y)
    v_y_values.append(v_y)
    a_values.append(a_y)
    t_values.append(i * (10 ** -9))

figure = matplotlib.pyplot.figure(figsize=(10, 10))
matplotlib.pyplot.subplots_adjust(wspace=0.6, hspace=0.6)

subplot1 = figure.add_subplot(2, 2, 1)
subplot1.set_title("y(x)")
subplot1.set_xlabel("x, m", color='black')
subplot1.set_ylabel("y, m", color='black')
subplot1.plot(x_values, y_values, color='green')

subplot2 = figure.add_subplot(2, 2, 2)
subplot2.set_title("v_y(t)")
subplot2.set_xlabel("t, s", color='black')
subplot2.set_ylabel("v_y, m/s", color='black')
subplot2.plot(t_values, v_y_values, color='green')

subplot3 = figure.add_subplot(2, 2, 3)
subplot3.set_title("a_y(t)")
subplot3.set_xlabel("t, s", color='black')
subplot3.set_ylabel("a, m/s^2", color='black')
subplot3.plot(t_values, a_values, color='green')
```

```
subplot4 = figure.add_subplot(2, 2, 4)
subplot4.set_title("y(t)")
subplot4.set_xlabel("t, s", color='black')
subplot4.set_ylabel("y, m", color='black')
subplot4.plot(t_values, y_values, color='green')

matplotlib.pyplot.show()

print("Potential difference: {:.7f}".format(u_right))
v = math.sqrt(v_x ** 2 + v_y ** 2)
print("Speed: {:.7f}".format(v))
print("Time: {:.7f}".format(total_flight_time))
```

5. Вывод

В данной работе проведено численное моделирование движения электрона в цилиндрическом конденсаторе под воздействием электрического поля. Используется метод бинарного поиска для определения минимальной разности потенциалов между пластинами конденсатора, при которой электрон не покидает его пределов. Исследование позволяет оценить влияние электрического поля на движение электрона и определить условия его задержания в конденсаторе.