**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG ĐIỆN - ĐIỆN TỬ**

A red and white sign with a yellow and red circle and a red and white rectangle

Description automatically generated

**BÀI TẬP ĐIỆN TỬ TƯƠNG TỰ II**

**Phầm mềm tính toán PHTK L, T, Pi**

**Giảng viên hướng dẫn: Thầy Nguyễn Nam Phong**

**Sinh viên thực hiện: Nguyễn Mạnh Dũng**

**Mssv: 20213838**

**Mã lớp: 150111**

Mục Lục

[**PHẦN 1: CƠ SỞ LÝ THUYẾT** 3](#_Toc163144746)

[**1.** **MẠCH PHỐI HỢP TRỞ KHÁNG L** 3](#_Toc163144747)

[1.1 LOWPASS Hi-Low MATCHING NETWORK 3](#_Toc163144748)

[1.2 HIGHPASS Hi-Low MATCHING NETWORK 3](#_Toc163144749)

[1.3. LOWPASS Low-Hi MATCHING NETWORK 3](#_Toc163144750)

[1.4 HIGHPASS Low-Hi MATCHING NETWORK 4](#_Toc163144751)

[**2.** **MẠCH PHỐI HỢP TRỞ KHÁNG PI** 4](#_Toc163144752)

[**3.** **MẠCH PHỐI HỢP TRỞ KHÁNG HÌNH T** 6](#_Toc163144753)

[**PHẦN 2: PHẦN MỀM** 9](#_Toc163144754)

[**1.** **Khung Chương Trình** 9](#_Toc163144755)

[**Import thư viện và khởi tạo các hàm** 9](#_Toc163144756)

[**tạo khung L của Notebook** 11](#_Toc163144757)

[**tạo khung T của Notebook** 12](#_Toc163144758)

[**tạo khung Pi của Notebook** 14](#_Toc163144759)

[**2.** **Chương trình PHTK L** 15](#_Toc163144760)

[**Các hàm để tính toán về PHTK L** 15](#_Toc163144761)

[**3.** **Chương trình PHTK T** 16](#_Toc163144762)

[**Các hàm để tính toán về PHTK T** 16](#_Toc163144763)

[**4.** **Chương trình PHTK Pi** 18](#_Toc163144764)

[**Các hàm để tính toán về PHTK Pi** 18](#_Toc163144765)

[**5.** **Giao diện phần mềm** 19](#_Toc163144766)

[**a)** **PHTK L** 19](#_Toc163144767)

[**b)** **PHTK T** 20](#_Toc163144768)

[**c)** **PHTK Pi** 20](#_Toc163144769)

[**d)** **Github** 20](#_Toc163144770)

**PHẦN 1: CƠ SỞ LÝ THUYẾT**

1. **MẠCH PHỐI HỢP TRỞ KHÁNG L**

1.1 LOWPASS Hi-Low MATCHING NETWORK

A diagram of a circuit

Description automatically generated

1.2 HIGHPASS Hi-Low MATCHING NETWORK

A diagram of a circuit

Description automatically generated

1.3. LOWPASS Low-Hi MATCHING NETWORK

A diagram of a circuit

Description automatically generated



1.4 HIGHPASS Low-Hi MATCHING NETWORK

A diagram of a circuit

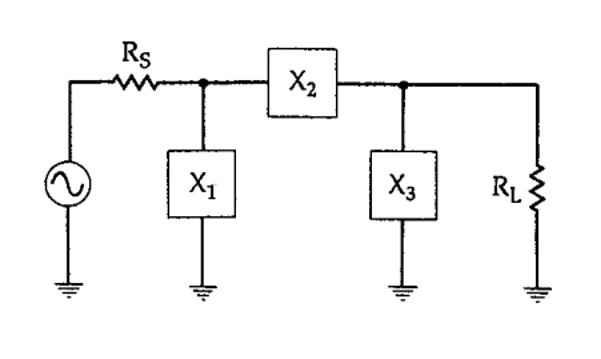
Description automatically generated

1. **MẠCH PHỐI HỢP TRỞ KHÁNG PI**

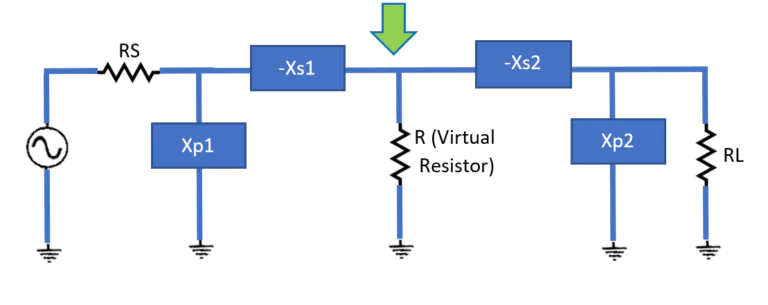
**CÁC LOẠI MẠCH:**

A diagram of a computer network

Description automatically generated

****

*(sơ đồ PI matching network)*

**

- Mạch Pi có thể được lắp với mạch L đối xứng quay lưng với nhau( back-to-back). Ở giữa 2 mạch L mắc nối tiếp có 1 điện trở ảo.

- Trở kháng trung tâm Rcenter (nhìn từ phía điện trở ảo đến Rs và RL có giá trị < min(Rs; RL)

* Dấu âm của -Xs1, -Xs2 có nghĩa là khi Xs1 là cuộn cảm L thì -Xs1 là tụ điện C ( và ngược lại). Tương tự với Xp2 và -Xp2
* Ta có công thức : , , 
* **Ví dụ**: phối hợp trở khánh sử dụng mạng PI với Rs=1000 Ohm, RL= 50 Ohm với tần số 2Mhz, Rcenter=10 Ohm

A diagram of a circuit

Description automatically generated

**Nguồn**: Tải:

1. **MẠCH PHỐI HỢP TRỞ KHÁNG HÌNH T**

**CÁC LOẠI MẠCH:**

**A diagram of a network

Description automatically generated**

A black and white image of a wire

Description automatically generated with medium confidence

* Mạch Pi có thể được lắp với mạch L đối xứng với nhau( front-to-front). Ở giữa 2 mạch L mắc nối tiếp có 1 điện trở ảo.
* Trở kháng trung tâm Rcenter (nhìn từ phía điện trở ảo đến Rs và RL có giá trị >max(Rs; RL)
* Dấu âm của -Xs1, -Xs2 có nghĩa là khi Xs1 là cuộn cảm L thì -Xs1 là tụ điện C ( và ngược lại). Tương tự với Xp2 và -Xp2
* Ta có công thức : , , 
* **Ví dụ**: phối hợp trở khánh sử dụng mạng PI với Rs=1000 Ohm, RL= 50 Ohm với tần số 2Mhz, Rcenter=2000 Ohm

A diagram of electrical wires

Description automatically generated with medium confidence

**Nguồn**: Tải

**PHẦN 2: PHẦN MỀM**

1. **Khung Chương Trình**

* Chương trình sử dụng ngôn ngữ python để lập trình.
* Dùng thư viện Tkinter để tạo giao diện người dùng

**Import thư viện và khởi tạo các hàm**

import tkinter as tk

from tkinter import ttk

import L

import T

import Pi

def clear\_screen(p):

    # Xóa nội dung của các Entry và Text Box

    if(p=="l"):

        text1.delete(1.0, tk.END)

        eline1l.delete(0,tk.END)

        eline2l.delete(0,tk.END)

        eline3l.delete(0,tk.END)

    if(p=="t"):

        text2.delete(1.0, tk.END)

        eline1t.delete(0,tk.END)

        eline2t.delete(0,tk.END)

        eline3t.delete(0,tk.END)

        eline4t.delete(0,tk.END)

    if(p=="p"):

        text3.delete(1.0, tk.END)

        eline1p.delete(0,tk.END)

        eline2p.delete(0,tk.END)

        eline3p.delete(0,tk.END)

        eline4p.delete(0,tk.END)

##########################################################

def solveL(loaimach):

    rll\_value = float(rll.get())

    zinl\_value = float(zinl.get())

    fl\_value = float(fl.get())

    text1.delete(1.0, tk.END)

    if(loaimach==1 and zinl\_value<rll\_value):

        text1.insert(tk.END, "Error")

        return 0

    if(loaimach==2 and zinl\_value<rll\_value):

        text1.insert(tk.END, "Error")

        return 0

    if(loaimach==3 and zinl\_value>rll\_value):

        text1.insert(tk.END, "Error")

        return 0

    if(loaimach==4 and zinl\_value>rll\_value):

        text1.insert(tk.END, "Error")

        return 0

    L\_value=L.solve\_L(zinl\_value,rll\_value,fl\_value,loaimach)

    C\_value=L.solve\_C(zinl\_value,rll\_value,fl\_value,loaimach)

    Q\_value=L.solve\_Q(zinl\_value,rll\_value,fl\_value,loaimach)

    print=f"L={L\_value} (H)\nC={C\_value} (F)\nQ={Q\_value}"

    text1.insert(tk.END, print)

####################################################

def solveT(loaimach):

    #1 highpass, 2 lowpasss

    text2.delete(1.0, tk.END)

    rlt\_value=float(rlt.get())

    zint\_value=float(zint.get())

    ft\_value=float(ft.get())

    rct\_value=float(rct.get())

    if(rct\_value<max(zint\_value,rlt\_value)):

        text2.insert(tk.END,"error")

        return 0

    thongso=T.T\_solve(zint\_value,rct\_value,rlt\_value,ft\_value,loaimach)

    L1\_value=thongso.solve\_L1()

    L2\_value=thongso.solve\_L2()

    C1\_value=thongso.solve\_C1()

    C2\_value=thongso.solve\_C2()

    Q1\_value=thongso.solve\_Q1()

    Q2\_value=thongso.solve\_Q2()

    print=f"L1={L1\_value} (H)\nL2={L2\_value} (H)\nC1={C1\_value} (F)\nC2={C2\_value} (F)\nQ1={Q1\_value}\nQ2={Q2\_value}"

    text2.insert(tk.END, print)

####################################################

def solvePi(loaimach):

    #1 highpass, 2 lowpasss

    text3.delete(1.0, tk.END)

    rlp\_value=float(rlp.get())

    zinp\_value=float(zinp.get())

    fp\_value=float(fp.get())

    rcp\_value=float(rcp.get())

    if(rcp\_value>min(zinp\_value,rlp\_value)):

        text3.insert(tk.END,"error")

        return 0

    thongso=Pi.Pi\_solve(zinp\_value,rcp\_value,rlp\_value,fp\_value,loaimach)

    L1\_value=thongso.solve\_L1()

    L2\_value=thongso.solve\_L2()

    C1\_value=thongso.solve\_C1()

    C2\_value=thongso.solve\_C2()

    Q1\_value=thongso.solve\_Q1()

    Q2\_value=thongso.solve\_Q2()

    print=f"L1={L1\_value} (H)\nL2={L2\_value} (H)\nC1={C1\_value} (F)\nC2={C2\_value} (F)\nQ1={Q1\_value}\nQ2={Q2\_value}"

    text3.insert(tk.END, print)

# root window

root = tk.Tk()

root.geometry('340x300')

root.title('PHTK DTTT2 20213838')

root.resizable(0,0)

# create a notebook

notebook = ttk.Notebook(root)

notebook.grid(row=0)

## **tạo khung L của Notebook**

#1-----------------------------------------------------------------

frame1 = ttk.Frame(notebook, width=340, height=280)

frame1.columnconfigure(2, weight=1)

text1=tk.Text(frame1, height=8, width=40)

text1.grid(columnspan = 3,sticky="w", padx=5)

rll=tk.StringVar()

zinl=tk.StringVar()

ql=tk.StringVar()

fl=tk.StringVar()

line1l = ttk.Label(frame1, text="Zin:")

eline1l = ttk.Entry(frame1, textvariable=zinl)

line1l.grid(row=1, column=0, sticky='w')

eline1l.grid(row=1, column=1, sticky='w')

line2l = ttk.Label(frame1, text="RL:")

eline2l = ttk.Entry(frame1, textvariable=rll)

line2l.grid(row=2, column=0, sticky='w')

eline2l.grid(row=2, column=1, sticky='w')

line3l = ttk.Label(frame1, text="f:")

eline3l = ttk.Entry(frame1, textvariable=fl)

line3l.grid(row=3, column=0, sticky='w')

eline3l.grid(row=3, column=1, sticky='w')

#line4 = ttk.Label(frame1, text="Q:")

#eline4 = ttk.Entry(frame1, textvariable=ql)

#line4.grid(row=4, column=0, sticky='w')

#eline4.grid(row=4, column=1, sticky='w')

button=tk.Button(frame1,text="Pass DC-RS>RL", command=lambda:solveL(1), width=15)

button.grid(column=2,row=1,sticky="w")

button=tk.Button(frame1,text="Block DC-RS>RL",command=lambda:solveL(2),width=15)

button.grid(column=2,row=2,sticky="w")

button=tk.Button(frame1,text="Pass DC-RS<RL",command=lambda:solveL(3),width=15)

button.grid(column=2,row=3,sticky="w")

button=tk.Button(frame1,text="Block DC-RS<RL",command=lambda:solveL(4),width=15)

button.grid(column=2,row=4,sticky="w")

button=tk.Button(frame1,text="Clear", command=lambda:clear\_screen("l"),width=10)

button.grid(column=1,row=4,sticky="w")

for i in frame1.winfo\_children():

    i.grid(pady=4, padx=6)

## **tạo khung T của Notebook**

#2-----------------------------------------------------------------

frame2 = ttk.Frame(notebook, width=340, height=280)

frame2.columnconfigure(2, weight=1)

text2=tk.Text(frame2, height=8, width=40)

text2.grid(columnspan = 3,sticky="w", padx=5)

rlt=tk.StringVar()

zint=tk.StringVar()

ft=tk.StringVar()

rct=tk.StringVar()

line1t = ttk.Label(frame2, text="Zin:")

eline1t = ttk.Entry(frame2, textvariable=zint)

line1t.grid(row=1, column=0, sticky='w')

eline1t.grid(row=1, column=1, sticky='w')

line2t = ttk.Label(frame2, text="RL:")

eline2t = ttk.Entry(frame2, textvariable=rlt)

line2t.grid(row=2, column=0, sticky='w')

eline2t.grid(row=2, column=1, sticky='w')

line3t = ttk.Label(frame2, text="f:")

eline3t = ttk.Entry(frame2, textvariable=ft)

line3t.grid(row=3, column=0, sticky='w')

eline3t.grid(row=3, column=1, sticky='w')

line4t = ttk.Label(frame2, text="Rcenter:")

eline4t = ttk.Entry(frame2, textvariable=rct)

line4t.grid(row=4, column=0, sticky='w')

eline4t.grid(row=4, column=1, sticky='w')

button=tk.Button(frame2,text="High Pass", command=lambda:solveT(1), width=15)

button.grid(column=2,row=1,sticky="w")

button=tk.Button(frame2,text="Low Pass",command=lambda:solveT(2),width=15)

button.grid(column=2,row=2,sticky="w")

button=tk.Button(frame2,text="Clear", command=lambda:clear\_screen("t"),width=15)

button.grid(column=2,row=3,sticky="w" )

for i in frame2.winfo\_children():

    i.grid(pady=4, padx=6)

