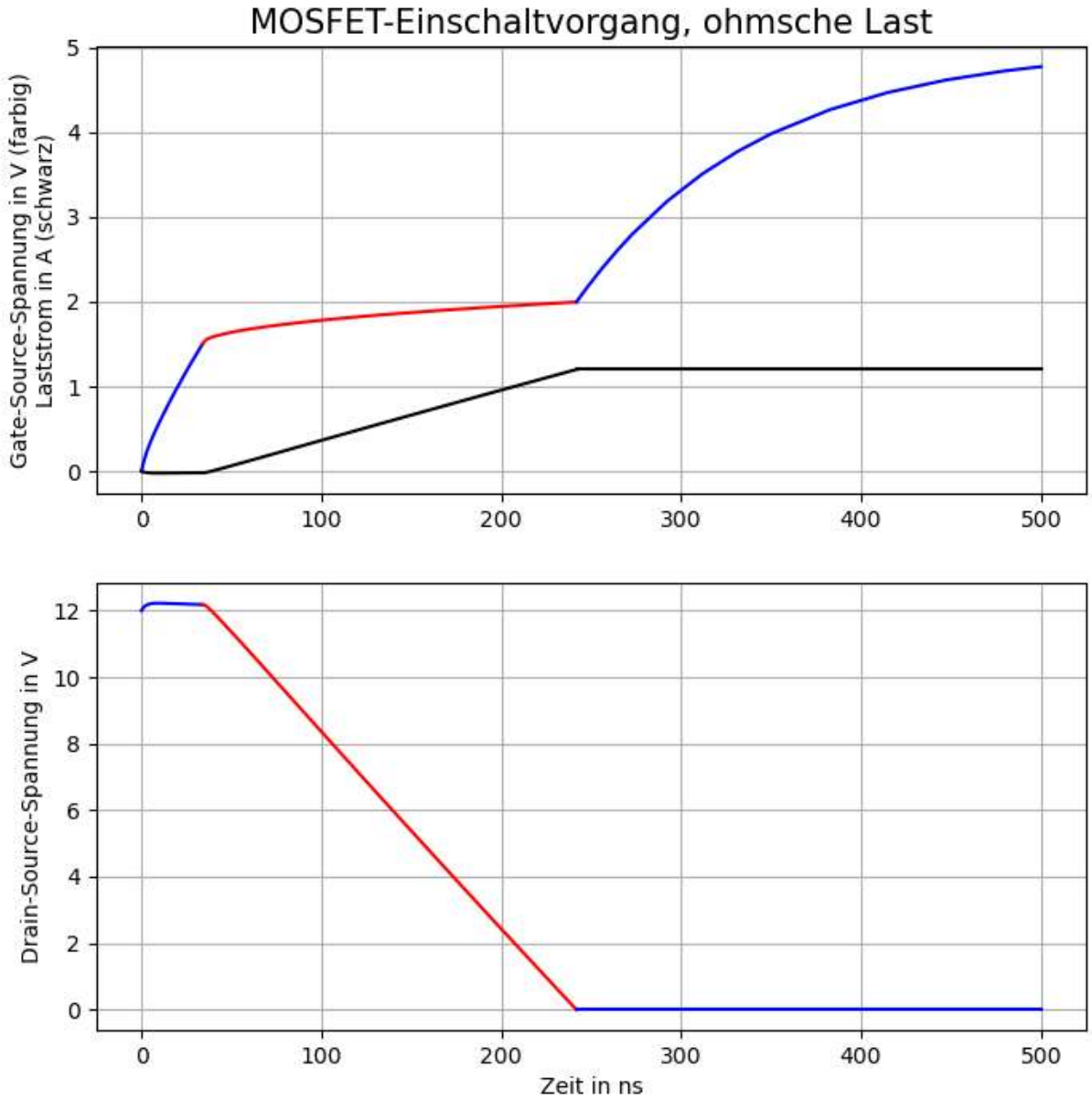
Simulation eines MOSFET-Einschaltvorgangs.

```
"""Auf der Lastseite des MOSFETs ist die Spannungsquelle UBAT über einen
Lastwiderstand RL mit dem Drain-Anschluss verbunden. Zum Einschalten des
MOSFETs wird die Spannung UE über einen Vorwiderstand RE an dessen Gate
gelegt. Die Drain-Gate-Kapatität CDG (Miller-Kapazität) wird bei dieser
Simulation ebenso berücksichtigt wie die Gate-Source-Kapazität CGS.

Während des Einschaltvorgangs werden der Reihe nach die folgenden drei
Zustände durchlaufen:

1. Die Spannung UE ist eingeschaltet, die Gate-Source-Spannung steigt an,
   es fließt aber noch kein Laststrom durch den MOSFET.

2. Die Gate-Source-Spanung des MOSFETs hat den Schwellenwert UTH
   überschritten, der Laststrom beginnt zu fließen und steigt weiter an.

3. Der Laststrom hat den Endwert UBAT / RL erreicht und bleibt konstant.
   Die Gate-Source-Spannung steigt weiter an und nähert sich asymptotisch
   der Eingangsspannung UE.

Tilman Küpper, https://kuepper.userweb.mwn.de, 2022-05-26
"""

import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

class MosfetSimul:
    """Klasse zur Simulation eines MOSFET-Einschaltvorgangs."""

    def set_defaults(self):
        """Setze alle Berechnungsparameter auf Default-Werte."""
        self.UE = 5             # Spannung am Gate-Vorwiderstand (in V)
        self.UBAT = 12          # Versorgungsspannung im Laststromkreis (in V)
        self.RE = 100           # Gate-Vorwiderstand (in Ohm)
        self.RL = 10            # Lastwiderstand (in Ohm)
        self.UTH = 1.5          # Threshold-Spannung (in V)
        self.K = 5              # Verstärkungsfaktor, MOSFET
        self.TSTOP = 500e-9     # Stopp-Zeitpunkt für Simulation (in s)
        self.CGS = 500e-12      # Gate-Source-Kapazität (in F)
        self.CDG = 500e-12      # Drain-Gate-Kapazität (Miller-Kapazität, in F)
        self.UGS0 = 0           # Spannung an CGS, Startwert (in V)
        self.UDG0 = self.UBAT   # Spannung an CDG, Startwert (in V)

    def __init__(self):
        """Konstruktor, setze Berechnungsparameter auf Default-Werte."""
        self.set_defaults()

    def _ode1(self, _t, y):
        """DGL für Zustand 1, UE eingeschaltet, noch kein Laststrom."""
        ugs, udg = y[0], y[1]
        ie = (self.UE - ugs) / self.RE
        il = (self.UBAT - ugs - udg) / self.RL
        ddt_ugs = (ie + il) / self.CGS
        ddt_udg = il / self.CDG
        return [ddt_ugs, ddt_udg]

    def _event1(self, _t, y):
        """Verlasse Zustand 1, sobald Gate-Source-Spannung > UTH ist."""
        ugs = y[0]
        return ugs - self.UTH

    def _load1(self, _t, y):
        """Berechne den Laststrom in Zustand 1."""
        ugs, udg = y[0], y[1]
        il = (self.UBAT - ugs - udg) / self.RL
        return il
```

```
def _ode2(self, _t, y):
    """DGL für Zustand 2, Gate-Source-Spannung > UTH, Laststrom steigt."""
    ugs, udg = y[0], y[1]
    id_int = self.K * (ugs - self.UTH) ** 2
    ie = (self.UE - ugs) / self.RE
    il = (self.UBAT - ugs - udg) / self.RL
    ddt_ugs = (ie + il - id_int) / self.CGS
    ddt_udg = (il - id_int) / self.CDG
    return [ddt_ugs, ddt_udg]

def _event2(self, _t, y):
    """Verlasse Zustand 2, wenn Laststrom den Wert UBAT / RL erreicht."""
    ugs, udg = y[0], y[1]
    il = (self.UBAT - ugs - udg) / self.RL
    return self.UBAT - self.RL * il

def _load2(self, _t, y):
    """Berechne den Laststrom in Zustand 2."""
    ugs, udg = y[0], y[1]
    il = (self.UBAT - ugs - udg) / self.RL
    return il

def _ode3(self, _t, y):
    """DGL für Zustand 3, Laststrom hat seinen Endwert erreicht."""
    ugs = y[0]
    ie = (self.UE - ugs) / self.RE
    ddt_ugs = ie / (self.CGS + self.CDG)
    ddt_udg = -ddt_ugs
    return [ddt_ugs, ddt_udg]

def _load3(self, t, _y):
    """Berechne den Laststrom in Zustand 3."""
    return t * 0 + self.UBAT / self.RL

def do_simul(self):
    """Führe MOSFET-Simulation durch, plotte die Ergebnisse."""

