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

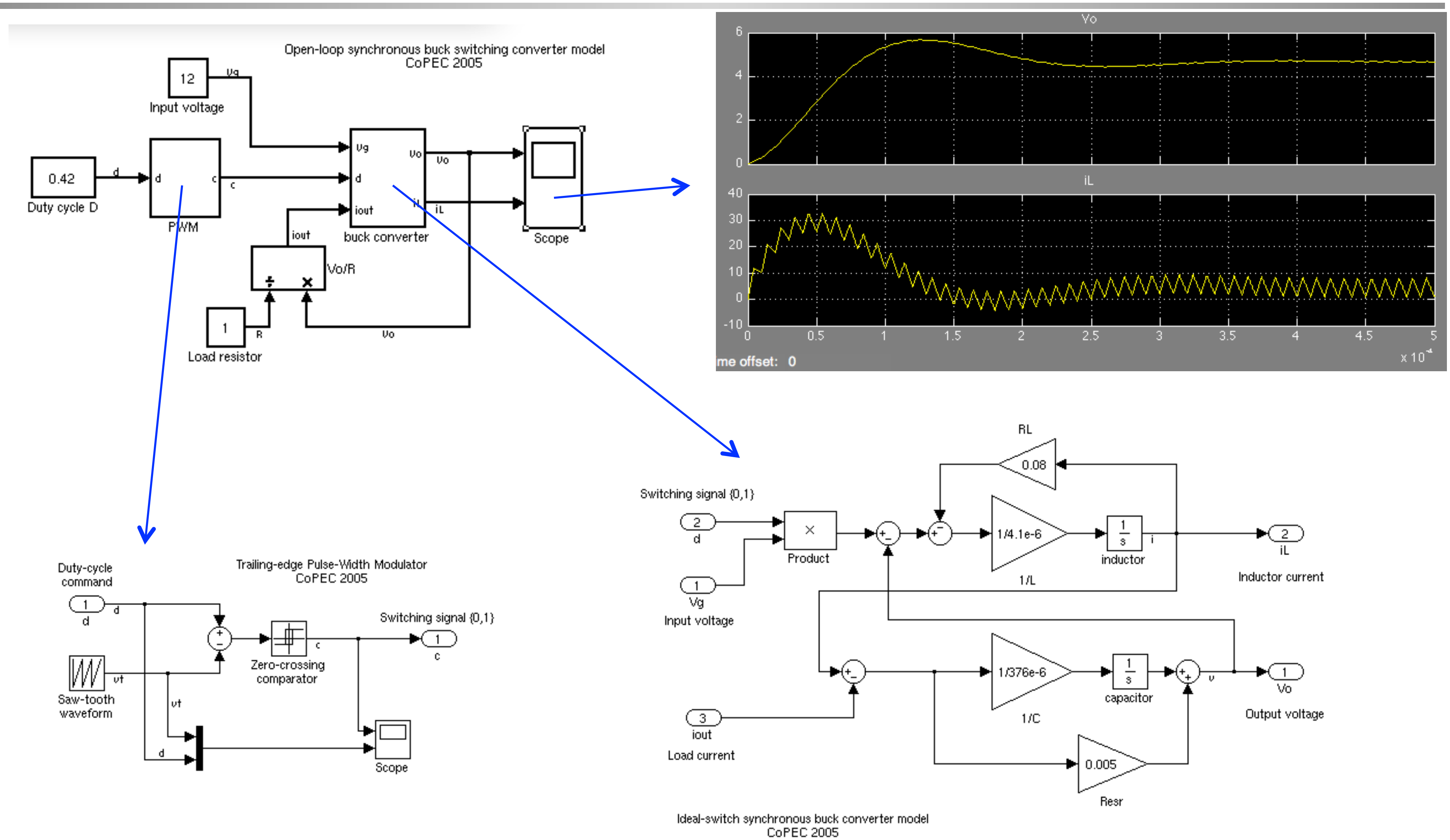
But:

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

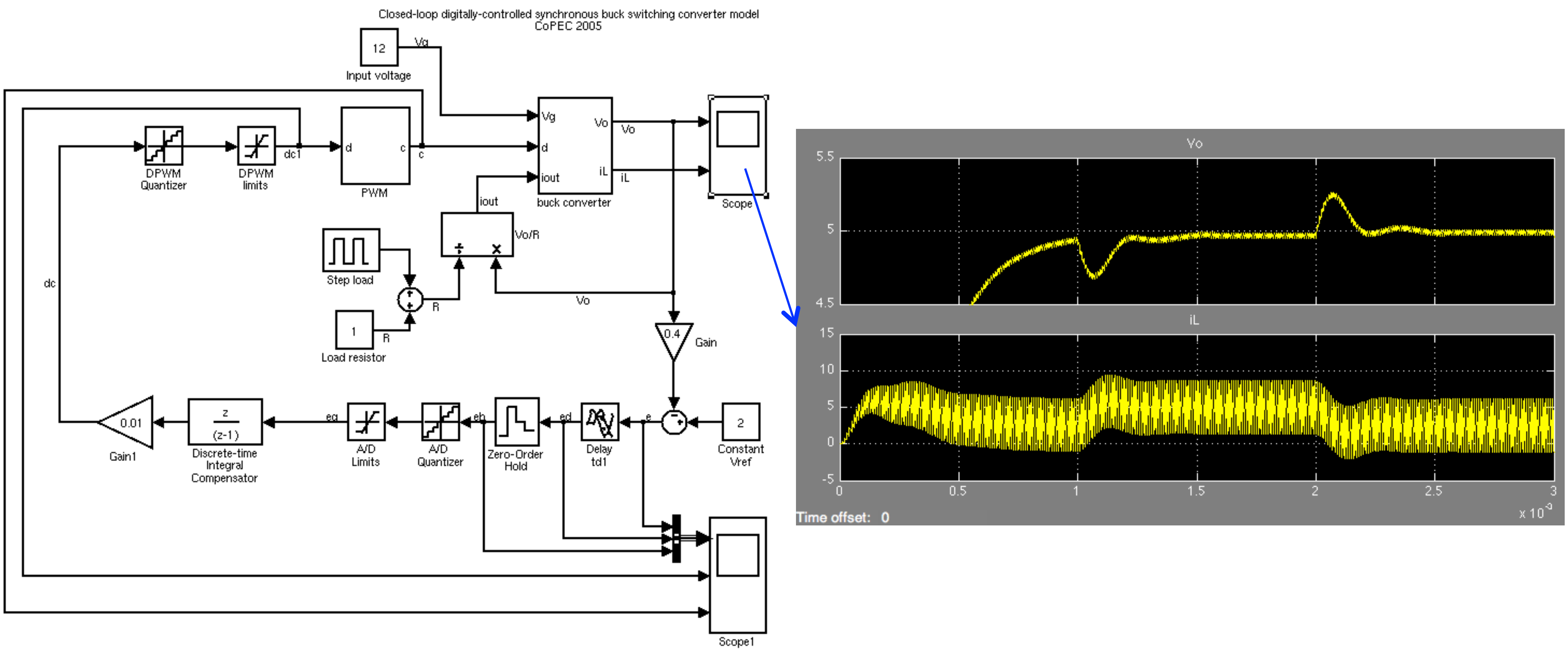
At CoPEC, nearly all simulation is done within MATLAB/Simulink

Open-loop buck converter

Time domain simulation including switching ripple

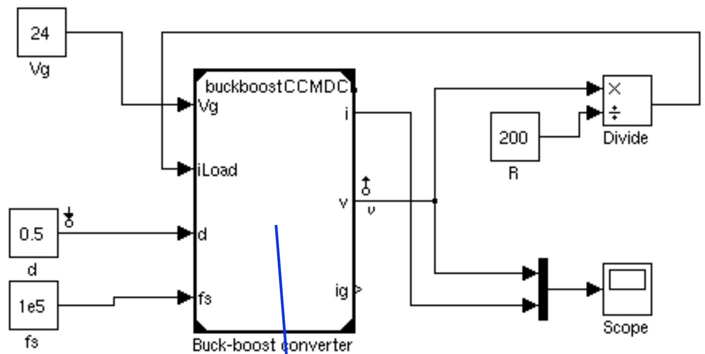


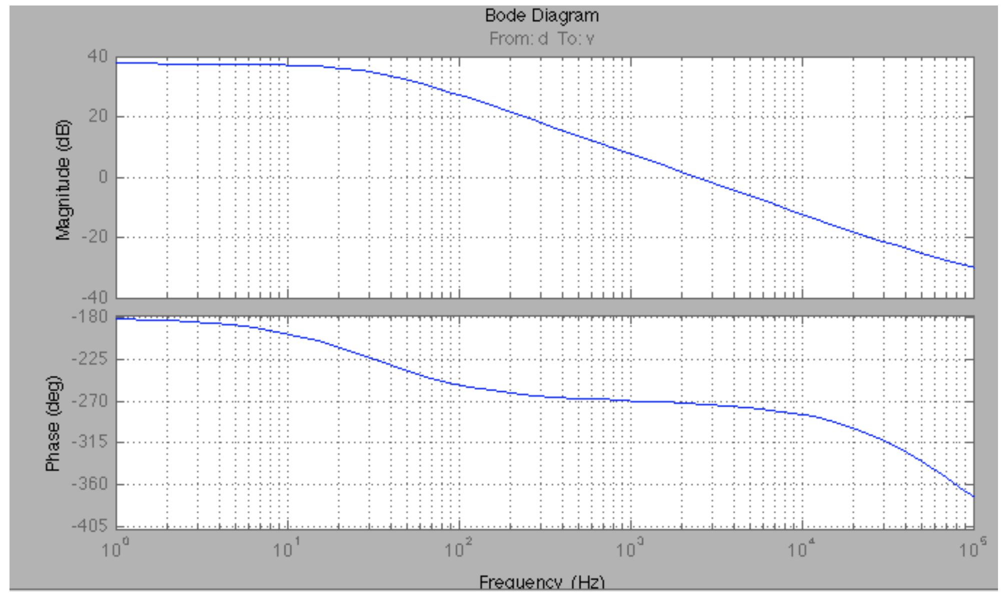
Closed-loop buck converter, digital control
Time domain simulation with switching ripple



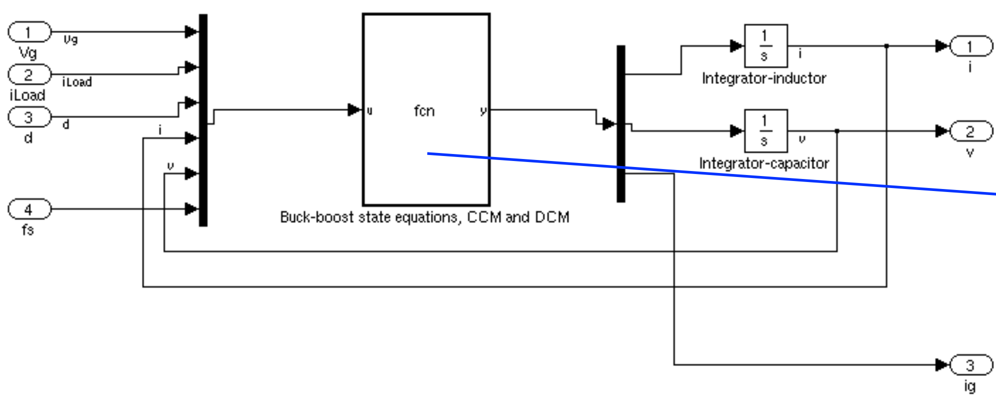
Open-loop buck-boost converter

Frequency domain simulation, averaged model





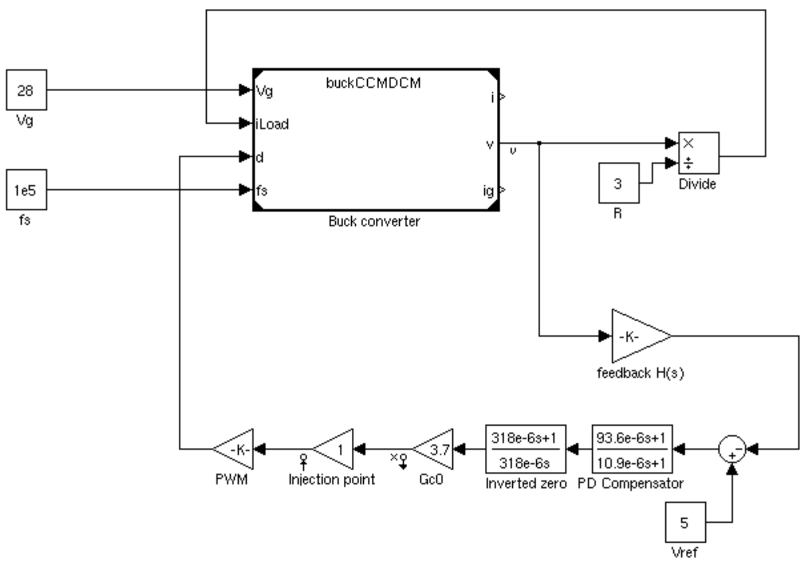
Control-to-output transfer function

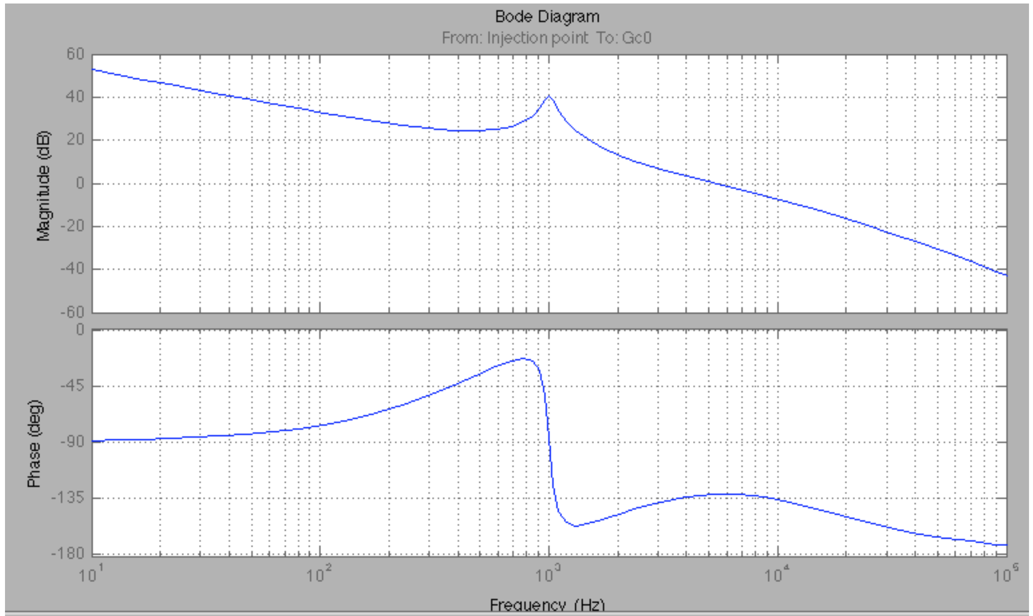



```
function y = fcn(u)
%#eml
% buck-boost converter in CCM or DCM, state equations
fs = u(6);
v = u(5);
Vg = u(1);
iLoad = u(2);
d = u(3);
i = u(4);
L = 100e-6;
C = 50e-6;
d2 = min(1-d,i*2*L*fs/(Vg*d)-d);
vL = d*Vg + d2*v;
iC = - i*d2/(d+d2) - iLoad;
ig = i*d/(d+d2);
y = [vL/L iC/C ig];
```

Closed-loop buck converter

Frequency domain simulation, averaged model





Loop gain: Bode plot

MATLAB/Simulink discussion

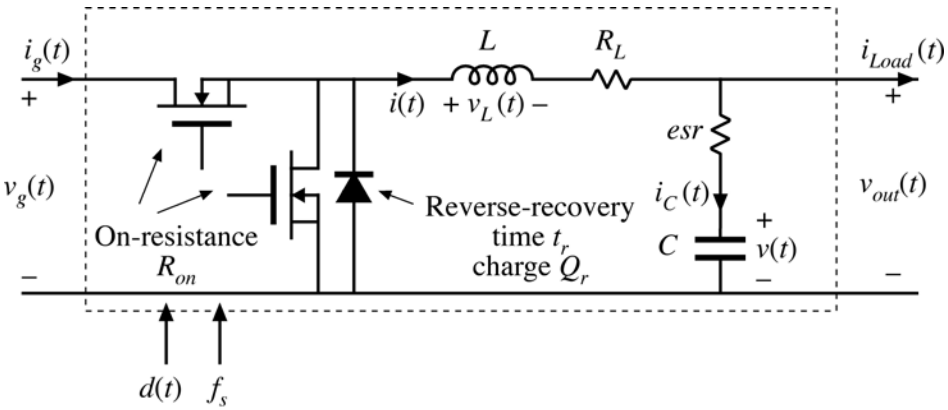
- A structured way to write the converter averaged equations, suitable for implementation in Simulink:

  State-space averaging

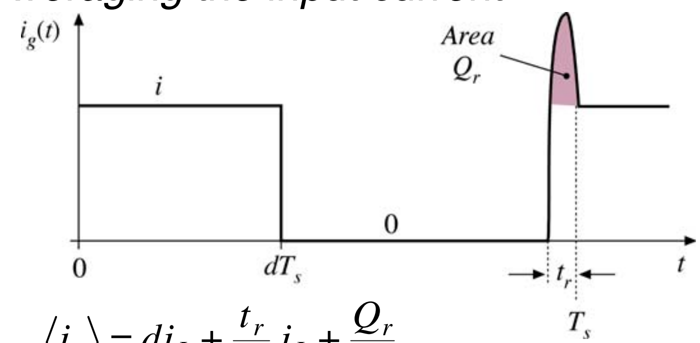
- Some basic converter models, implemented in Simulink
- How to plot small-signal transfer functions in Simulink
- Modeling the discontinuous conduction mode

Synchronous buck converter

Formulating state equations for Simulink model

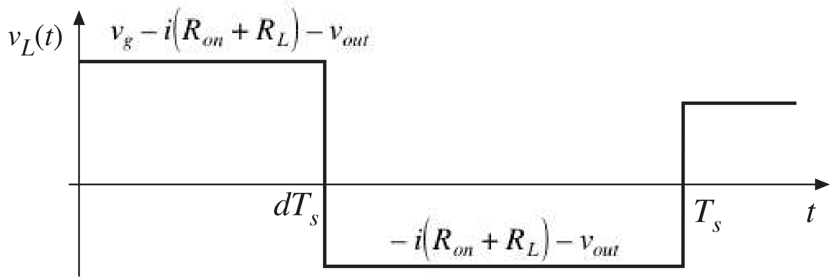


Averaging the input current



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

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

Averaging the capacitor current:

For both intervals,

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

Resulting state equations:

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

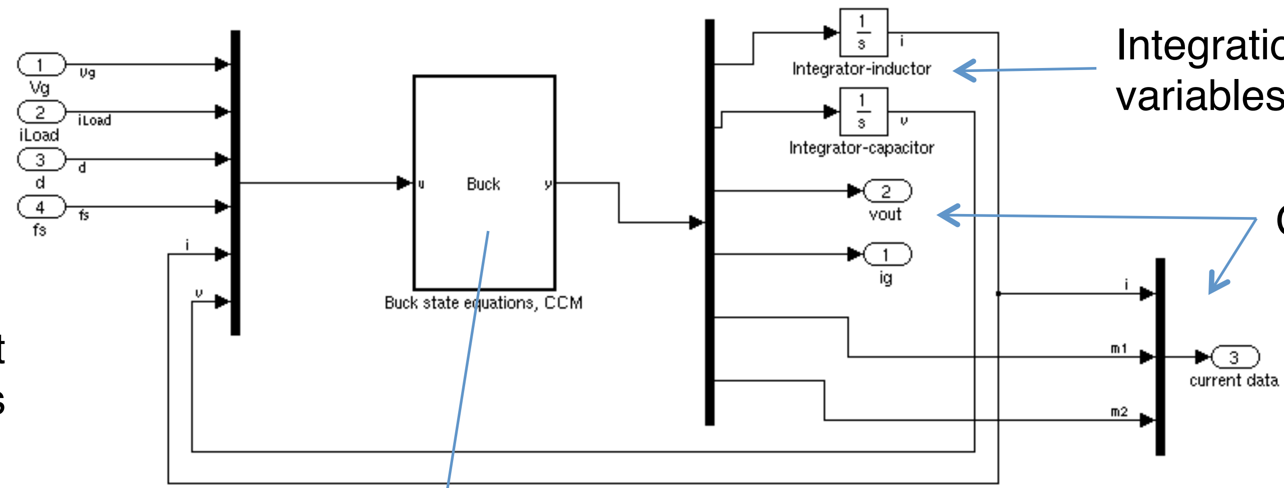
$$L\frac{di}{dt} = \langle v_L \rangle = d v_g - i(R_{on} + R_L) - v - (i - i_{Load})\, esr$$

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

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

Basic buck converter model

Averaged model for Simulink



Integration of state variables

Outputs

Independent inputs

(used in current mode control)

Embedded MATLAB code block:

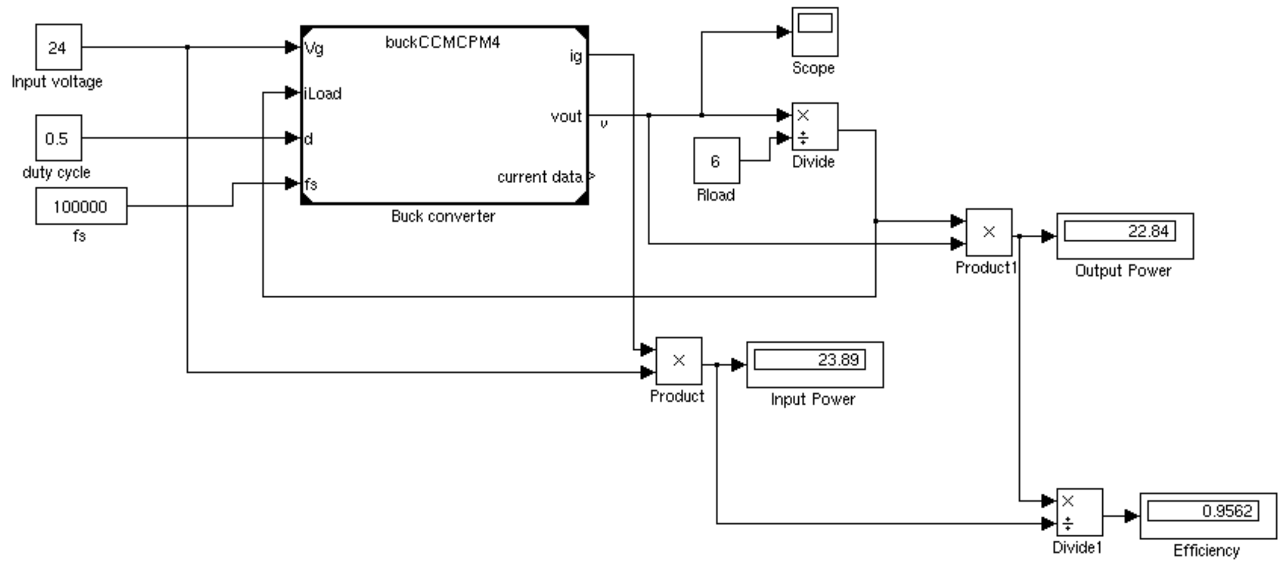
- Load inputs from ***u*** vector
- Set circuit parameters
- Calculate state equations and outputs
- Place results in output ***y*** vector