## **tạo khung Pi của Notebook**

#3-------------------------------------------------------

frame3 = ttk.Frame(notebook, width=340, height=280)

frame3.columnconfigure(2, weight=1)

text3=tk.Text(frame3, height=8, width=40)

text3.grid(columnspan = 3,sticky="w", padx=5)

rlp=tk.StringVar()

zinp=tk.StringVar()

rcp=tk.StringVar()

fp=tk.StringVar()

line1p = ttk.Label(frame3, text="Zin:")

eline1p = ttk.Entry(frame3, textvariable=zinp)

line1p.grid(row=1, column=0, sticky='w')

eline1p.grid(row=1, column=1, sticky='w')

line2p = ttk.Label(frame3, text="RL:")

eline2p = ttk.Entry(frame3, textvariable=rlp)

line2p.grid(row=2, column=0, sticky='w')

eline2p.grid(row=2, column=1, sticky='w')

line3p = ttk.Label(frame3, text="f:")

eline3p = ttk.Entry(frame3, textvariable=fp)

line3p.grid(row=3, column=0, sticky='w')

eline3p.grid(row=3, column=1, sticky='w')

line4p = ttk.Label(frame3, text="Rcenter:")

eline4p = ttk.Entry(frame3, textvariable=rcp)

line4p.grid(row=4, column=0, sticky='w')

eline4p.grid(row=4, column=1, sticky='w')

button=tk.Button(frame3,text="High Pass", command=lambda:solvePi(1), width=15)

button.grid(column=2,row=1,sticky="w")

button=tk.Button(frame3,text="Low Pass",command=lambda:solvePi(2),width=15)

button.grid(column=2,row=2,sticky="w")

button=tk.Button(frame3,text="Clear", command=lambda:clear\_screen("p"),width=15)

button.grid(column=2,row=3,sticky="w" )

for i in frame3.winfo\_children():

    i.grid(pady=4, padx=6)

frame1.grid()

frame2.grid()

frame3.grid()

# add frames to notebook

notebook.add(frame1, text='L Matching')

notebook.add(frame2, text='T Matching')

notebook.add(frame3, text='Pi Matching')

root.mainloop()

1. **Chương trình PHTK L**

## **Các hàm để tính toán về PHTK L**

import math

def solve\_Q(zin, rl,f, loaimach):

    Q=None

    if(loaimach==1):

       Q=math.sqrt((zin/rl-1))

    if(loaimach==2):

       Q=math.sqrt((zin/rl-1))

    if(loaimach==3):

       Q=math.sqrt((rl/zin-1))

    if(loaimach==4):

       Q=math.sqrt((rl/zin-1))

    return Q

def solve\_L(zin, rl, f, loaimach):

   #1-4 tinh L

   #2-3 tinh C

    L=None

    Cs=None

    C=None

    Q=solve\_Q(zin, rl,f,loaimach)

    if(loaimach==1):

       L=Q\*rl/(2\*3.14\*f)

    if(loaimach==2):

       C=solve\_C(zin, rl, f, loaimach)

       L=rl\*\*2\*C/(1+Q\*\*2)

    if(loaimach==3):

       C=solve\_C(zin, rl, f, loaimach)

       Cs=((Q\*Q+1)/(Q\*Q))\*C

       L=1/((2\*3.14\*f)\*\*2\*Cs)

    if(loaimach==4):

       L=rl/(Q\*2\*3.14\*f)

    return L

def solve\_C(zin, rl, f, loaimach):

    C=None

    L=None

    Q=solve\_Q(zin, rl,f,loaimach)

    if(loaimach==1):

       L=solve\_L(zin, rl, f, 1)

       C=(L/rl\*\*2)/(Q\*\*2+1)

    if(loaimach==2):

       C=1/(rl\*Q\*2\*3.14\*f)

    if(loaimach==3):

       C=Q/(rl\*2\*3.14\*f)

    if(loaimach==4):

       L=solve\_L(zin, rl, f, 4)

       C=(L/(rl\*\*2))\*(Q\*\*2+1)

    return C

1. **Chương trình PHTK T**

## **Các hàm để tính toán về PHTK T**

import math

class T\_solve:

    def \_\_init\_\_(self, zin, rc,rl, f,loaimach):

        self.zin = zin

        self.rc=rc

        self.rl=rl

        self.f=f

        self.loaimach=loaimach

        self.w=2\*3.14\*self.f

    def Xparallel\_Source(self):

        return self.rc/self.solve\_Q1()

    def Xserial\_Source(self):

        return self.zin\*self.solve\_Q1()

    def Xparallel\_Load(self):

        return self.rc/self.solve\_Q2()

    def Xserial\_Load(self):

        return self.rl\*self.solve\_Q2()

    def solve\_L1(self):

