Segundo Relatório de Lab de Eletronica 1

Henrique da Silva
henrique.pedro@ufpe.br

13 de julho de 2023

Sumário

1 Introdução

Neste relatório, descreveremos o experimento realizado para analisar um circuito conhecido como "grampeador de tensão". O circuito utiliza um diodo e um capacitor para fixar o valor mínimo de tensão em 0 V. Realizamos uma análise teórica e simulações computacionais, seguidas pela montagem prática do circuito em uma protoboard e medições de tensão de saída. O objetivo é aprofundar nosso conhecimento sobre circuitos com diodos e aplicar técnicas de análise teórica e prática em eletrônica.

Todos arquivos utilizados para criar este relatório, e o relatorio em si estão em: `https://github.com/Shapis/ufpe_ee/tree/main/6thsemester/Eletronica1/`

O código utilizado para a análise numérica também se encontra no anexo ao final do relatório.

2 Análise preliminar

Na análise teórica, considera-se o comportamento não linear do diodo e a capacidade de armazenamento do capacitor. Determina-se as equações que descrevem a relação entre a tensão de entrada e a tensão de saída do circuito, levando em conta os parâmetros do diodo e do capacitor. Com essas equações permite-se compreender como o circuito "grampeia"a tensão mínima em 0 V e como a forma e amplitude da onda de entrada afetam a tensão de saída.

