МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**РГЗ**

по дисциплине: Системное моделирование

тема: Математическое моделирование работы электронно-механической измерительной системы

Выполнил: ст. группы ПВ-223

Игнатьев Артур Олегович

Проверил:

Полунин Александр Иванович

Белгород 2024 г.

**Цель работы:**

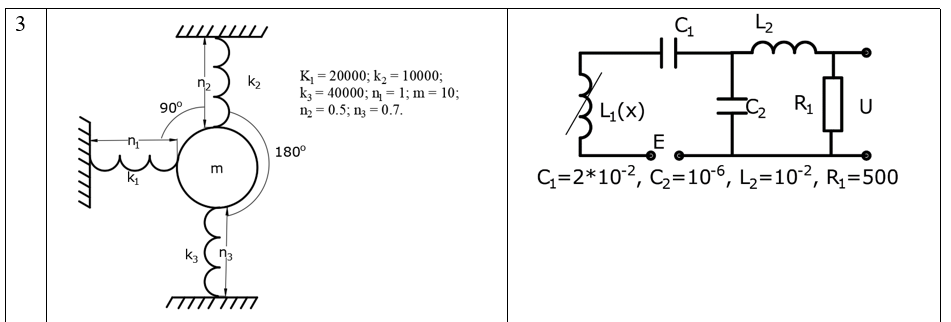
1. Формулировка задачи.

2. Математическая постановка задачи: вывод необходимых формул, выбор и запись расчётных методов и алгоритмов.

3. Блок-схема программы.

4. Результаты расчётов — графики.

**Вариант 3**

****

**Выполнение работы:**

**Для механической системы:**

Уравнение Лагранжа второго рода имеет вид:

Для данной системы система будет выглядеть как

Найдём кинетическую энергию:

Найдём потенциальную энергию

Далее найдём необходимые производные для составления уравнения:

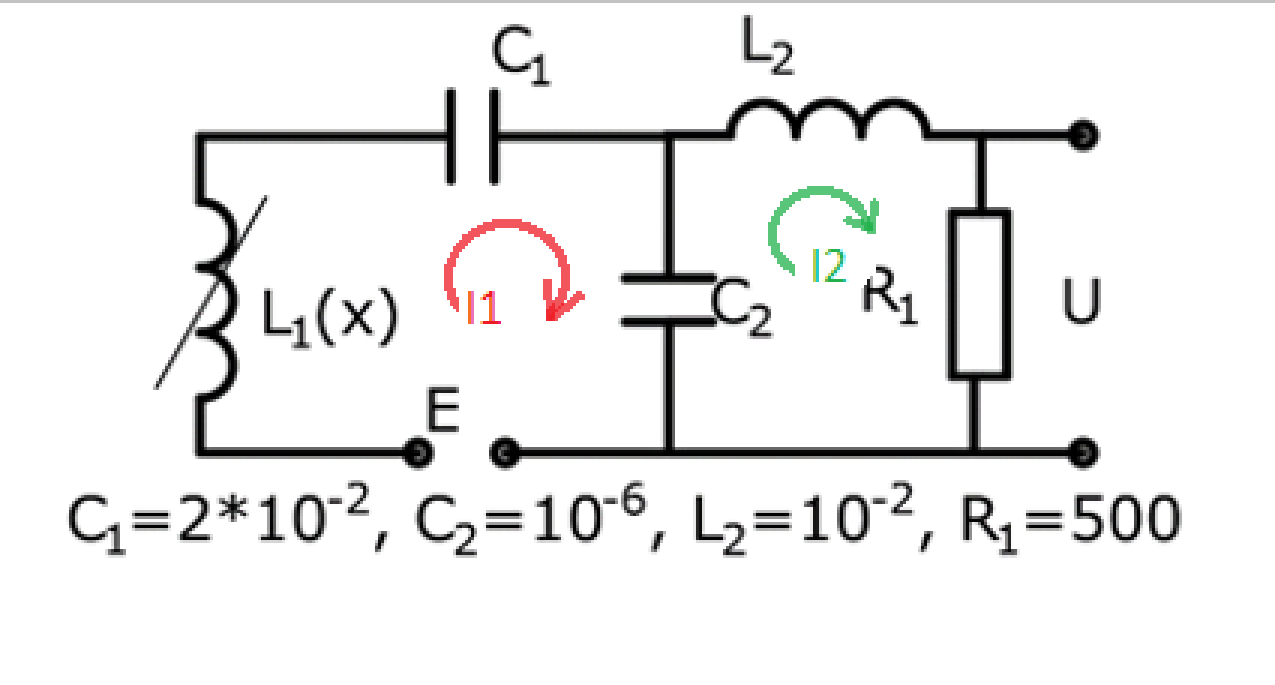
Для кинетической энергии:

Аналогично,

Для потенциальной энергии:

Получим систему:

Для электрической системы:



Составим Уравнения согласно второму закону Кирхгофа:

Подставим:

Код программы:

import math

from scipy.integrate import odeint

import numpy as np

import matplotlib.pyplot as plt

# MS

k1 = 20000

k2 = 10000

k3 = 40000

n1 = 1

n2 = 0.5

n3 = 0.7

m = 10

g = 9.81

# ES

C1 = 2e-2

C2 = 1e-6

L2 = 1e-2

R1 = 500

E = 1

mu = 1

L0 = 5e-2

coef = 0.1

def models(solve, time):

    global k1

    global k2

    global k3

    global m

    global n1

    global n2

    global n3

    global g

    V1, y, V2, x = solve[:4]

    V2\_ = -(k1 \* x) / m

    x\_ = V2

    V1\_ = -(m\*g + k2 \* y + k3 \* y) / m

    y\_ = V1

    global C1

    global C2

    global L2

    global R1

    global E

    global mu

    global L0

    global coef

    I1, I2, Q1, Q2 = solve[4:]

    I0 = E / R1

    mu\_ = mu

    if (x >= 0):

        mu\_ = -mu

    def L1(x):

        if (abs(x) <= coef):

            return L0 - mu \* abs(x)

        else:

            return R1 \* 2

    I1\_ = (E - Q1 / C1 - Q1 / C2 + Q2 / C2 - mu\_ \* (V1\_ + V2\_) \* (I0 + I1)) / L1(x)

    I2\_ = (Q1 / C2 - Q2 / C2 - R1 \* (I0 + I2)) / L2

    Q1\_ = I0 + I1

    Q2\_ = I0 + I2

    return [V1\_, y\_ , V2\_, x\_, I1\_, I2\_, Q1\_, Q2\_]

if \_\_name\_\_ == '\_\_main\_\_':

    time\_ = 1.75

    t = np.linspace(0, time\_, 1000)

    solve0 = [0, 0.03, 0, 0.03, 0, 0, 0, 0]  # Vy, y, Vx, x, I1, I2, Q1, Q2

    solve = odeint(models, solve0, t)

    Vy = solve[:, 0]

    y = solve[:, 1]

    Vx = solve[:, 2]

    x = solve[:, 3]

    I1 = solve[:, 4]

    I2 = solve[:, 5]

    Q1 = solve[:, 6]

    Q2 = solve[:, 7]

    U\_in\_R = np.array([I2[\_] \* R1 for \_ in range(len(I2))])

    fig, axes = plt.subplots(2,1)

    line\_width = 2

    color1 = '#F27E28'  # orange

    color2 = '#53D11C'  # green

    color3 = '#9F28F2'  # purple

    f1 = axes[0]

    f1.plot(t, y, label='y(time)', color=color1, linewidth=line\_width)

    f1.plot(t, x, label='x(time)', color=color2, linewidth=line\_width)

    # f1.plot(t, I2, label='I2(time)', color=color3, linewidth=line\_width)

    f1.set\_xlabel('time')

    f1.set\_ylabel('м')

    f1.set\_title('График смещения х и у', fontsize=8)

    f1.legend()

    f1.grid()

    f2 = axes[1]

    f2.plot(t, U\_in\_R, label='U(time)', color=color3, linewidth=line\_width)

    f2.set\_xlabel('time')

    f2.set\_ylabel('В')

    f2.legend()

    f2.grid()

    plt.show()

Результат работы программы:

Изображение выглядит как текст, График, диаграмма, линия

Автоматически созданное описание

Изображение выглядит как текст, График, диаграмма, линия

Автоматически созданное описание