    # MOSFET im Zustand 1, noch kein Laststrom
    def evt1(t, y):
        return self._event1(t, y)

    evt1.terminal = True
    sol1 = solve_ivp(
        self._ode1,
        [0, self.TSTOP],
        [self.UGS0, self.UDG0],
        events=evt1,
        method="BDF",
    )

    # Letzte Werte von t, ugs, udg sind Startwerte für folgenden Zustand
    t1, ugs1, udg1 = sol1.t[-1], (sol1.y[0])[-1], (sol1.y[1])[-1]
    print(f"t1: {t1*1e9:.2f} ns, ugs1: {ugs1:.2f} V, udg1: {udg1:.2f} V")

    # MOSFET im Zustand 2, Laststrom steigt
    def evt2(t, y):
        return self._event2(t, y)

    evt2.terminal = True
    sol2 = solve_ivp(
        self._ode2,
        [t1, self.TSTOP],
        [ugs1, udg1],
        events=evt2,
        method="BDF"
    )
```

```
        # Letzte Werte von t, ugs, udg sind Startwerte für folgenden Zustand
        t2, ugs2, udg2 = sol2.t[-1], (sol2.y[0])[-1], (sol2.y[1])[-1]
        print(f"t2: {t2*1e9:.2f} ns, ugs2: {ugs2:.2f} V, udg2: {udg2:.2f} V")

        # MOSFET im Zustand 3, Laststrom hat Endwert erreicht
        sol3 = solve_ivp(
            self._ode3,
            [t2, self.TSTOP],
            [ugs2, udg2],
            method="BDF"
        )

        # Zeichne oberes Diagramm, ugs (farbig) und il (schwarz)
        fig, (a1, a2) = plt.subplots(2, 1)
        fig.set_size_inches(8, 8)

        a1.plot(sol1.t * 1e9, sol1.y[0], "b-")
        a1.plot(sol2.t * 1e9, sol2.y[0], "r-")
        a1.plot(sol3.t * 1e9, sol3.y[0], "b-")
        a1.plot(sol1.t * 1e9, self._load1(sol1.t, sol1.y), "k-")
        a1.plot(sol2.t * 1e9, self._load2(sol2.t, sol2.y), "k-")
        a1.plot(sol3.t * 1e9, self._load3(sol3.t, sol3.y), "k-")
        a1.grid(True)
        a1.set_title("MOSFET-Einschaltvorgang, ohmsche Last", fontsize=15)
        a1.set_ylabel(
            "Gate-Source-Spannung in V (farbig)\nLaststrom in A (schwarz)"
        )

        # Zeichne unteres Diagramm, uds (farbig)
        uds1 = self.UBAT - self.RL * self._load1(sol1.t, sol1.y)
        uds2 = self.UBAT - self.RL * self._load2(sol2.t, sol2.y)
        uds3 = self.UBAT - self.RL * self._load3(sol3.t, sol3.y)
        a2.plot(sol1.t * 1e9, uds1, "b-")
        a2.plot(sol2.t * 1e9, uds2, "r-")
        a2.plot(sol3.t * 1e9, uds3, "b-")
        a2.grid(True)
        a2.set_xlabel("Zeit in ns")
        a2.set_ylabel("Drain-Source-Spannung in V")

        plt.savefig("mosfet_simul.png")
        plt.show()


# Führe MOSFET-Simulation durch, plotte die Ergebnisse.
if __name__ == "__main__":
    simul = MosfetSimul()
    simul.do_simul()
```


MOSFET-Einschaltvorgang, ohmsche Last
Gate-Source-Spannung in V (farbig)
Laststrom in A (schwarz)
0
1
2
3
4
5
0
100
200
300
400
500
Drain-Source-Spannung in V
0
2
4
6
8
10
12
0
100
200
300
400
500
Zeit in ns