

Converter System Modeling via MATLAB/Simulink

A powerful environment for system modeling and simulation

MATLAB: programming and scripting environment

Simulink: block diagram modeling environment that runs inside MATLAB

Things we can achieve, relative to Spice:

- Higher level of abstraction, suitable for higher-level system models
- More sophisticated controller models
- Arbitrary system elements

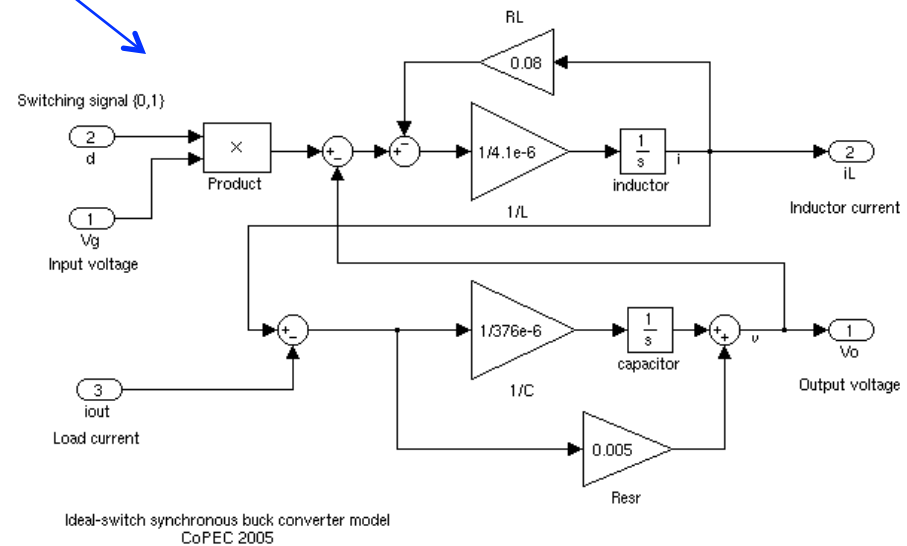
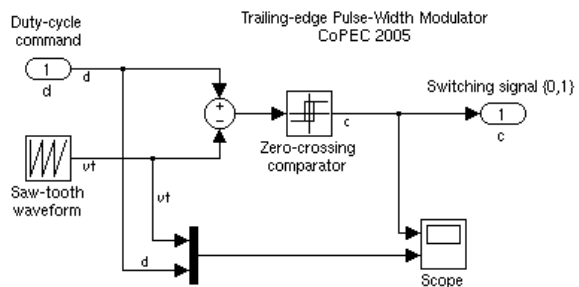
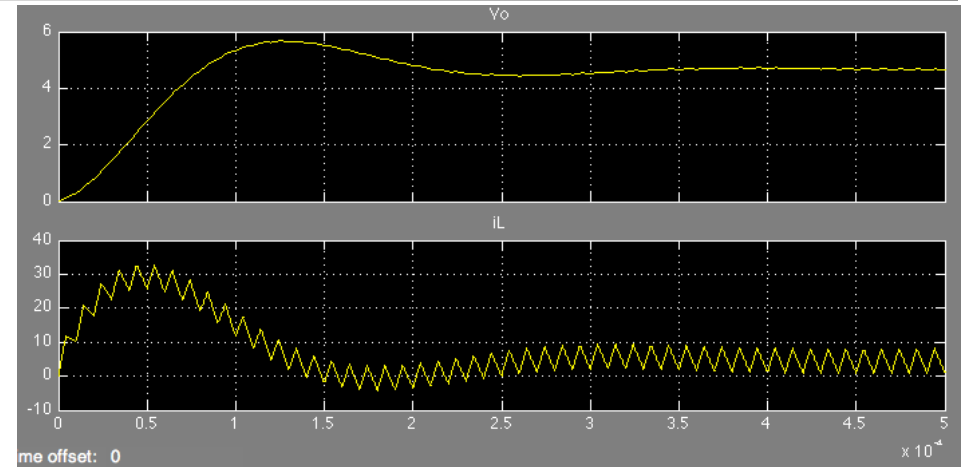
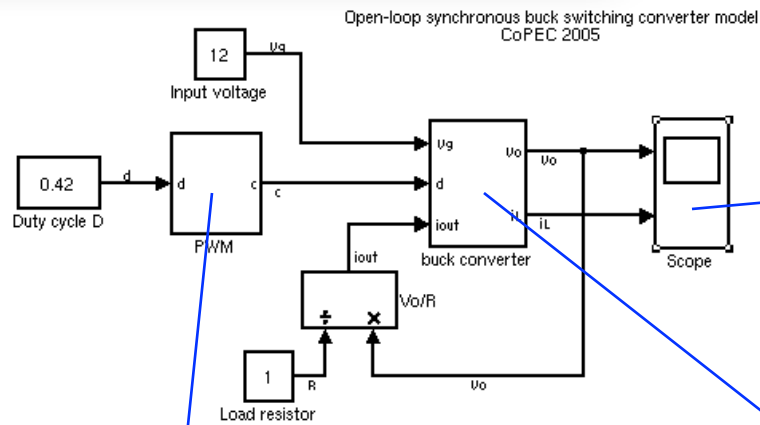
But:

- We have to derive our own mathematical models
- Simulink signals are unidirectional as in conventional block diagrams

At CoPEC, nearly all simulation is done within MATLAB/Simulink

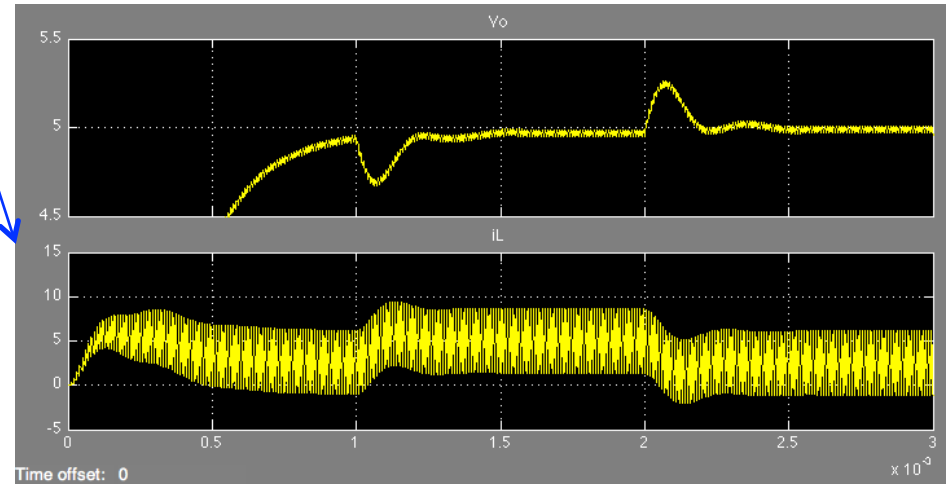
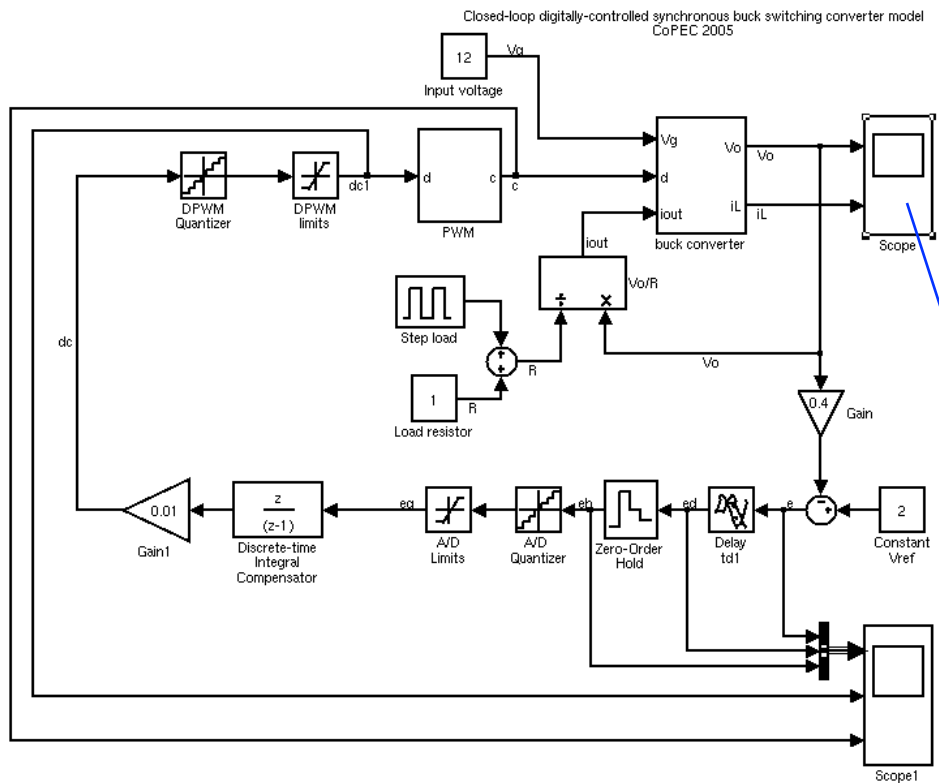
Open-loop buck converter

Time domain simulation including switching ripple



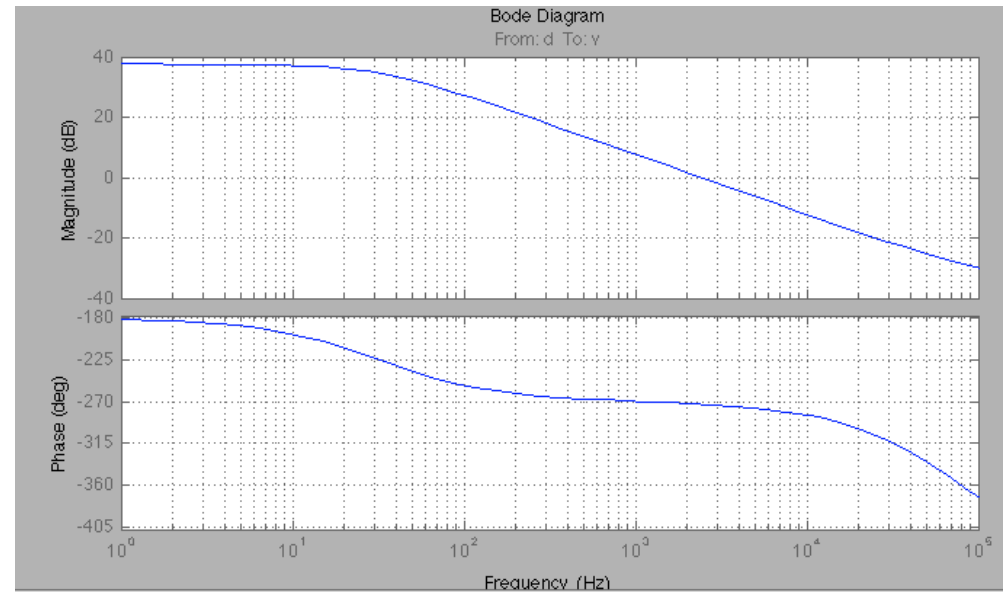
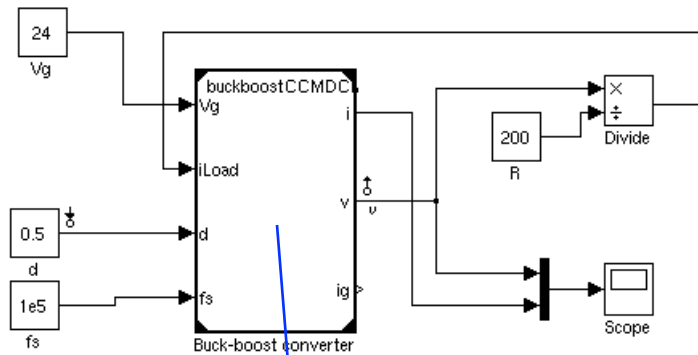
Closed-loop buck converter, digital control

Time domain simulation with switching ripple

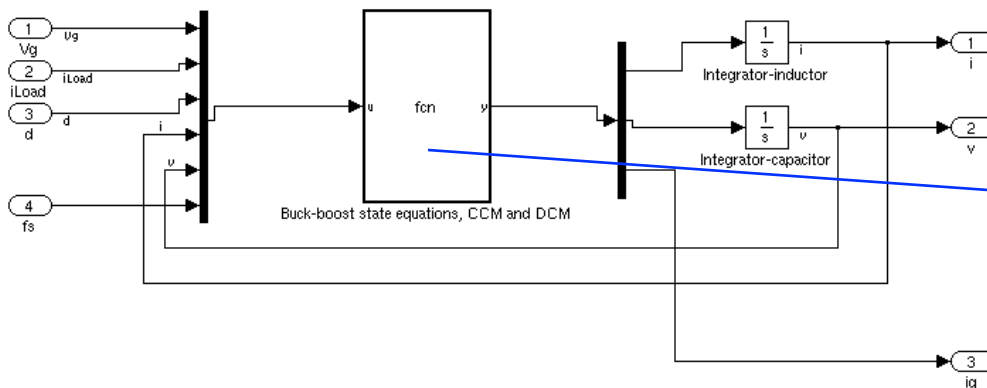


Open-loop buck-boost converter

Frequency domain simulation, averaged model



Control-to-output transfer function

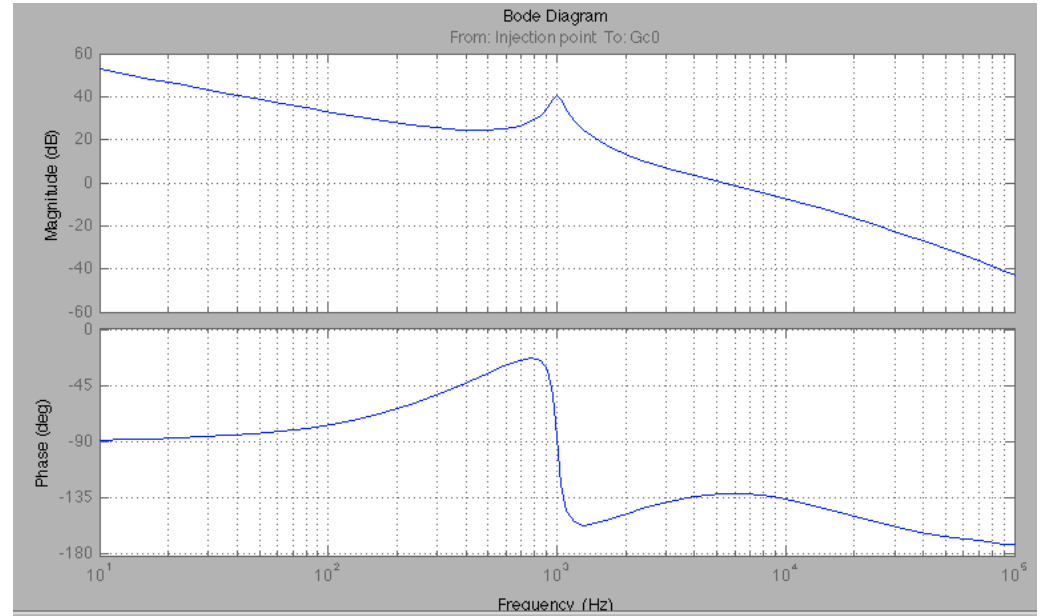
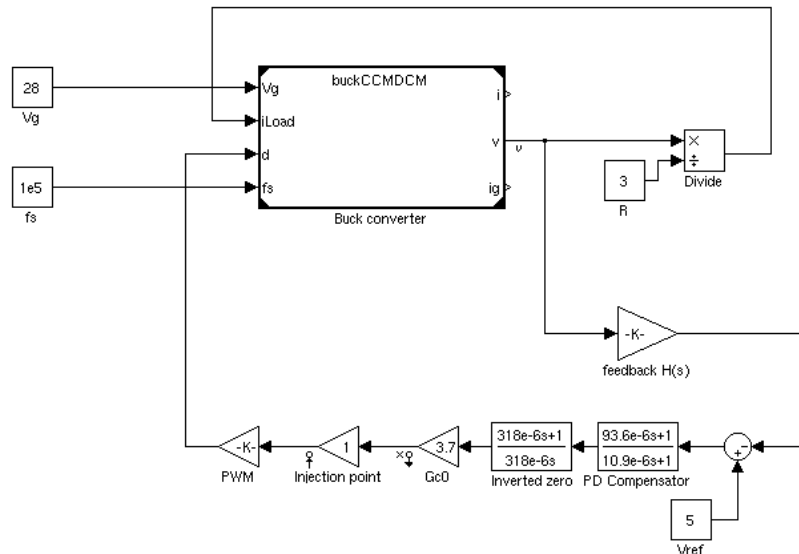


```

1 function y = fcn(u)
2 %#eml
3 % buck-boost converter in CCM or DCM, state equations
4 fs = u(6);
5 v = u(5);
6 Vg = u(1);
7 iLoad = u(2);
8 d = u(3);
9 i = u(4);
10 L = 100e-6;
11 C = 50e-6;
12 d2 = min(1-d, i*2*L*fs/(Vg*d)-d);
13 vL = d*Vg + d2*v;
14 iC = - i*d2/(d+d2) - iLoad;
15 ig = i*d/(d+d2);
16 y = [vL/L iC/C ig];
    
```

Closed-loop buck converter

Frequency domain simulation, averaged model



Loop gain: Bode plot

MATLAB/Simulink discussion

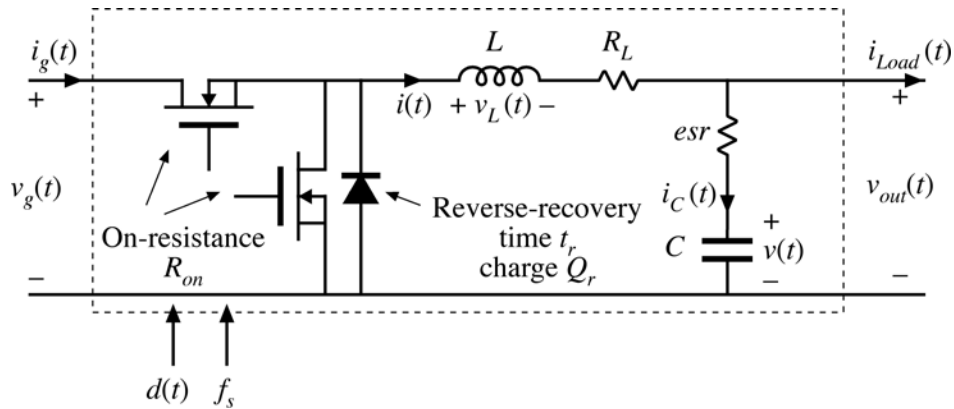
- A structured way to write the converter averaged equations, suitable for implementation in Simulink:

State-space averaging

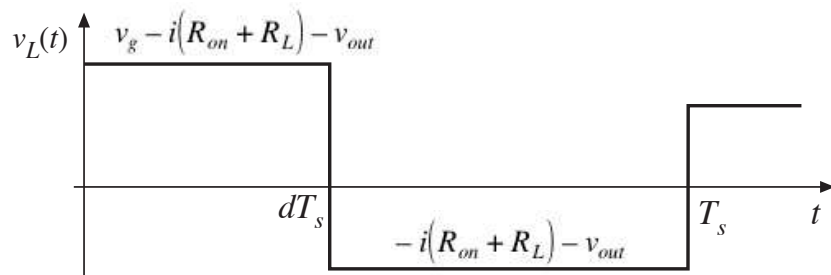
- Some basic converter models, implemented in Simulink
- How to plot small-signal transfer functions in Simulink
- Modeling the discontinuous conduction mode

Synchronous buck converter

Formulating state equations for Simulink model



Averaging the inductor voltage

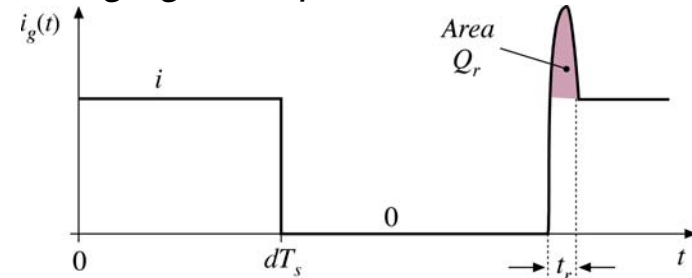


$$\langle v_L \rangle = d \left[v_g - i(R_{on} + R_L) - v_{out} \right] + d' \left[-i(R_{on} + R_L) - v_{out} \right]$$

with $v_{out} = v + (i - i_{Load}) esr$

so $\langle v_L \rangle = dv_g - i(R_{on} + R_L) - v - (i - i_{Load}) esr$

Averaging the input current



$$\langle i_g \rangle = di_L + \frac{t_r}{T_s} i_L + \frac{Q_r}{T_s}$$

Averaging the capacitor current:

For both intervals,

$$i_C = i - i_{Load}$$

Resulting state equations:

$$\langle i_g \rangle = di_L + \frac{t_r}{T_s} i_L + \frac{Q_r}{T_s}$$

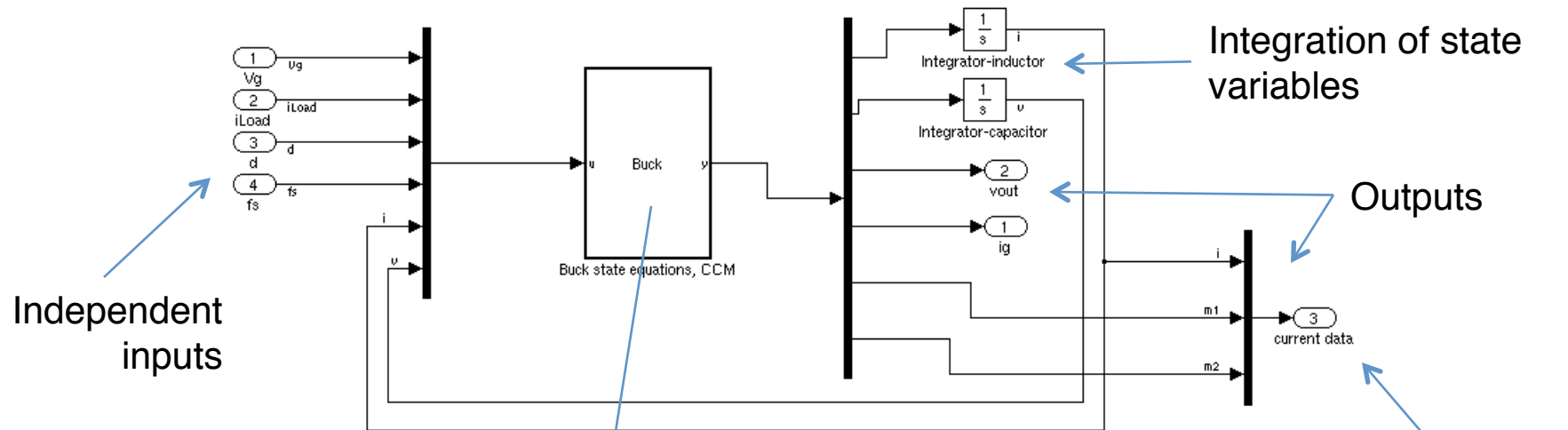
$$L \frac{di}{dt} = \langle v_L \rangle = dv_g - i(R_{on} + R_L) - v - (i - i_{Load}) esr$$

$$C \frac{dv}{dt} = \langle i_C \rangle = i - i_{Load}$$

$$v_{out} = v + (i - i_{Load}) esr$$

Basic buck converter model

Averaged model for Simulink



Embedded MATLAB code block:

- Load inputs from u vector
- Set circuit parameters
- Calculate state equations and outputs
- Place results in output y vector

```

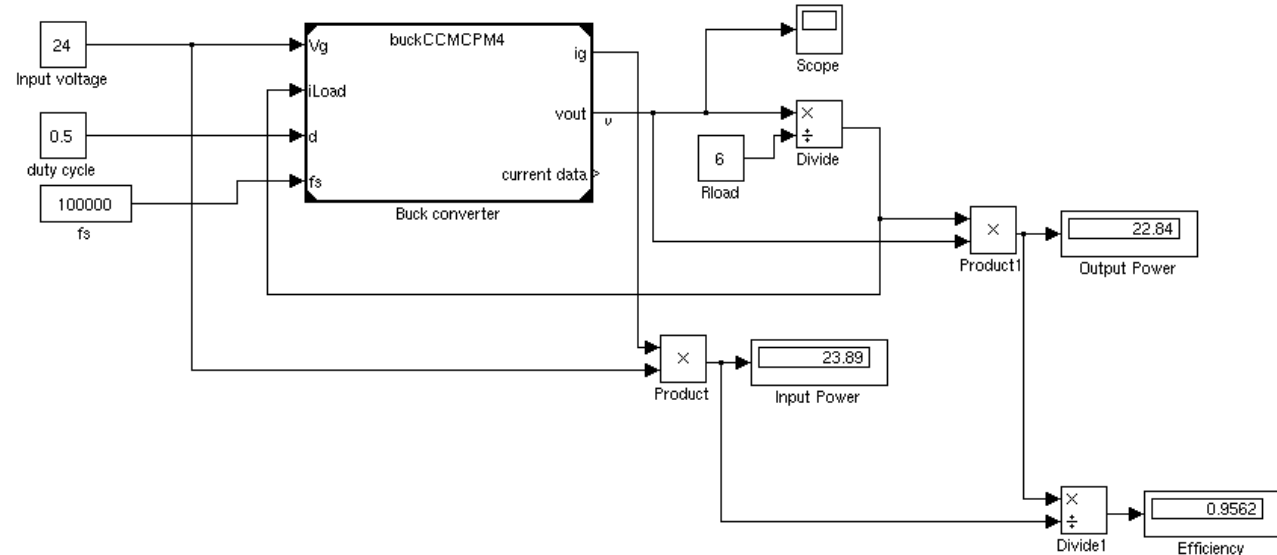
1 function y = Buck(u)
2 %#eml
3 % buck converter in CCM, including basic loss model
4 % Signal inputs
5 v = u(6); % capacitor voltage
6 Vg = u(1); % input voltage
7 iLoad = u(2); % output current
8 d = u(3); % duty cycle
9 i = u(5); % inductor current
10 % Circuit parameters
11 fs = u(4); % switching frequency
12 L = 100e-6; % inductance
13 C = 50e-6; % output capacitance
14 esr = 0.05; % capacitor ESR
15 Ron = 0.05; % MOSFET on resistance
16 RL = 0.1; % inductor dc resistance
17 tr = 50e-9; % diode reverse recovery time
18 Qr = 100e-9; % diode recovered charge
19 % Calculation of inductor current slopes
20 m1 = (Vg - v - i*(Ron+RL+esr) + iLoad*esr)/L; % up-slope, during transistor conduction
21 m2 = (v + i*(Ron+RL+esr) - iLoad*esr)/L; % - down-slope, during diode conduction
22 % Calculation of state equations
23 vL = d*Vg - v - i*(Ron+RL+esr) + iLoad*esr; % average inductor voltage
24 iC = i - iLoad; % average capacitor current
25 vout = v + i*esr - iLoad*esr; % converter output voltage
26 ig = i*d + tr*i*fs + Qr*fs; % converter input current
27 % Output results
28 y = [vL/L iC/C vout ig m1 m2];
  
```

(used in current mode control)

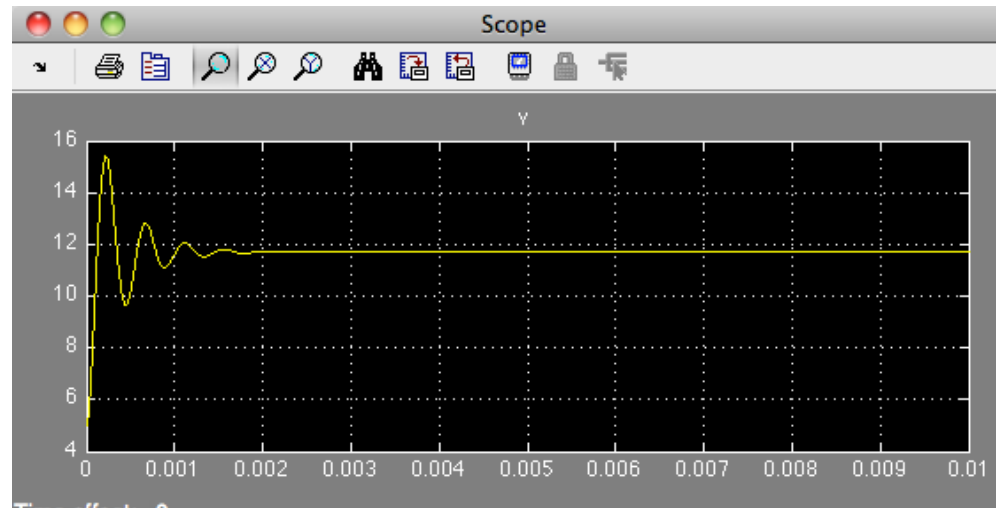
Time-domain simulation

Synchronous buck example, Simulink

Simulink model employing synchronous buck model, with voltage mode control

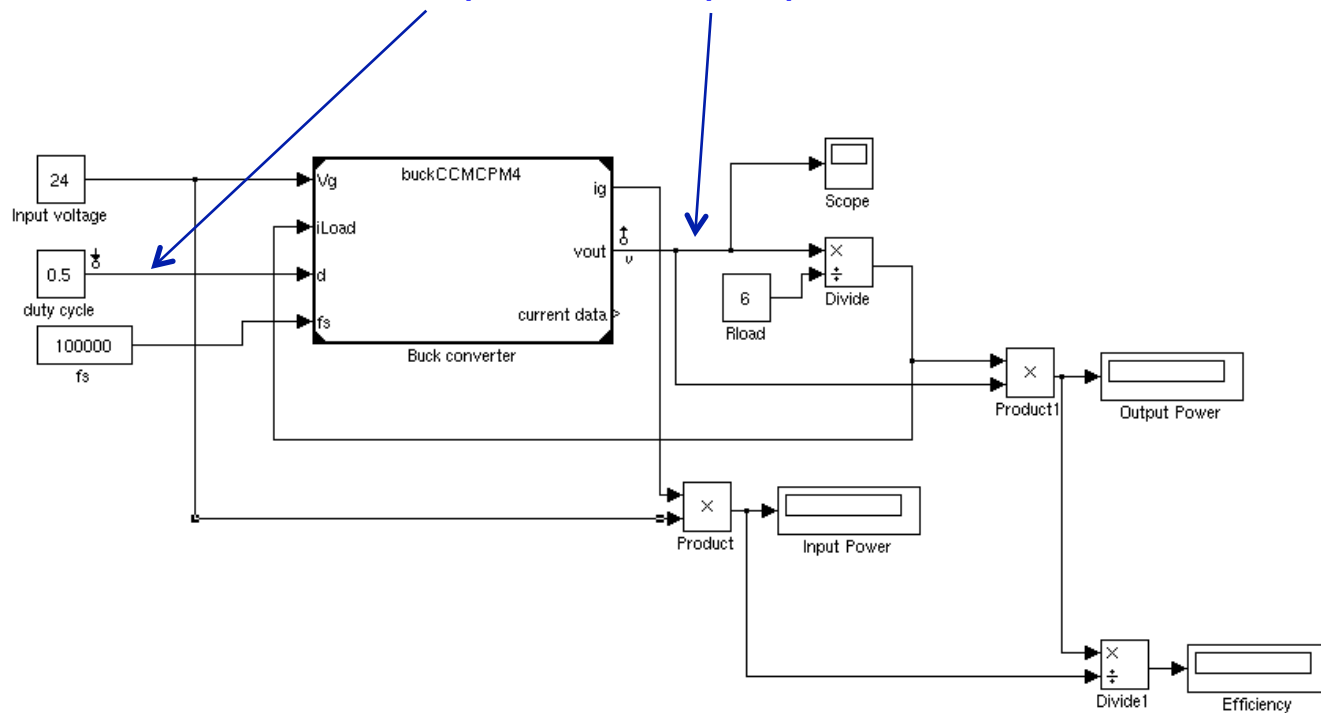


Output voltage transient response



Generating a Bode Plot from the Simulink file

1. Set transfer function input and output points



- Right-click on the desired wire
- Select “Linearization Points”, then “input point” or “output point”

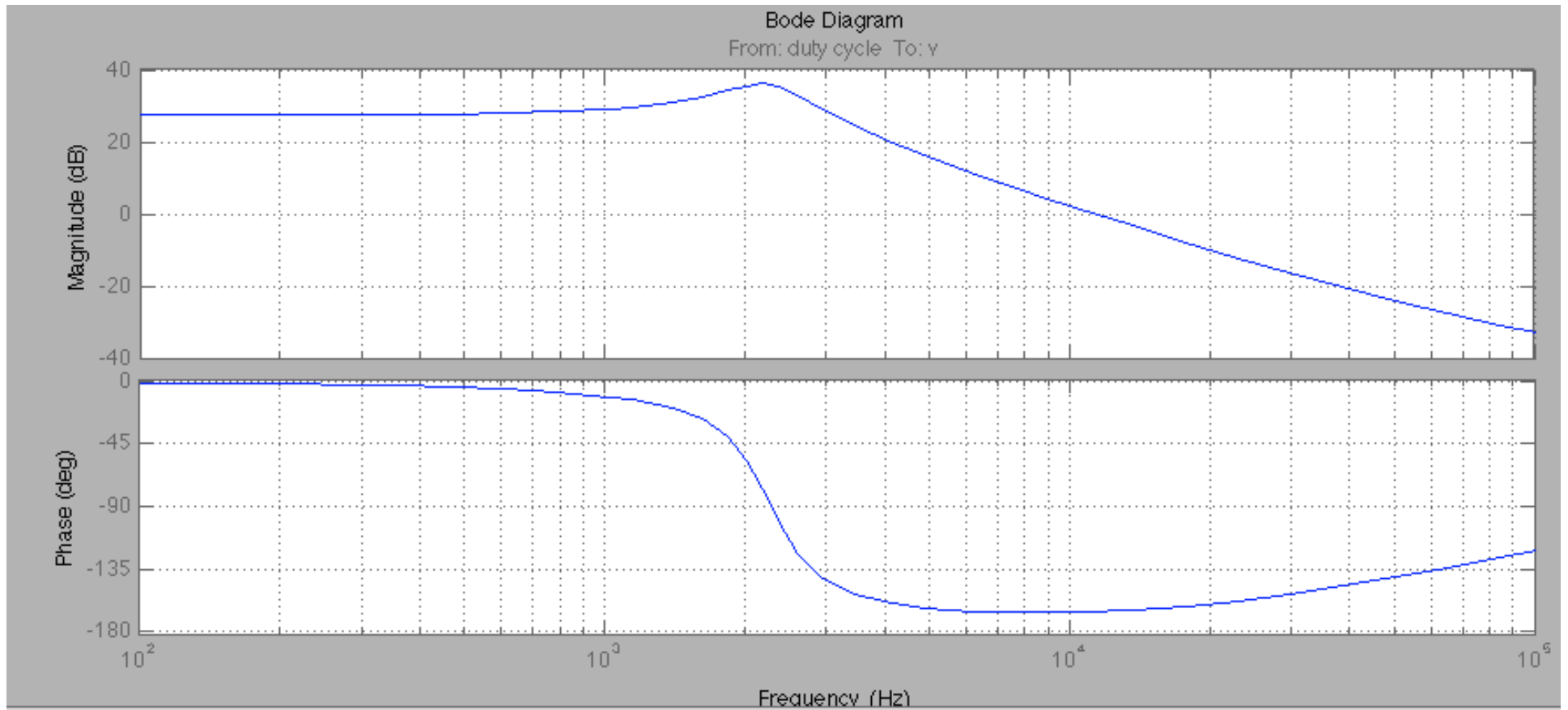
Generating a Bode Plot from Simulink, p. 2

```
%% Bode plotter using linearization tool
% requires simulink control design toolbox
mdl = 'buckCPM4Vmodetester'; % set to file name of simulink model. Must
    have i/o points set within this model
io = getlinio(mdl) % get i/o signals of mdl
op =operspec(mdl)
op = findop(mdl,op) % calculate model operating point
lin = linearize(mdl,op,io) % compute state space model of linearized
    system
ltiview(lin) % send linearized model to LTI Viewer tool
```

- Save this as a script (".m file") and run it whenever you want to generate a Bode plot
- This script finds the steady-state operating point and linearizes the model
- The last line opens the LTI Viewer tool, which generates various small-signal plots including Bode, step response, pole/zero, Nyquist, etc.

