

```

function Ausgabe = Triode (freq, Vruhe, Usupply, rp, rb, RL, RK, strp, ktrp)

mufix = 105; %Verstärkungsfaktor
Fs = 100000; %Sample Rate
t = 0:(1/Fs):.01; % Berechne Zeitvektor

IBias = ( 1 / RL ) * ( Usupply - Vruhe ); %Bias-Strom ermitteln
VBias = RK * IBias ; %Bias-Spannung an Gitter ermitteln

%berechne koeffizienten für ip-Funktion
a1 = mufix / (rp+rb);
a2 = -mufix ^ 2 * rp * strp / (rp + rb) ^ 3 / 0.2;
a3 = mufix ^ 3 * rp * ( (2 * rp - rb)* strp^2 - rp*(rp+rb) * ktrp ) / (6 * (rp+rb) ^ 5 );

%Ug=0V Punkt der ip-funktion für Offsetberechnung im Overload-fall
OVip = a1 * (Vruhe / mufix) + a2 * (Vruhe / mufix) ^ 2 + a3 * (Vruhe / mufix) ^ 3;

%%Berechnung der Koeffizienten für Tanh ->
%%tanfunc=p1*( tanh( p2*(eg1nl) ) +p3)
%Punkt, an dem die Kurve nichtlinear wird
ugnl=1.333e-007*Vruhe^3 + -8e-005*Vruhe^2 + 0.007667*Vruhe + -0.9;

%Punkt, an dem ib=0A erreicht werden soll -> ug0A->ugoa
ugoa= -0.06764*ugnl^3 + 0.0487*ugnl^2 + 1.855*ugnl + -0.7189;
%Wert der ip-Funktion für ugnl
offsetnl=0.3;
enl=(ugnl+offsetnl) + Vruhe/mufix;
ipnl= a1 * enl + a2 * enl ^ 2 + a3 * enl ^ 3;
ipnl=ipnl+IBias;
%Steigung der ip-Funktion für ugnl
sfipnl=a1 + 2 * a2 * enl + 3 * a3 * enl^2;
%Berechne Parameter für Tanh
ugoakorr=ugoa-ugnl+offsetnl;

%Berechne Parameter p2 durch Newton-Verfahren

x0=1.5; %Starwert
%Erste Iteration
fx= -sfipnl * tanh(x0*ugoakorr) / ipnl -x0;
fxab= -sfipnl * (1 - tanh(x0 * ugoakorr) ^ 2) * ugoakorr / ipnl - 1;
xneu=x0-(fx/fxab);
xalt=xneu;
%2.- 10. Iteration
while fx>0.001
    fx= -sfipnl * tanh(xalt*ugoakorr) / ipnl -x0; %Berechne f(xn-1)
    fxab= -sfipnl * (1 - tanh(xalt * ugoakorr) ^ 2) * ugoakorr / ipnl - 1; %
    Berechne f'(xn)
    xneu=xalt-(fx/fxab); %Berechne xn
    xalt=xneu; %Wert für neue Iteration einsetzen
end

p2=xneu; %Übertrage Wert aus letzter Iteration

%Berechne p1 und p3
p1=sfipnl/p2;

```

```
p3=ipnl/p1;
```

```
%%Ende der Konstantenberechnung
```

```
function [egreturn] = eg(freq, VBias) %Generierung des Eingangssignales
```

```
egreturn= -5:0.01:2; % Berechne Werte-Sweep
```

```
%egreturn= 0.6 * sin(2 * pi * freq * t)- VBias;
```

```
end
```

```
function createfigurecurve(X1, Y1)
```

```
%CREATEFIGURE(X1,Y1)
```

```
% X1: vector of x data
```

```
% Y1: vector of y data
```

```
% Auto-generated by MATLAB on 19-Jun-2009 17:06:12
```

```
% Create figure
```

```
figure1 = figure('PaperSize',[20.98 29.68]);
```

```
% Create axes
```

```
axes('Parent',figure1, 'YGrid','on','XGrid','on');
```

```
box('on');
```

```
hold('all');
```

```
% Create plot
```

```
plot(X1,Y1);
```

```
% Create title
```

```
title('Ib-Ug1 bei festem Verstärkungsfaktor und Anodenwiderstand');
```

```
% Create xlabel
```

```
xlabel('Ug1 in V');
```

```
% Create ylabel
```

```
ylabel('Ib in A');
```

```
end
```

```
function createfigurefft(X1, Y1)
```

```
%CREATEFIGURE(X1,Y1,X2,Y2)
```

```
% X1: vector of x data
```

```
% Y1: vector of y data
```

```
% X2: vector of x data
```

```
% Y2: vector of y data
```

```
% Auto-generated by MATLAB on 19-Jun-2009 17:11:02
```

```
% Create figure
```

```
figure1 = figure('PaperType','a4letter','PaperSize',[20.98 29.68]);
```

```
% Create axes
```

```
axes('Parent',figure1, 'YGrid','on','XGrid','on');
```

```

% Uncomment the following line to preserve the X-limits of the axes
% xlim([0 3.5e+004]);
% Uncomment the following line to preserve the Y-limits of the axes
% ylim([0 0.0008]);
box('on');
hold('all');

```

```

% Create plot
stem(X1,Y1, 'Color',[0 0 1]); %plot(X1,Y1);

```

```

ylim([0 0.001]);
xlim([15 15000]);

```

```

% Create title
title('Single-Sided Amplitude Spectrum of y(t)');

```

```

% Create xlabel
xlabel('Frequency (Hz)');

```

```

% Create ylabel
ylabel('|Y(f)|');

```

```

end

```

```

%Generiere Eingangssignal
eg1=eg(freq, VBias);

```

```

n=length(eg1);

```

```

%berechne Faktor für Stromberechnung
for i=1:1:n,
    e1(i)= eg1(i)+ Vruhe / mufix; %Berechne e-faktor
end

```

```

%berechne anodenstrom

```

```

for i=1:1:n,

```

```

    if (eg1(i)<=(ugnl+offsetnl)) %Nichtlinearer Teil der Kennlinie

```

```

        eg1tan=eg1(i)-(ugnl+offsetnl); %Verschiebe Wert für tanfunc

```

```

        tanfunc = p1*(tanh(p2*eg1tan)+p3); %Berechne AC-Anteil im nichtlinearen

```

```

        Fall

```

```

        ib(i) = tanfunc; %Addiere Gleichanteil für totalen Anodentrom

```

```

        if (ib(i)<=0) %Negative Werte der Tanh-Funktion eliminieren ->

```

```

        Clipping/Cutoff

```

```

            ib(i)=0;

```

```

        end

```

```

    elseif (eg1(i)>(ugnl+offsetnl) && eg1(i)<=0) %Quasi-linearer Teil der

```

```

    Kennlinie

```

```
ip(i) = a1 * e1(i) + a2 * e1(i) ^ 2 + a3 * e1(i) ^ 3; %Wert des AC-Anteiles
des Anodenstromes

ib(i) = IBias + ip(i); %Addiere Gleichanteil für totalen Anodenstrom

elseif (eg1(i)>0) %Overload/Overdrive

odi = 0.0007 * (1 - exp( -eg1(i) / 0.55 ));

ib(i) = IBias + OVip + odi; %Addiere Gleichanteil, Offset zu 0V und
Overload-Funktion

end

end;

%berechne totale-anodenspannung

for i=1:1:n,

eb(i) = Vruhe - ( ( ib(i) - IBias ) * rb ) ; %Totale Anodenspannung

end;

%% Ausgabe der Werte

%%Ausgabe und FFT
Ausgabe = [eb; ib; eg1;];

createfigurecurve(eg1,ib);

L = 1001;

NFFT = 2^nextpow2(L); % Next power of 2 from length of y
Y = fft(ib,NFFT)/L;
f = Fs/2*linspace(0,1,NFFT/2);

% Plot single-sided amplitude spectrum.
createfigurefft(f,2*abs(Y(1:NFFT/2)));

end

%%
```

Ib-Ug1 bei festem Verstärkungsfaktor und Anodenwiderstand