        L1=None

        if(self.loaimach==1):

            #L1=Paralell1

            L1=self.Xparallel\_Source()/self.w

        if(self.loaimach==2):

            #L1=Serail1

            L1=self.Xserial\_Source()/self.w

        return L1

    def solve\_L2(self):

        L2=None

        if(self.loaimach==1):

            #L2=Parallel2

            L2=self.Xserial\_Load()/self.w

        if(self.loaimach==2):

            #L2=Serial2

            L2=self.Xserial\_Load()/self.w

        return L2

    def solve\_C1(self):

        C1=None

        if(self.loaimach==1):

            #C1=S1

            C1=1/(self.w\*self.Xserial\_Source())

        if(self.loaimach==2):

            #C1=P1

            C1=1/(self.w\*self.Xparallel\_Source())

        return C1

    def solve\_C2(self):

        C2=None

        if(self.loaimach==1):

            #C2=S2

            C2=1/(self.w\*self.Xserial\_Load())

        if(self.loaimach==2):

            #C2=P2

            C2=1/(self.w\*self.Xparallel\_Load())

      return C2

    def solve\_Q1(self):

        return math.sqrt((self.rc/self.zin)-1)

    def solve\_Q2(self):

        return math.sqrt((self.rc/self.rl)-1)

1. **Chương trình PHTK Pi**

## **Các hàm để tính toán về PHTK Pi**

import math

class Pi\_solve:

    def \_\_init\_\_(self, zin, rc,rl, f,loaimach):

        self.zin = zin  #input impedance of

        self.rc=rc

        self.rl=rl

        self.f=f

        self.loaimach=loaimach

        self.w=2\*3.14\*self.f

    def Xparallel\_Source(self):

        return self.zin/self.solve\_Q1()

    def Xserial\_Source(self):

        return self.rc\*self.solve\_Q1()

    def Xparallel\_Load(self):

        return self.rl/self.solve\_Q2()

    def Xserial\_Load(self):

        return self.rc\*self.solve\_Q2()

    def solve\_L1(self):

        L1=None

        if(self.loaimach==1):

            #L1=Paralell1

            L1=self.Xparallel\_Source()/self.w

        if(self.loaimach==2):

            #L1=Serail1

            L1=self.Xserial\_Source()/self.w

        return L1

    def solve\_L2(self):

        L2=None

        if(self.loaimach==1):

            #L2=Parallel2

            L2=self.Xserial\_Load()/self.w

        if(self.loaimach==2):

            #L2=Serial2

            L2=self.Xserial\_Load()/self.w

        return L2

    def solve\_C1(self):

        C1=None

        if(self.loaimach==1):

            #C1=S1

            C1=1/(self.w\*self.Xserial\_Source())

        if(self.loaimach==2):

            #C1=P1

            C1=1/(self.w\*self.Xparallel\_Source())

        return C1

    def solve\_C2(self):

        C2=None

        if(self.loaimach==1):

            #C2=S2

            C2=1/(self.w\*self.Xserial\_Load())

        if(self.loaimach==2):

            #C2=P2

            C2=1/(self.w\*self.Xparallel\_Load())

        return C2

    def solve\_Q1(self):

        return math.sqrt((self.zin/self.rc)-1)

    def solve\_Q2(self):

        return math.sqrt((self.rl/self.rc)-1)

1. **Giao diện phần mềm**
2. **PHTK L**

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

(nếu vi phạm điều kiện sẽ in ra “error”)

1. **PHTK T**

**A screenshot of a computer

Description automatically generated**

1. **PHTK Pi**

A screenshot of a computer

Description automatically generated

1. **Github**

[https://github.com/nguyenmanhdung183/phoihoptrokhang.git](%20https:/github.com/nguyenmanhdung183/phoihoptrokhang.git)