2.1 O circuito

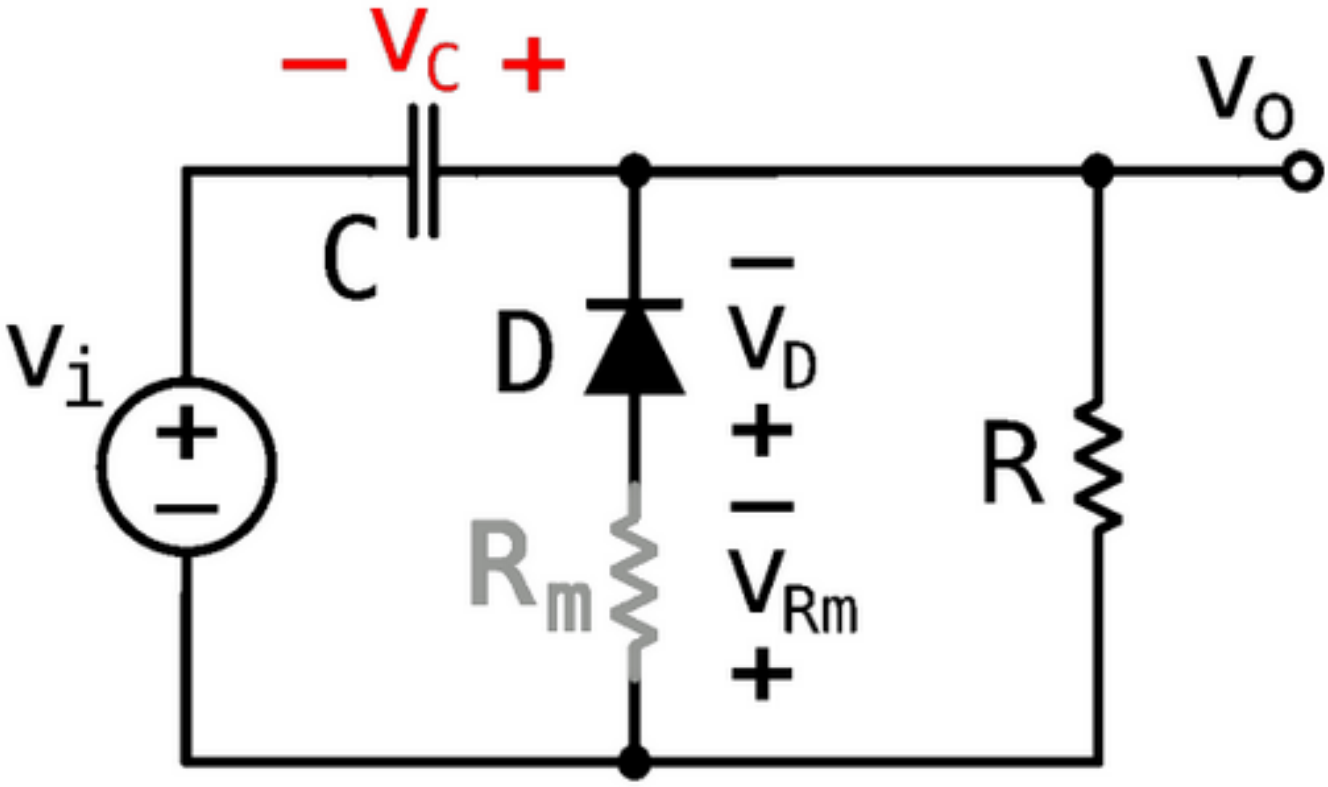

Figura 1: Circuito com diodo em configuração de grampeador.

2.2 Análise simbólica

A análise é conduzida, examinando-se cada estado do diodo separadamente. O processo tem início com o diodo polarizado diretamente, seguido pelo diodo polarizado reversamente.

2.2.1 Estado 1: Polarizacao direta

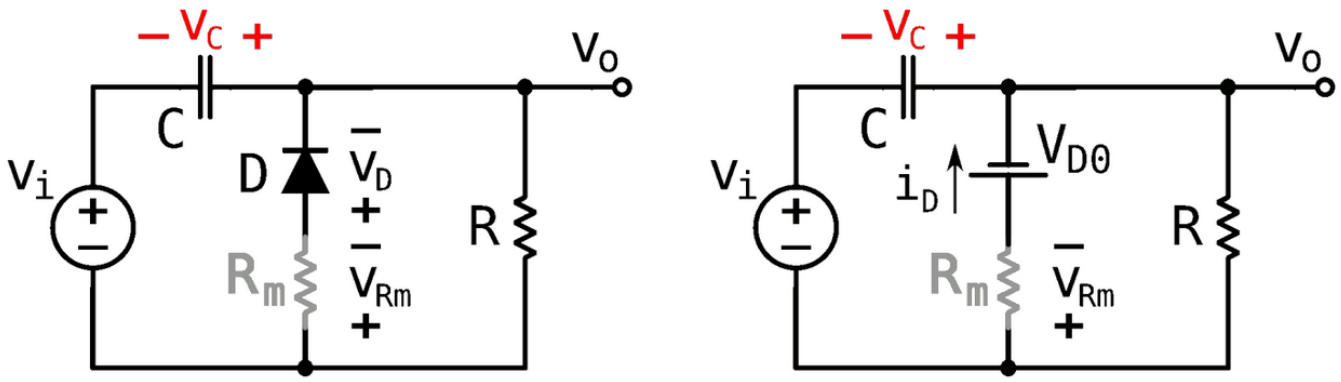

Figura 2: Circuito grampeador de tensao em configuração de polarizacao direta.

Esse estado é alcançado quando o diodo está "ligado".
Nesse caso, existem restrições.

$$\begin{aligned} I_d &> 0 \\ V_d &= V_{D0} \end{aligned} \tag{1}$$

A lei de Kirchhoff é aplicada ao nó denominado V_o.

$$\frac{V_o}{R} + \frac{V_o - (-V_{D0})}{R_m} + C\frac{\mathrm{d}V_c}{\mathrm{d}t} = 0 \tag{2}$$

Observa-se que:

$$V_c = V_o - V_i \tag{3}$$

$$\frac{\mathrm{d}V_c}{\mathrm{d}t} + \left(\frac{1}{R_m} + \frac{1}{R}\right)\frac{V_o}{C} = -\frac{V_{D0}}{R_m C} \tag{4}$$

A equação é rearranjada:

$$\frac{\mathrm{d}V_c}{\mathrm{d}t} + \left(\frac{1}{R_m} + \frac{1}{R}\right)\frac{V_o}{C} = -\frac{V_{D0}}{R_m C} \tag{5}$$

Como V_c é igual a $V_o - V_i$, obtém-se:

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} + \frac{1}{C}\left(\frac{1}{R_m} + \frac{1}{R}\right)V_o = \frac{\mathrm{d}V_i}{\mathrm{d}t} - \frac{V_{D0}}{R_m C} \tag{6}$$

A condição inicial para V_o em função de $V_{C0} = v(t = 0)$ é dada por $V_o(0) = V_{C0} + V_i$.
Nesta prática, são feitas duas premissas:

- $R_m \ll R$.

- V_i é uma onda quadrada com amplitude $\pm V_m$ e período T_s.

A partir da primeira premissa, obtém-se:

$$\frac{1}{R_m} + \frac{1}{R} \approx \frac{1}{R_m} \tag{7}$$

E da segunda premissa, durante as transições entre $\pm V_m$, tem-se:

$$\frac{\mathrm{d}V_i}{\mathrm{d}t} = 0 \tag{8}$$

Pode-se então simplificar a equação 6 da seguinte forma:

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} + \frac{V_o}{R_m C} = - - \frac{V_{D0}}{R_m C} \tag{9}$$

Com a condição inicial: $V_o(t = 0^+) = V_{C0} \pm V_m$.

A solução da equação diferencial 9 é dada pela soma da solução particular e da solução homogênea, como segue:

$$\frac{\mathrm{d}V_{oH}}{\mathrm{d}t} + \frac{\mathrm{d}V_{oH}}{\mathrm{d}R_m C} = 0 \qquad \text{Parte homogenea}$$

$$V_{oH}(t) = K e^{-\frac{t}{R_m C}} \qquad \text{Solucao generica}$$

Com K determinado pelas condições iniciais.
Por substituição, determina-se que uma solução particular pode ser obtida por:

$$V_{oP}(t) = -V_{D0} \tag{10}$$

Assim, tem-se a solução completa como:

$$V_o(t) = K e^{-\frac{t}{R_m C}} - V_{D0} \qquad \text{Solucao completa}$$

Como a condição inicial é $V_o(0) = V_{C0} \pm V_m$, a solução completa é dada por:

$$V_o(t) = (V_{C0} \pm V_m + V_{D0}) e^{\frac{-t}{R_m C}} - V_{D0} \tag{11}$$

Como $V_c = V_o - V_i$, temos a seguinte solução para $V_c(t)$:

$$V_c(t) = (V_{C0} \pm V_m + V_{D0}) e^{\frac{-t}{R_m C}} \mp V_m - V_{D0} \tag{12}$$

Dessa forma, obtém-se a constante de tempo $\tau_1 = R_m C$.
Das restrições desse estado, tem-se que:

$$I_D = \frac{V R_m}{R_m} = -\frac{V_i + V_c + V_{D0}}{R_m} > 0 \tag{13}$$

Logo:

$$V_i + V_c < -V_{D0} \tag{14}$$

Com base nisso, as equações que regem as tensões V_c (Equação 12) e V_o (Equação 11) no estado 1 são obtidas.

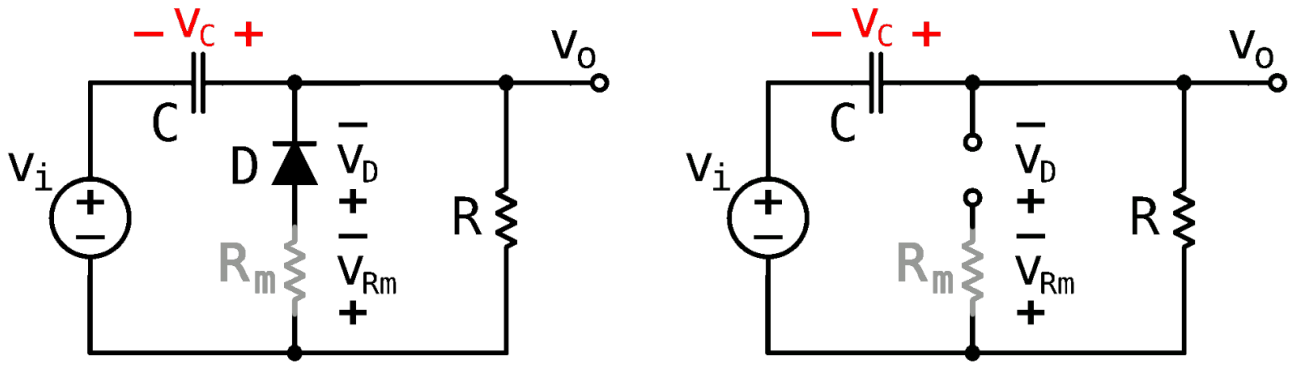

Figura 3: Circuito grampeador de tensao em configuração de polarizacao reversa.

2.2.2 Estado 2: Polarizacao reversa

Esse estado é alcançado quando o diodo está "desligado".
Nesse caso, existem restrições.

$$\begin{aligned} V_D &< V_{D0} \\ I_D &= 0 \end{aligned} \tag{15}$$

Nesse estado, observa-se que é um simples circuito RC com a tensão de saída medida sobre o resistor R.

Observa-se que $V_r = -R_{ic} = -RC\frac{\mathrm{d}V_c}{\mathrm{d}t}$, e a equação que governa o circuito é dada por:

$$V_i + V_c + RC\frac{\mathrm{d}V_c}{\mathrm{d}t} = 0 \tag{16}$$

Considerando que $V_c = V_o - V_i$, podemos concluir que:

$$V_o = -RC\left(\frac{\mathrm{d}V_o}{\mathrm{d}t} - \frac{\mathrm{d}V_i}{\mathrm{d}t}\right) \tag{17}$$

Isso pode ser rearranjado em:

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} + \frac{V_o}{RC} = \frac{\mathrm{d}V_i}{\mathrm{d}t} \tag{18}$$

A entrada V_i é uma onda quadrada, o que implica que $\frac{\mathrm{d}V_i}{\mathrm{d}t} = 0$, simplificando assim a equação para:

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} + \frac{V_o}{RC} = 0 \tag{19}$$

A condição inicial é $V_o(t = 0^+) = \pm V_m + V_{C0}$.
Portanto, a solução para V_o é:

$$V_o(t) = (\pm V_m + V_{C0})e^{\frac{-t}{RC}} \tag{20}$$

Como $V_c = V_o - V_i$, temos a seguinte solução para $V_c(t)$:

$$V_c(t) = (\pm V_m + V_{C0})e^{\frac{-t}{RC}} \mp V_m \tag{21}$$

Dessa forma, obtém-se a constante de tempo $\tau_2 = RC$.
Como $R_m \ll R$, pode-se observar que $\tau_1 \ll \tau_2$.

6 Apêndice

Abaixo se encontra o código utilizado para a análise simbólica e numérica do circuito.

```
import matplotlib.pyplot as plt
import sympy as smp
from sympy import *

# Definindo as variaveis simbolicas
Vo, Vi, Va, Vc, R1, R2, Rp, Cp, A, w, j, Hjw, wp, wc, K = smp.symbols(
'V_o V_i V_a V_c R_1 R_2 R_p C_p A w j H_jw w_p, w_c, K', real=True)

# Analise nodal do circuito

eq1 = smp.Eq((Va - Vi)/R1 + (Va - Vo)/R2 + Va/(Rp + (1/(j * w * Cp))), 0)
eq2 = smp.Eq(Va * ((1/(j * w * Cp)))/(((1/(j * w * Cp))) + Rp), -Vc)
eq3 = smp.Eq(A * Vc, Vo)
# print('Equacoes em latex:')
# smp.pprint(smp.latex(eq1))
# smp.pprint(smp.latex(eq2))
# smp.pprint(smp.latex(eq3))
# print("")

print("Equacoes do circuito:")
print("Equacao 1:")
smp.pprint(eq1)
print("Equacao 2:")
smp.pprint(eq2)
print("Equacao 3:")
smp.pprint(eq3)
print("")

sols = smp.solve([eq1, eq2, eq3], [Va, Vc, Vo])

print("Solucao para Vo:")
smp.pprint(sols[Vo])
# print("")
# smp.pprint(smp.latex(sols[Vo]))
print("")

print("Aqui fazemos a seguinte simplificacao:")
print("Rp >> R1 , Rp >> R2, e A >> 1")

Vo = (-A * R2 * Vi)/((Rp*(R1 + R2))*j*w*Cp + A*R1)

print("Equacao de Vo simplificada:")
smp.pprint(Vo)
# smp.pprint(smp.latex(Vo))
print("")

eqHjw = smp.Eq(Hjw, Vo/Vi)

print("Equacao de Hjw:")
smp.pprint(eqHjw)
print("")

print("Resolvendo a equacao Hjw e colocando no formato canonico da um fitro
                                        passa baixa (-K wp / (jw + wp))
                                        obtemos o seguinte:")

eqHjw = smp.Eq(Hjw, -(K * wc) / (I * w + wc))

Hjw = -(K * wc) / (I * w + wc)
```

```
smp.pprint(eqHjw)

print("")

eqwp = smp.Eq(wp, 1/(Rp*Cp))
smp.pprint(eqwp)

eqK = smp.Eq(K, R2/R1)
smp.pprint(eqK)

eqwc = smp.Eq(wc, (A*wp)/(1 + K))
smp.pprint(eqwc)

print("")

# Hjw = Hjw.subs({K: R2/R1, wc: A*wp/(1+K)})

print("Valor absoluto de Hjw:")
Hjw_abs = smp.Abs(Hjw)
smp.pprint(Hjw_abs)
# smp.pprint(smp.latex(Hjw_abs))
print("")


print("Exemplo 1:")
print("Para R1 = 4.7E3 ohms, R2 = 2.2E4 ohms, wp = 2E1 pi e A = 1E5\n")

eqK1 = eqK.subs({R1: 4.7E3, R2: 2.2E4})
K1 = smp.solve(eqK1, K)[0]
smp.pprint(eqK1)
print("")

eqwc1 = eqwc.subs({A: 1E5, wp: 20 * smp.pi, K: K1}).evalf()
wc1 = smp.solve(eqwc1, wc)[0]
smp.pprint(eqwc1)
print("")

print("Valores absolutos para w = [0.5, 1, 2, 4, 10, 20, 40]*wc:")

# ex1interval = [0.02, 0.01, 0.05, 0.2, 0.5, 1, 2, 4, 10, 20, 40]
ex1interval = [0.5, 1, 2, 4, 10, 20, 40]

ex1Vals = []

for val in ex1interval:
temp = Hjw_abs.subs({w: wc1 * val, wc: wc1, K: K1})
ex1Vals.append(temp)
print('para freq:', round(((val*wc1)/(2*smp.pi)).evalf(), 2),
'\ttemos:', round(temp, 2))

print("")

print("Exemplo 2:")
print("Para R1 = 4.7E3 ohms, R2 = 5.6E5 ohms, wp = 2E1 pi e A = 1E5\n")

eqK2 = eqK.subs({R1: 4.7E3, R2: 5.6E5})
K2 = smp.solve(eqK2, K)[0]
smp.pprint(eqK2)
print("")

eqwc2 = eqwc.subs({A: 1E5, wp: 20 * smp.pi, K: K2}).evalf()
wc2 = smp.solve(eqwc2, wc)[0]
smp.pprint(eqwc2)
print("")
```

```
ex2interval = [0.5, 1, 5, 20, 50, 200, 500, 1000]
ex2Vals = []

for val in ex2interval:
temp = Hjw_abs.subs({w: wc2 * val, wc: wc2, K: K2})
ex2Vals.append(temp)
print('para freq:', round(((val*wc2)/(2*smp.pi)).evalf(), 2),
'\ttemos:', round(temp, 2))
print("")


# Plotando os graficos

frequencias_plot = [i for i in range(1, 100000, 100)]

plotH1 = [Hjw_abs.subs({w: i, wc: wc1, K: K1}) for i in frequencias_plot]
plotH2 = [Hjw_abs.subs({w: i, wc: wc2, K: K2}) for i in frequencias_plot]

fig, ax = plt.subplots()

ax.plot(frequencias_plot, plotH1, color='blue', label='Exemplo 1')
ax.plot(frequencias_plot, plotH2, color='orange', label='Exemplo 2')
ax.legend(['Exemplo 1', 'Exemplo 2'])
plt.xlabel('w rad/s')
plt.ylabel('|H(jw)|')
plt.title('Magnitude de H(jw)')


plt.show()

# Medicoes na pratica

# Exemplo1
```

