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

**Лабораторная работа №6**

по дисциплине: Системное моделирование

тема: Переходные процессы в электрических цепях

Выполнил: ст. группы ПВ-223

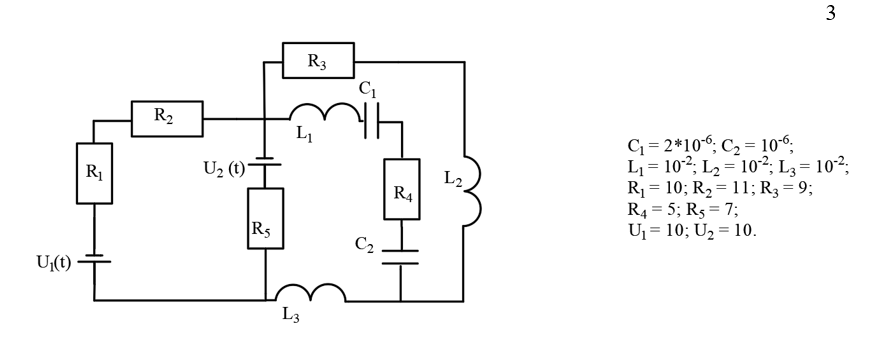
Игнатьев Артур Олегович

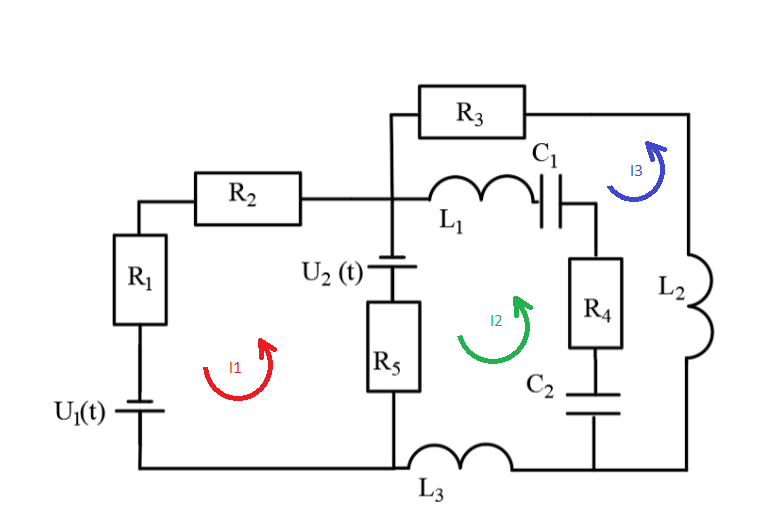
Проверил:

Полунин Александр Иванович

Белгород 2024 г.

**Вариант 3**





На основании второго закона Кирхгофа получим:

Перенесем производные в левую часть и подставим:

Тогда имеем систему уравнений:

Решим получившуюся систему методом обратной матрицы и получим уравнения:

Код программы:

import numpy as np  
from scipy.integrate import ode  
import matplotlib.pyplot as plt  
  
  
# Функция, задающая систему дифференциальных уравнений  
def func(t, y):  
 # Константы (емкости, индуктивности и сопротивления)  
 C1 = 2E-6 # Емкость C1 в Фарадах  
 C2 = 1E-6 # Емкость C2 в Фарадах  
 L1, L2, L3 = 1E-2, 1E-2, 1E-2 # Индуктивности L1, L2, L3 в Генри  
 R1, R2, R3, R4, R5 = 10, 11, 9, 5, 7 # Сопротивления R1, R2, R3, R4, R5 в Омах  
 U1 = 10 # Входное напряжение 1 в Вольтах  
 U2 = 10 # Входное напряжение 2 в Вольтах  
  
 # Суммы индуктивностей  
 sum\_L3\_L2 = L1 + L2  
 sum\_L1\_L3 = L1 + L3  
  
 # Переменные состояния (заряды и токи)  
 Q1, Q2, Q3, I1, I2, I3 = y  
  
 # Определение производных  
 dI1dt = 0 # Производная тока I1 (заданная как 0)  
 dI2dt = ((sum\_L3\_L2 \* U2 - L3 \* U2) +  
 (sum\_L3\_L2 \* (-R5 \* (-I2 - I3 + I1)) - L3 \* (-I3 \* R5)) +  
 (sum\_L3\_L2 \* (-Q2 / C2) - L3 \* (-I2 \* R5)) +  
 (sum\_L3\_L2 \* (-I2 \* R4) - L3 \* (I1 \* R5)) +  
 (sum\_L3\_L2 \* (-Q2 / C1) - L3 \* (-I3 \* R3))) / (L3 \* L2 + L1 \* L3 + L1 \* L2)  
 dI3dt = ((-L3 \* U2 + sum\_L1\_L3 \* U2) +  
 (-L3 \* (-R5 \* (-I2 - I3 + I1)) + sum\_L1\_L3 \* (-I3 \* R5)) +  
 (-L3 \* (-Q2 / C2) + sum\_L1\_L3 \* (-I2 \* R5)) +  
 (-L3 \* (-I2 \* R4) + sum\_L1\_L3 \* (I1 \* R5)) +  
 (-L3 \* (-Q2 / C1) + sum\_L1\_L3 \* (-I3 \* R3))) / (L3 \* L2 + L1 \* L3 + L1 \* L2)  
 dQ1dt = (U1 - U2 + I2 \* R5 + I3 \* R5) / (R1 + R2 + R5)  
 dQ2dt = I2  
 dQ3dt = I3  
  
 return [dQ1dt, dQ2dt, dQ3dt, dI1dt, dI2dt, dI3dt]  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 # Начальные условия для переменных состояния (заряды и токи)  
 y0 = [0, 0, 0, -1, -2, -5] # Q1, Q2, Q3, I1, I2, I3  
  
 # Временные параметры: начальное и конечное время, шаг интегрирования  
 t0, t1 = 0, 0.04 # Начальное и конечное время интегрирования в секундах  
 dt = 0.000001 # Шаг интегрирования в секундах  
  
 # Создание объекта интегратора  
 r = ode(func)  
 r.set\_integrator('dopri5')  
 r.set\_initial\_value(y0, t0)  
  
 # Массивы для хранения времени и результатов интегрирования  
 t = [t0]  
 y = [y0]  
  
 # Численное интегрирование системы уравнений  
 while r.successful() and r.t < t1:  
 ti = r.t + dt  
 yi = r.integrate(ti)  
 t.append(ti)  
 y.append(yi)  
  
 # Преобразование списков в массивы numpy для удобства работы  
 t = np.array(t)  
 y = np.array(y)  
  
 # Построение графиков результатов  
 plt.plot(t, y[:, 0], 'r', label='I1(t)')  
 plt.plot(t, y[:, 4], 'g', label='I2(t)')  
 plt.plot(t, y[:, 5], 'b', label='I3(t)')  
 plt.legend(loc='best')  
 plt.xlabel('t')  
 plt.grid()  
 plt.show()

Результат работы программы:

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание