

SREDNJA ŠKOLA:

“17. SEPTEMBAR”

LAJKOVAC

TEMA:

**Simulacija rednog RLC
kola u programskom
jeziku Delphi**

Profesor: Nebojša Petrović

Učenik: Lukić Marko

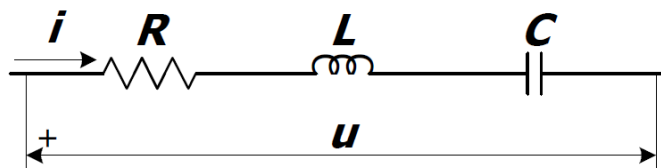
Lajkovac, Jun 2010

Sadržaj:

UVOD	3
OPIS IZGLEDA PROGRAMA.....	3
REZULTATI	4
OPIS RADA PROGRAMA.....	4
LITERATURA	9

UVOD

Ako imamo prijemnik koji se sastoji od redne veze otpornika, kalema i kondenzatora, onda taj prijemnik ima svoju **aktivnu, reaktivnu i prividnu otpornost** i razliku faza napona na krajevima prijemnika i struje koja protice kroz njega.



Impedansa prijemnika Z je **prividna otpornost prijemnika**. Definiše se kolicnikom amplitude napona i struje na prijemniku:

$$Z = \frac{U_m}{I_m} = \frac{U}{I} = \sqrt{R^2 + X^2}.$$

Impedansa je uvek pozitivna (amplitude U_m i I_m su uvek pozitivne).

R je **aktivna otpornost prijemnika** i *uvek je pozitivna*.

X je reaktivna otpornost prijemnika:

$$X = X_L - X_C = \omega L - \frac{1}{\omega C}.$$

$$X \begin{cases} > 0, & \omega L > \frac{1}{\omega C} & , \text{ kolo je pretežno induktivno} \\ = 0, & \omega L = \frac{1}{\omega C} & , \text{ rezonantno kolo} \\ < 0, & \omega L < \frac{1}{\omega C} & , \text{ kolo je pretežno kapacitivno} \end{cases}$$

Reaktivna otpornost može biti *pozitivna* (kada je induktivni deo veci od kapacitivnog). Tada kažemo da se prijemnik ponaša kao induktivnost. Reaktivna otpornost može biti *negativna* (kada je induktivni deo manji od kapacitivnog). Tada kažemo da se prijemnik ponaša kao kapacitivnost.

Rezonantno kolo je redno RLC kolo u kome kalem i kondenzator imaju takve vrednosti

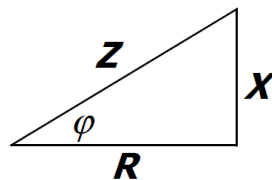
induktivnosti i kapacitivnosti da je reaktivna otpornost kola jednaka 0.

Glavne karakteristike rezonantnog kola su da je impedansa minimalna, a elektricna struja u grani maksimalna.

Fazna razlika napona i struje rednog RLC prijemnika je:

$$\varphi = \theta - \psi = \operatorname{arctg} \frac{X}{R} = \operatorname{arctg} \frac{\omega L - \frac{1}{\omega C}}{R}$$

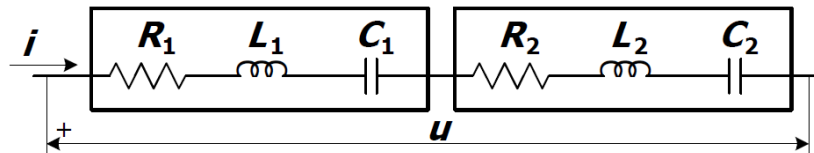
Svi ovi obrasci mogu se pamtitu pomoću "trougla impedansi":



$$\begin{aligned} Z &= \sqrt{R^2 + X^2} \\ \operatorname{tg} \varphi &= \frac{X}{R} \Rightarrow \varphi = \operatorname{arctg} \frac{X}{R} \\ R &= Z \cos \varphi \\ X &= Z \sin \varphi \end{aligned}$$