Control-to-output transfer function G_{vd}

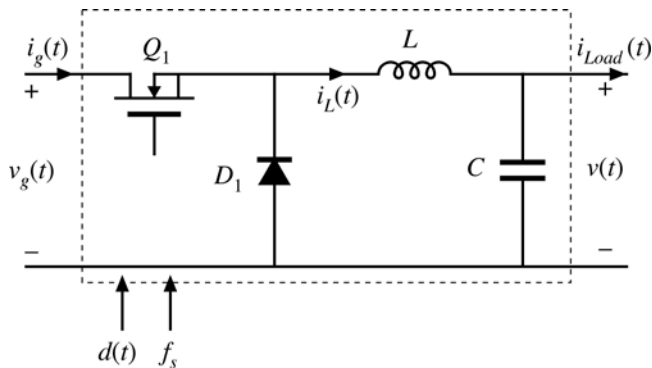
Generated by Simulink



Synchronous buck example of previous slides

Modeling DCM

Buck example, Simulink model

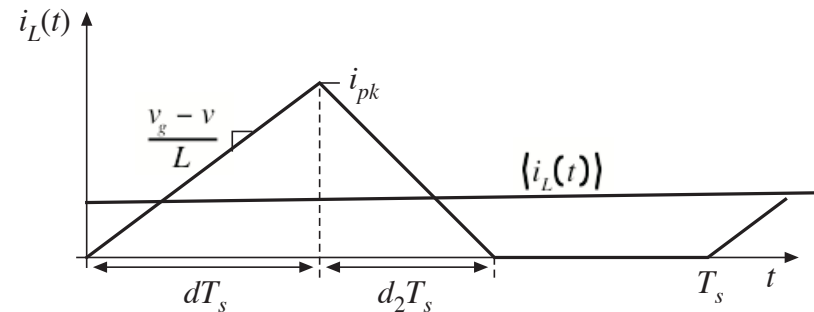


In DCM, the average inductor current can be expressed as:

$$\langle i_L(t) \rangle = \frac{1}{2} i_{pk} (d + d_2)$$

$$\text{with } i_{pk} = \frac{(v_g - v)}{L} d T_s$$

$$\text{so } \langle i_L(t) \rangle = \frac{1}{2} \frac{(v_g - v)}{L} d T_s (d + d_2)$$



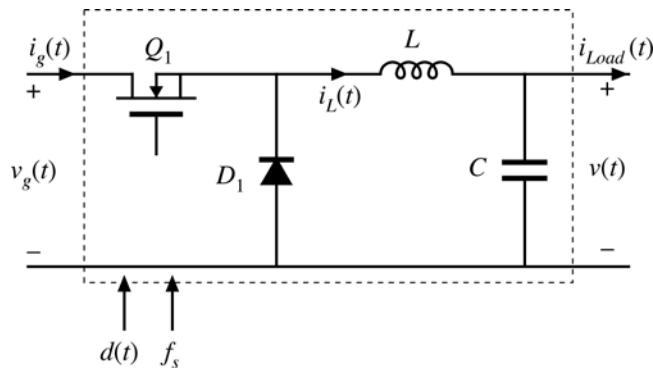
Treat the average inductor current as an independent state, and solve for d_2 :

$$d_2 = \frac{2L f_s \langle i_L(t) \rangle}{d(v_g - v)} - d$$

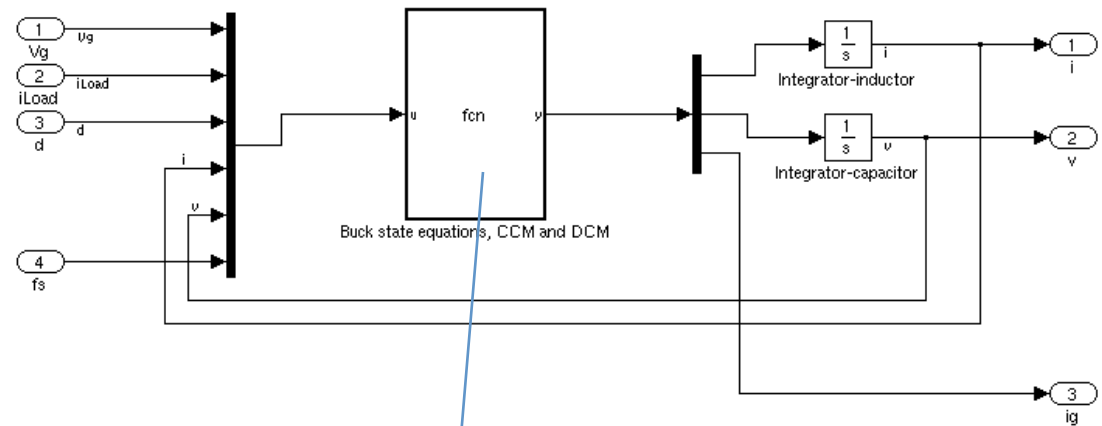
Note that $d_2 = 1 - d$ in CCM, and $d_2 < 1 - d$ in DCM

Combined CCM-DCM model

Buck converter



Simulink model



State equations:

$$\langle v_L(t) \rangle = d v_g - (d + d_2)v$$

$$\langle i_C(t) \rangle = i - i_{Load}$$

$$\langle i_g(t) \rangle = \frac{d}{d + d_2} i$$

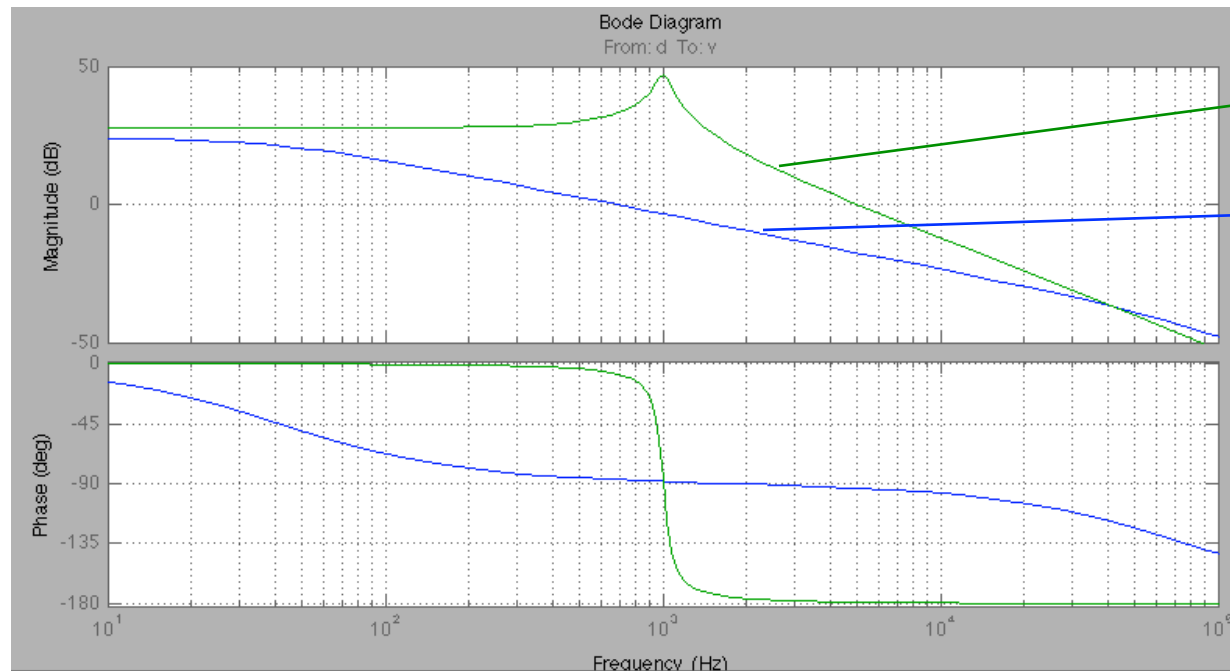
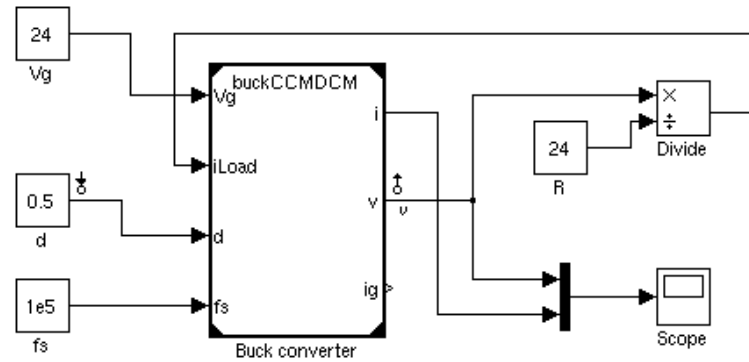
$$d_2 = \min \left(d', \frac{2L f_s i}{d(v_g - v)} - d \right)$$

```

1 function y = fcn(u)
2 %#eml
3 % buck converter state equations, CCM or DCM
4 v = u(5);
5 Vg = u(1);
6 iLoad = u(2);
7 d = u(3);
8 i = u(4);
9 fs = u(6);
10 L = 50e-6;
11 C = 500e-6;
12 d2 = min(1-d, 2*L*fs*i/((Vg-v)*d)-d);
13 vL = d*Vg - (d+d2)*v;
14 iC = i - iLoad;
15 ig = d*i/(d+d2);
16 y = [vL/L iC/C ig];
    
```

Buck control-to-output transfer function

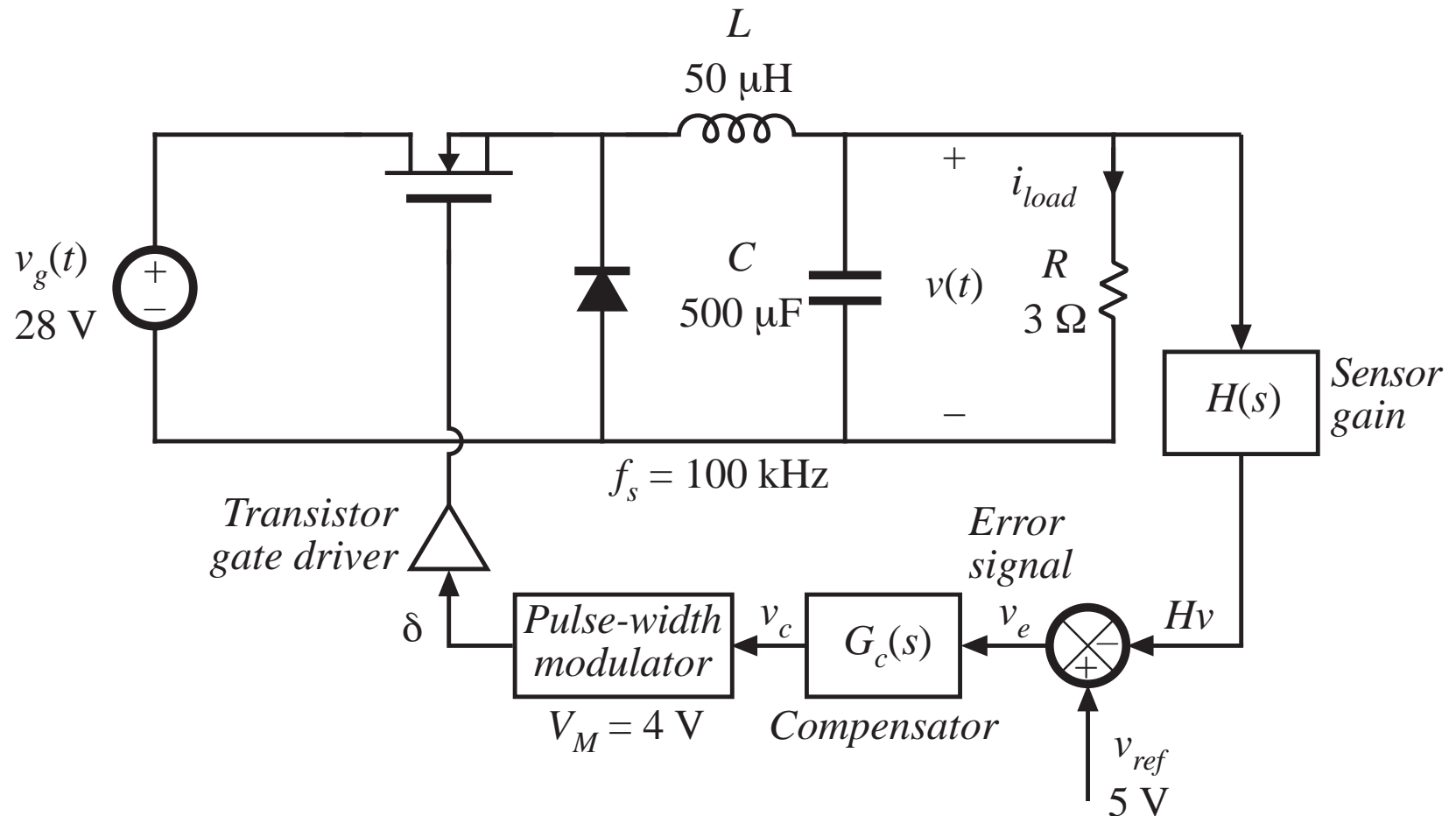
CCM vs. DCM



$R = 3\Omega$: CCM

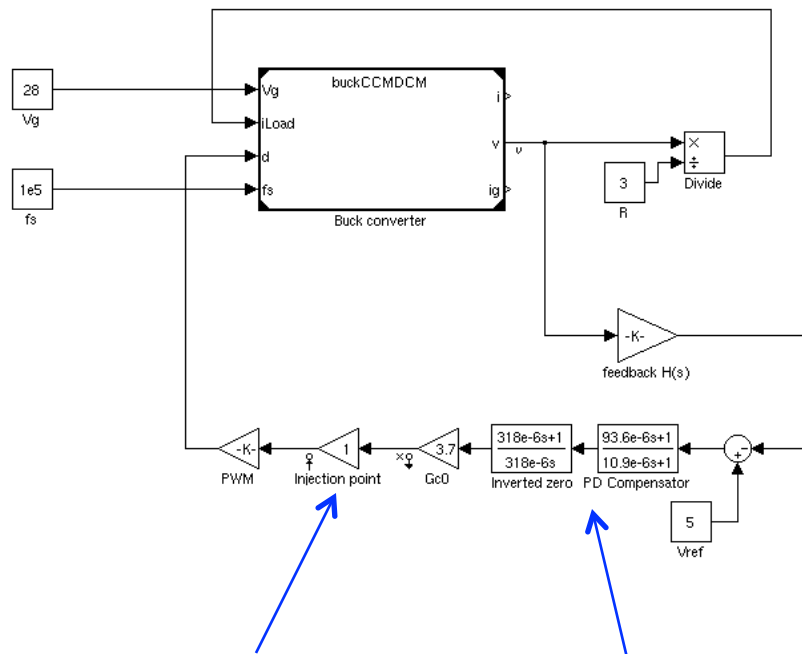
$R = 24\Omega$: DCM

9.5.4. Design example



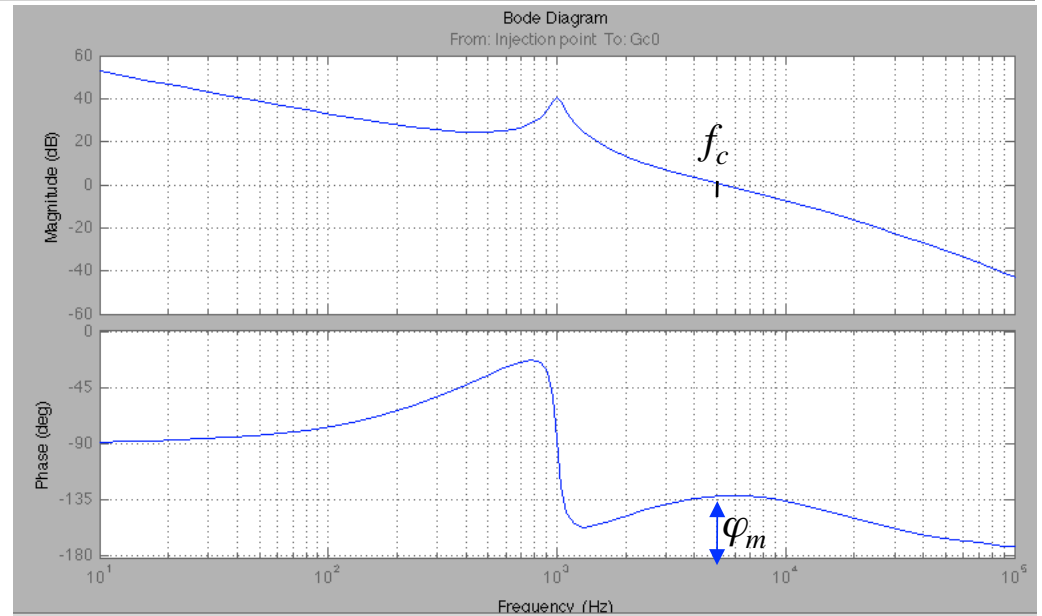
Closed-loop buck converter

Simulink frequency domain simulation, averaged model



Injection point for measurement of loop gain $T(s)$

Transfer function blocks:
Implementing the PID compensator

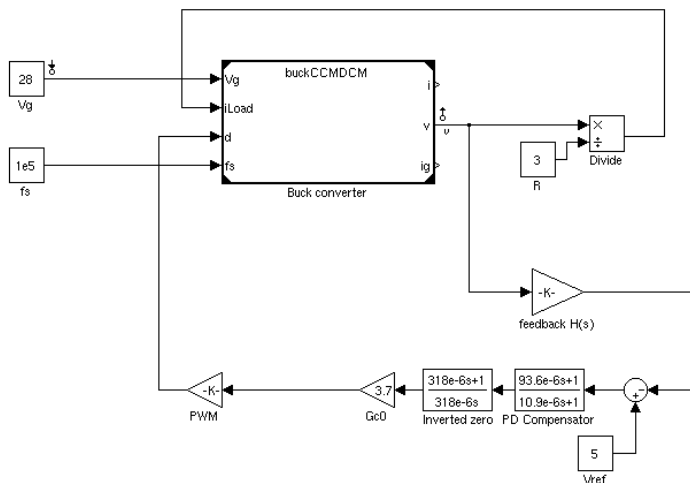


Loop gain: Bode plot

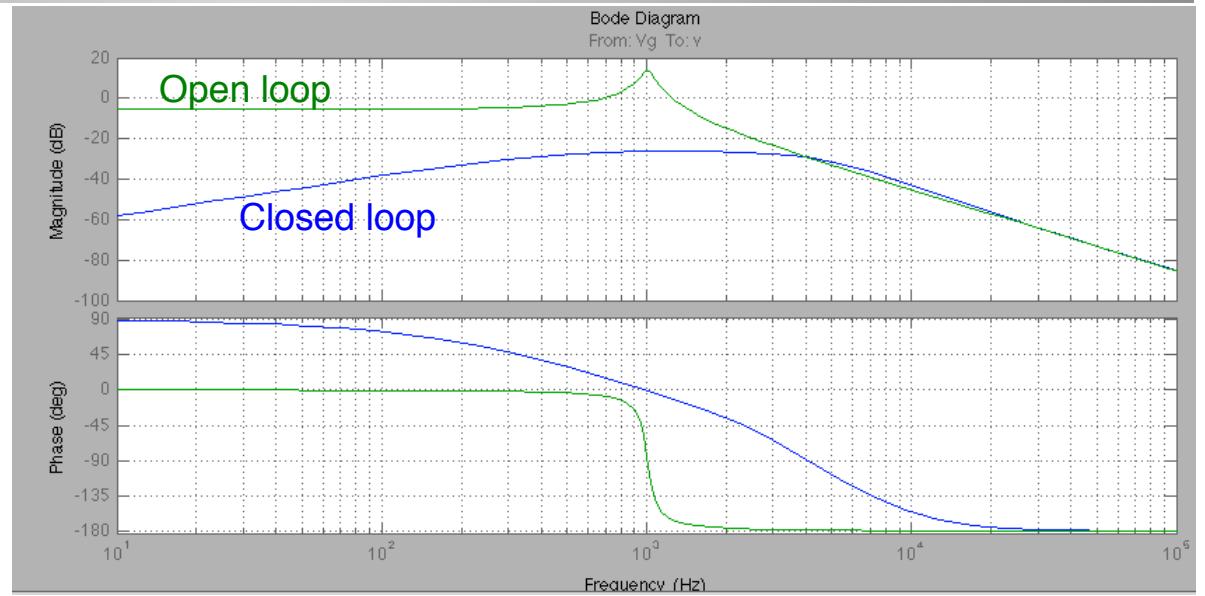
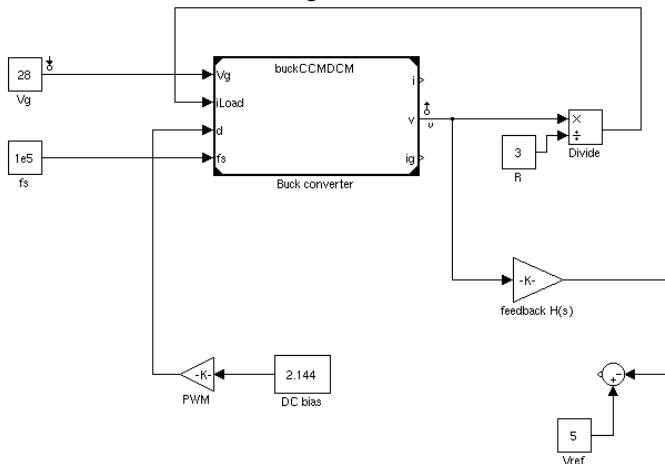
Closed-loop line-to-output transfer function

Simulink frequency domain simulation

Closed loop G_{vg}



Open loop G_{vg}



Script that generates both plots

```

1 %% Bode plotter using linearization tool
2 % requires simulink control design toolbox
3 mdl = 'BuckFeedbackExampleCh9PIDvg'; % set
4 io = getlinio mdl % get i/o signals of mdl
5 op =operspec mdl
6 op = findop mdl,op % calculate model oper
7 lin = linearize mdl,op,io % compute state
8 mdl = 'BuckFeedbackExampleCh9GvgOpenLoop';
9 io = getlinio mdl % get i/o signals of mdl
10 op =operspec mdl
11 op = findop mdl,op % calculate model oper
12 lino = linearize mdl,op,io % compute state
13 ltview lin,lino % send linearized model
    
```