7 Anexos

Código utilizado para geração de gráficos de bode, e análise utilizando as frequências de cortes obtidas experimentalmente.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit



fc1 = 210000
fc2 = 11800


freqs1 = np.array([0.002*fc1, 0.01*fc1, 0.05*fc1, 0.2*fc1, 0.5*fc1,
        0.8*fc1, fc1, 2*fc1, 4*fc1, 10*fc1, 20*fc1, 40*fc1])
vin1 = np.array([0.2975, 0.297, 0.298, 0.297, 0.298,
        0.301, 0.297, 0.297, 0.3, 0.302, 0.301, 0.315])
vout1 = np.array([1.404, 1.405, 1.403, 1.393, 1.306, 1.124,
        0.975, 0.54, 0.276, 0.114, 0.057, 0.031])


freqs2 = np.array([0.05*fc2, 0.1*fc2, 0.2*fc2, 0.5*fc2, 0.8*fc2, fc2,
        2*fc2, 5*fc2, 20*fc2, 50*fc2, 200*fc2, 500*fc2, 1000*fc2])
vin2 = np.array([0.1191, 0.1199, 0.12, 0.121, 0.122, 0.122,
        0.1218, 0.123, 0.123, 0.121, 0.121, 0.125, 0.151])
vout2 = np.array([14.01, 13.95, 13.77, 12.61, 10.85, 9.8, 5.92,
        2.548, 0.645, 0.258, 0.0652, 0.0205, 0.0122])




def mag_sqr_fun(f, K, fc):
return (K*fc)**2/(f**2 + fc**2)


def dB(m):
return 20*np.log10(m)



(K1, fc1), _ = curve_fit(lambda f, K, fc: dB(mag_sqr_fun(f, K, fc)),
freqs1, 2*dB(vout1/vin1))


print(f"""O ganho K eh {round(K1, 1)}
A frequencia de corte eh {round(fc1, 1)} Hz""")

f1 = np.logspace(np.log10(freqs1[0]) - 1, np.log10(freqs1[-1]) + 0.3)
mag1 = dB(mag_sqr_fun(f1, K1, fc1))/2

plt.semilogx(f1, mag1)
plt.semilogx(freqs1, dB(vout1/vin1), "*")
plt.xlabel("Freq (Hz)")
plt.ylabel("Mag(H) (dB)")
plt.title("Grafico de Bode de magnitude para o 1 circuito")
plt.grid()
plt.savefig("figura1.png")
plt.show()
```

```
    (K2, fc2), _ = curve_fit(lambda f, K, fc: dB(mag_sqr_fun(f, K, fc)),
    freqs2, 2*dB(vout2/vin2))

    print("\n\nResultados para o 2 circuito:\n")
    print(f"""O ganho K eh {round(K2, 1)}
    A frequencia de corte eh {round(fc2, 1)} Hz""")

    f2 = np.logspace(np.log10(freqs2[0]) - 1, np.log10(freqs2[-1]) + 0.3)
    mag2 = dB(mag_sqr_fun(f2, K2, fc2))/2

    plt.semilogx(f2, mag2)
    plt.semilogx(freqs2, dB(vout2/vin2), "*")
    plt.xlabel("Freq (Hz)")
    plt.ylabel("Mag(H) (dB)")
    plt.title("Grafico de Bode de magnitude para o 2 circuito")
    plt.grid()
    plt.savefig("figura2.png")
    plt.show()




    plt.semilogx(f1, mag1)
    plt.semilogx(f2, mag2)
    plt.semilogx(freqs1, dB(vout1/vin1), "*")
    plt.semilogx(freqs2, dB(vout2/vin2), "*")
    plt.xlabel("Freq (Hz)")
    plt.ylabel("Mag(H) (dB)")
    plt.title("Grafico de Bode de magnitude para ambos os circuitos")
    plt.grid()
    plt.savefig("figura3.png")
    plt.show()
```