Redna veza dva prijemnika

Ako imamo rednu vezu dva RLC prijemnika, onda prilikom proračuna važe sledeća pravila:



Aktivne otpornosti u rednoj vezi se **smeju** sabirati:

$$R_e = R_1 + R_2$$

Reaktivne otpornosti u rednoj vezi se **smeju** sabirati:

$$X_e = X_1 + X_2$$

Impedanse u rednoj vezi se **ne smeju** sabirati:

$$Z_e \neq Z_1 + Z_2$$

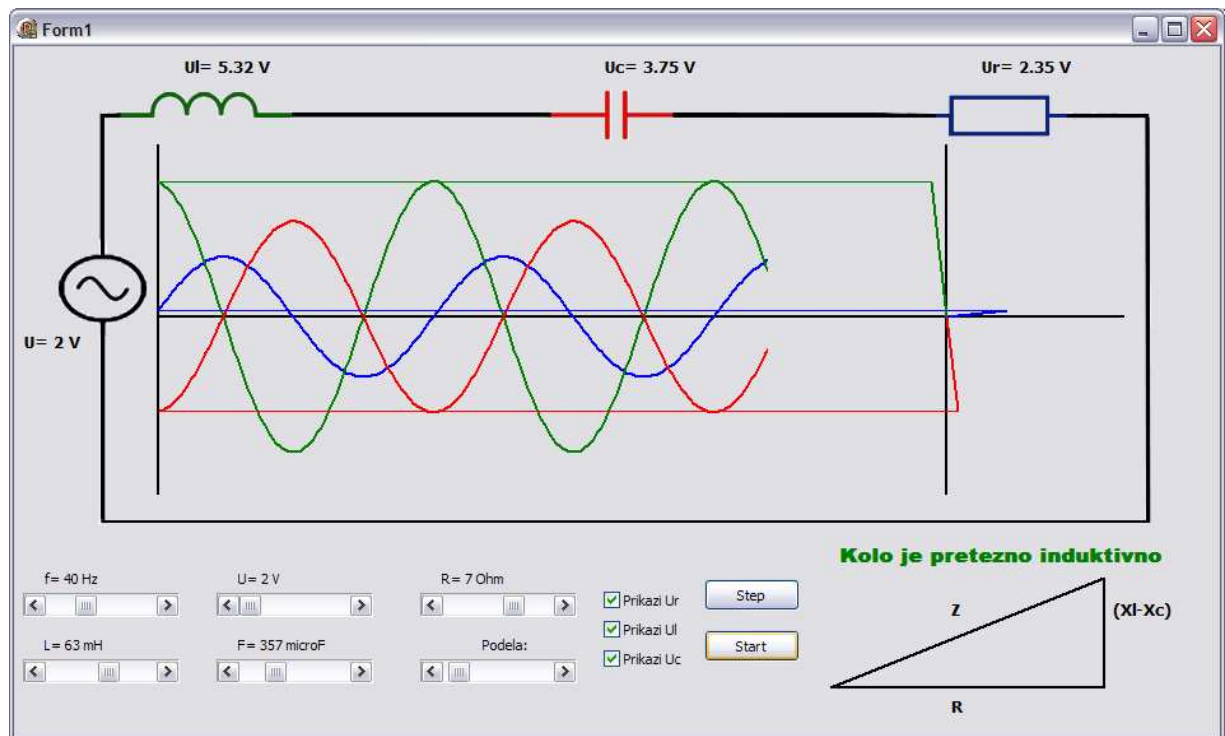
Fazne razlike u rednoj vezi se **ne smeju** sabirati:

$$\varphi_e \neq \varphi_1 + \varphi_2$$

U rednoj vezi se ne smeju sabirati parametri paralelne veze!

OPIS IZGLEDA PROGRAMA

Za simulaciju rada rednog RLC kola napisan je program u Delphi-ju. Pozivom programa dobija se forma kao na slici 1.



Slika 1.

Pritiskom na dugme **Start** pokreće se simulacija kola, odnosno pokreće se pomeranje sinusoida. Vrednosti promenljivih menjaju se na ScrollBar-ovima. Na formi je prikazana šema rednog RLC kola. Kao što se vidi sa slike: kalem, kondenzator i otpornik su obojeni razlicitim bojama, njihove boje odgovaraju bojama sinusoida (sinusoida zelene boje odgovara naponu na kalem, itd.). Iznad simbola za naponski izvor, kalem, kondenzator i otpornik prikazane su maksimalne vrednosti napona na njima. U donjem desnom uglu forme prikazan je trougao, koji se menja u trougao napona, impedance, i trougao RMS. Unutar

određene sinusoide i fazora (npr. Ako nije štikliran CheckBox **Prikaži Ur**, neće se videti fazor i sinusoida napona na otporniku).

REZULTATI

Dati su parametri rednog RLC kola, i to:

$$U = 2 \text{ V}, \quad f = 40 \text{ Hz}, \quad R = 7 \text{ Ohm}, \quad L = 63 \text{ mH}, \quad C = 357 \text{ microF}$$

Za date vrednosti parametara rednog RLC kola izracunate su vrednosti struje (I_m), vrednosti impedanse kola (Z), i vrednosti napona na kalemu, otporniku i kondenzatoru (U_l , U_r i U_c) i rezultati su dati u donjoj tabeli:

	Rezultati iz programa	Rezultati dobijeni računskim putem
Z	8.417 Ohm	8.416 Ohm
I_m	0.3360 A	0.33501 A
U_l	5.317 V	5.301 V
U_c	3.7469 V	3.7353 V
U_r	2.352 V	2.345 V

OPIS RADA PROGRAMA

Za crtanje sinusoide koristi se procedura čiji je kod sledeći:

```
Procedure TForm1.GrafikUI(start:integer);  
var  
  PodelaPoX: Integer;  
  Pocetak: TPoint;  
  interval: Double;  
  bmp : TBitmap;  
begin
```

```

PodelaPoX := (image1.Width ) div 4; // pixels in Pi
pocetak := Point(1, image1.Height div 2);
radian := 0;
interval := (4 * Pi) / 200.0;
bmp := TBitmap.Create;
bmp.Canvas.Pen.Width := 2;
bmp.Canvas.Pen.Color := clGreen;
bmp.Width := image3.Width;
bmp.Height := Image3.Height;
bmp.Transparent := true;
bmp.TransparentColor := clBtnFace;
for i := 0 to high(FPoints) do
begin
  FPoints[i].X := pocetak.x + Round((radian) * ((PodelaPoX-(f-30)) /pi));
  FPoints[i].Y := pocetak.y - Round(UI*sin(radian+pi/2+(start/5)) * podela);
  radian := radian + interval;
end;
if CheckBox3.Checked then
  bmp.Canvas.Polyline(Fpoints);
  Image2.Canvas.Draw(0,0,bmp);
  bmp.free;
end;

```

U ovoj proceduri se koristi metoda **DoubleBuffering**, iz razloga da bi se smanjio takozvani „flicker“, odnosno treptanje slika koje se crtaju. To se izvodi tako što se sve promene na slici crtaju u bitmap, pa se onda taj bitmap „lepi“ na canvas komponente Image.

PodelaPoX je promenljiva koja određuje broj piksela po jednoj poluperiodi. **Pocetak** je tačka koja se nalazi na levoj strani komponente Image. Promenljiva **radian** je vrednost trenutnog ugla izražena u radijanima. Za crtanje se koristi niz **Fpoints**. To je jednodimenzionalni statički niz tacaka:

FPoints: array [0..480] of TPoint;

Gde se X koordinata elementa niza racuna tako što se na x koordinatu pocetka doda proizvod trenutnog ugla i podele po x koja je umanjena za vrednost frekvencije. Umanjuje se za frekvenciju iz razloga da bi se sinusoida skupljala, odnosno širila pri promeni frekvencije.

FPoints[i].X := pocetak.x + Round((radian) * ((PodelaPoX-(f-30)) /pi));

Y koordinata elementa niza se racuna tako što se na Y koordinatu pocetka doda proizvod efektivne vrednosti napona na kalemu UI i sinusa trenutnog ugla. **Start/5** je promenljiva čija se vrednost menja unutar tajmera, i ona služi za pomeranje sinusoide u levu stranu, odnosno postizanje efekta pomeranja sinusoide. **Podela** određuje visinu sinusoide, i ona se menja na ScrollBar-u.

FPoints[i].Y := pocetak.y - Round(UI*sin(radian+pi/2+(start/5)) * podela);

Nakon kreiranja elementata niza tačke se spajaju komandom **bmp.Canvas.Polyline(Fpoints)**. Sada je na canvasu bitmapa nacrtana slika sinusoide, i onda se ona "lepi" na canvas komponente Image komandom **Image2.Canvas.Draw(0,0,bmp)**;

Procedure za crtanje ostale dve sinusoide su skoro identične sa ovom procedurom. Razlikuju se po tome što se pri računanju Y koordinate elementa niza Fpoints, umesto Ul, sinus trenutnog ugla množi sa Ur ili Uc.

Za pomeranje fazora koriste se dve procedure. Jedna za kreiranje tačaka i druga za crtanje i pomeranje fazora. Procedura za kreiranje tačaka ima sledeći kod:

```

Procedure TForm1.KreiranjeTacaka(start:integer);
begin
    pocetak1 :=Point(PaintBox1.Width - PaintBox1.Height div 2,PaintBox1.Height div
2);
    for i := 0 to high(tacke) do
        Begin
            // kondenzator
            Tacke[i].X := pocetak1.X + Round(Uc*cos(+pi/2-(start/5)) * podela);
            Tacke[i].Y := pocetak1.y + Round(Uc*sin(+pi/2-(start/5)) * podela);
            // otpornik
            Tacke1[i].X := pocetak1.X + Round(Ur*cos(-(start/5)) * podela);
            Tacke1[i].Y := pocetak1.y + Round(Ur*sin(-(start/5)) * podela);
            // kalem
            Tacke2[i].X := pocetak1.X + Round(Ul*cos(-pi/2-(start/5)) * podela);
            Tacke2[i].Y := pocetak1.y + Round(Ul*sin(-pi/2-(start/5)) * podela);
        End;
    end;

```

Pocetak1 je tačka koja se nalazi na desnoj strani komponente Image, odnosno označava koordinatni početak fazorskog dijagrama. Za crtanje fazora koriste se tri dinamička jednodimenzionalna niza tačaka. Koriste se dinamički nizovi opet iz razloga da bi se smanjio "flicker", odnosno da bi se smanjio broj prolaza u For petlji, i tako smanjio broj iscrtavanja. Elementi nizova, odnosno tačke obrazuju kružnicu. X koordinata računa se kao kosinus trenutnog ugla, a Y koordinata kao sinus trenutnog ugla. Kada se kreiraju tačke poziva se procedura koja crta fazor, odnosno spaja tačku iz gore kreiranog niza sa koordinatnim početkom. Ta procedura ima sledeći kod:

```

procedure TForm1.PomeranjeFazoraUc;
begin
    pocetak1 := Point(PaintBox1.Width - PaintBox1.Height div 2,PaintBox1.Height div
2);
    KreiranjeTacaka(count1);
    for i := high(tacke) - 1 to high(tacke) do
        Begin
            // kondenzator
            PaintBox1.Refresh;
            if CheckBox1.Checked then
                Begin

```



```

    PaintBox1.Canvas.Pen.Width := 2;
    PaintBox1.Canvas.Pen.Color := clRed;
    PaintBox1.Canvas.MoveTo(pocetak1.X,pocetak1.Y);
    PaintBox1.Canvas.LineTo(tacke[i].X,tacke[i].y);
    PaintBox1.Canvas.Pen.Width := 1;
    PaintBox1.Canvas.MoveTo(tacke[i].X,tacke[i].y);
    PaintBox1.Canvas.LineTo(1,tacke[i].Y);
End;
//    Otpornik
if CheckBox2.Checked then
Begin
    PaintBox1.Canvas.Pen.Width := 2;
    PaintBox1.Canvas.Pen.Color := clBlue;
    PaintBox1.Canvas.MoveTo(pocetak1.X,pocetak1.Y);
    PaintBox1.Canvas.LineTo(tacke1[i].X,tacke1[i].y);
    PaintBox1.Canvas.Pen.Width := 1;
    PaintBox1.Canvas.MoveTo(tacke1[i].X,tacke1[i].y);
    PaintBox1.Canvas.LineTo(1,tacke1[i].Y);
End;
//    Kalem
if CheckBox3.Checked then
Begin
    PaintBox1.Canvas.Pen.Width := 2;
    PaintBox1.Canvas.Pen.Color := clGreen;
    PaintBox1.Canvas.MoveTo(pocetak1.X,pocetak1.Y);
    PaintBox1.Canvas.LineTo(tacke2[i].X,tacke2[i].y);
    PaintBox1.Canvas.Pen.Width := 1;
    PaintBox1.Canvas.MoveTo(tacke2[i].X,tacke2[i].y);
    PaintBox1.Canvas.LineTo(1,tacke2[i].Y);
End;
End;
end;

```

Pritiskom na dugme Start poziva se sledeća procedura:

```

procedure TForm1.Button2Click(Sender: TObject);
begin
if Button2.Caption = 'Start' then
Begin
    Button2.Caption := 'Pause';
    Timer1.Enabled := true;
    Timer2.Enabled := True;
    Timer3.Enabled := true;
    DoubleBuffered := true
End
Else

```

```
Begin  
  Button2.Caption := 'Start';  
  Timer1.Enabled := False;  
  Timer2.Enabled := False;  
  Timer3.Enabled := False;  
End;  
end;
```

Ono što radi ova procedura je sledeće: ako je caption dugmeta **'Start'** onda se aktiviraju tajmeri i menja se caption dugmeta na **'Pause'**, odnosno ako je caption dugmeta **'Pause'** onda se deaktiviraju tajmeri i menja se caption dugmeta u **'Start'**.

Pritiskom na dugme **'Step'** poziva se sledeća procedura:

```
procedure TForm1.Button3Click(Sender: TObject);  
begin  
  Timer1.Enabled := false;  
  Inc(count1);  
  GrafikUc(Count1);  
  GrafikUr(Count1);  
  GrafikUl(Count1);  
  
  SetLength(Tacke,count);  
  SetLength(Tacke1,count);  
  SetLength(Tacke2,count);  
  PomeranjeFazoraUc;  
  Inc(count);  
  Button2.Caption := 'Start';  
end;
```

Ova procedura služi za pomeranje sinusoida i fazora "korak po korak". Ono što ona radi je da svakim pritiskom na ovo dugme, povećava dva brojača **Count** i **Count1** za 1. I nakon toga poziva tri procedure za crtanje sinusoida, postavlja dužinu dinamičkih nizova na trenutnu vrednost brojača **Count** i nakon toga poziva proceduru za crtanje fazora.

Prvi tajmer, odnosno **Timer1** ima interval 100 ms i u okviru njega se pozivaju sve procedure za crtanje i povećavaju se vrednosti brojača **Count** i **Count1**. Druga dva tajmera služe za ispisivanje teksta na trouglu.

LITERATURA

1. „Osnove elektrotehnike “ za drugi razred srednjih škola
2. „Wikipedia“ <http://www.wikipedia.org/>
3. Sajt Više Elektrotehničke škole na Voždovcu www.viser.edu.rs