Project #1\buck.m

```matlab
% ID Number: 229,506
% ECE 31033 - Project #1
% buck.m

% The file (buck.m) contains the Forward Euler integration algorithm within a while
% loop (FOR LOOPS ARE NOT ALLOWED). buck is not a function. The file buck.m only
% contains a single while loop (i.e. while (t(k)<tend)) to solve for all circuit voltages
% and currents of your buck converter. Within the while loop, you will call the
% function sw at each time instant to determine the value of your transistor gate
% (on or off).  Voltages of currents and voltages of circuit components must be
% determined within the while loop.
while t_vec(k) < tend
    if (ideal_boolean) % If the circuit is ideal.
        switch_state(k) = sw(D, t_vec(k)); % calling sw.m

        % Inductor Current and Load Voltage Calculation
        i_L_vec(k+1) = i_L_vec(k) + dt * ((switch_state(k)) * V_in - V_load_vec(k)) / L; %i_L
        V_load_vec(k+1) = V_load_vec(k) + dt * ((i_L_vec(k) - (V_load_vec(k) / R_load)) / C); %V_load

        % Switch 1 and 2: Voltage and Current Calculations
        if(switch_state(k))
            V_switch1(k+1) = V_in;
            i_switch1(k+1) = i_L_vec(k+1) * switch_state(k);

            V_switch2(k+1) = 0;
            i_switch2(k+1) = 0;

            V_L_vec(k+1) = V_in - V_load_avg;
        else
            V_switch1(k+1) = 0;
            i_switch1(k+1) = 0;

            V_switch2(k+1) = -1 * V_in;
            i_switch2(k+1) = i_L_vec(k) * (1 - switch_state(k));

            V_L_vec(k+1) = -1 * V_load_avg;
        end

        % Capacitor: Voltage and Current Calculations
        i_C_vec(k+1) = i_L_vec(k) - (V_load_vec(k) / R_load);
        V_C_vec(k+1) = V_load_vec(k+1);

        % Load: Current Calculation
        i_load_vec(k+1) = V_load_vec(k+1) / R_load;

    else % If the circuit is non ideal.
        switch_state(k) = sw(D_non_ideal, t_vec(k)); % calling sw.m

        if(switch_state(k))
            %i_L_vec(k+1) = i_L_vec(k) + dt * ((V_in - V_T_on - V_load_vec(k) - (R_T_on * i_L_vec(k)))) / L;    %i_L
            i_L_vec(k+1) = i_L_vec(k) + dt * ((V_in - V_T_on - V_load_vec(k) - (R_T_on * i_L_vec(k))) / L);    %i_L

            % Switch 1: Voltage and Current Calculations
            V_switch1(k+1) = 0;
            i_switch1(k+1) = i_L_vec(k+1);
            P_switch1(k+1) = (R_T_on * i_L_vec(k+1) + V_T_on) * i_L_vec(k+1);

            % Switch 2: Voltage and Current Calculations
            V_switch2(k+1) = V_D_on + (R_D_on * i_L_vec(k+1)) - V_in;
            i_switch2(k+1) = 0;
        else
            i_L_vec(k+1) = i_L_vec(k) + dt * ((-1 * V_load_vec(k) - V_D_on - (R_D_on * i_L_vec(k)))) / L;

            % Switch 1: Voltage and Current Calculations
            V_switch1(k+1) = V_in + (R_D_on * i_L_vec(k+1)) + V_D_on;
            i_switch1(k+1) = 0;

            % Switch 2: Voltage and Current Calculations
            V_switch2(k+1) = 0;
            i_switch2(k+1) = i_L_vec(k+1);
            P_switch2(k+1) = (R_D_on * i_L_vec(k+1) + V_D_on) * i_L_vec(k+1);
        end

        V_load_vec(k+1) = V_load_vec(k) + dt * ((i_L_vec(k) - (V_load_vec(k) / R_load)) / C); %V_load
    end

    % Increment the time and index
    t_vec(k + 1) = t_vec(k) + dt;
    k = k + 1;
end
```