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

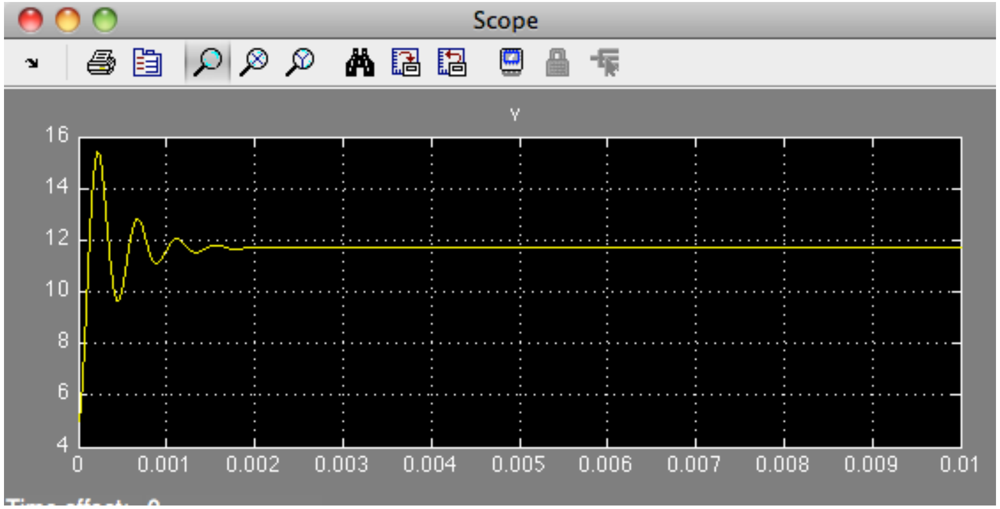
Time-domain simulation

Synchronous buck example, Simulink

Simulink model employing synchronous buck model, with voltage mode control

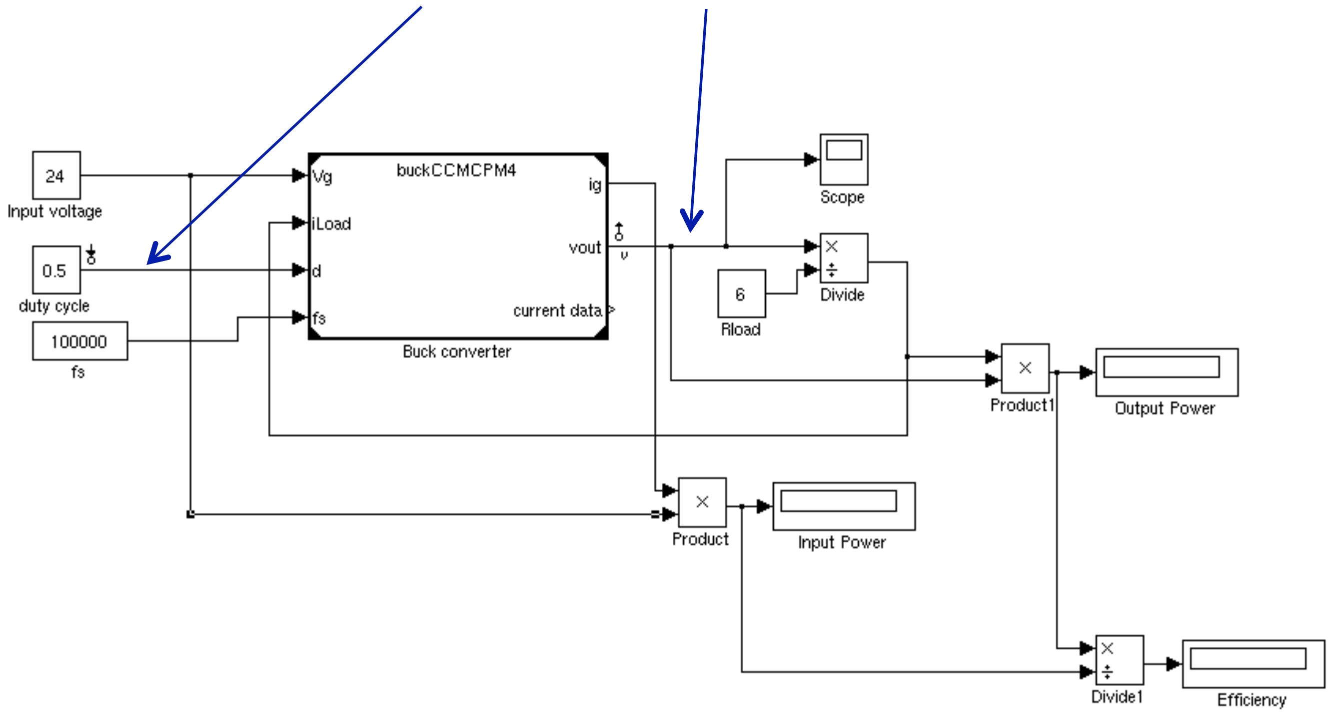


Output voltage transient response



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”

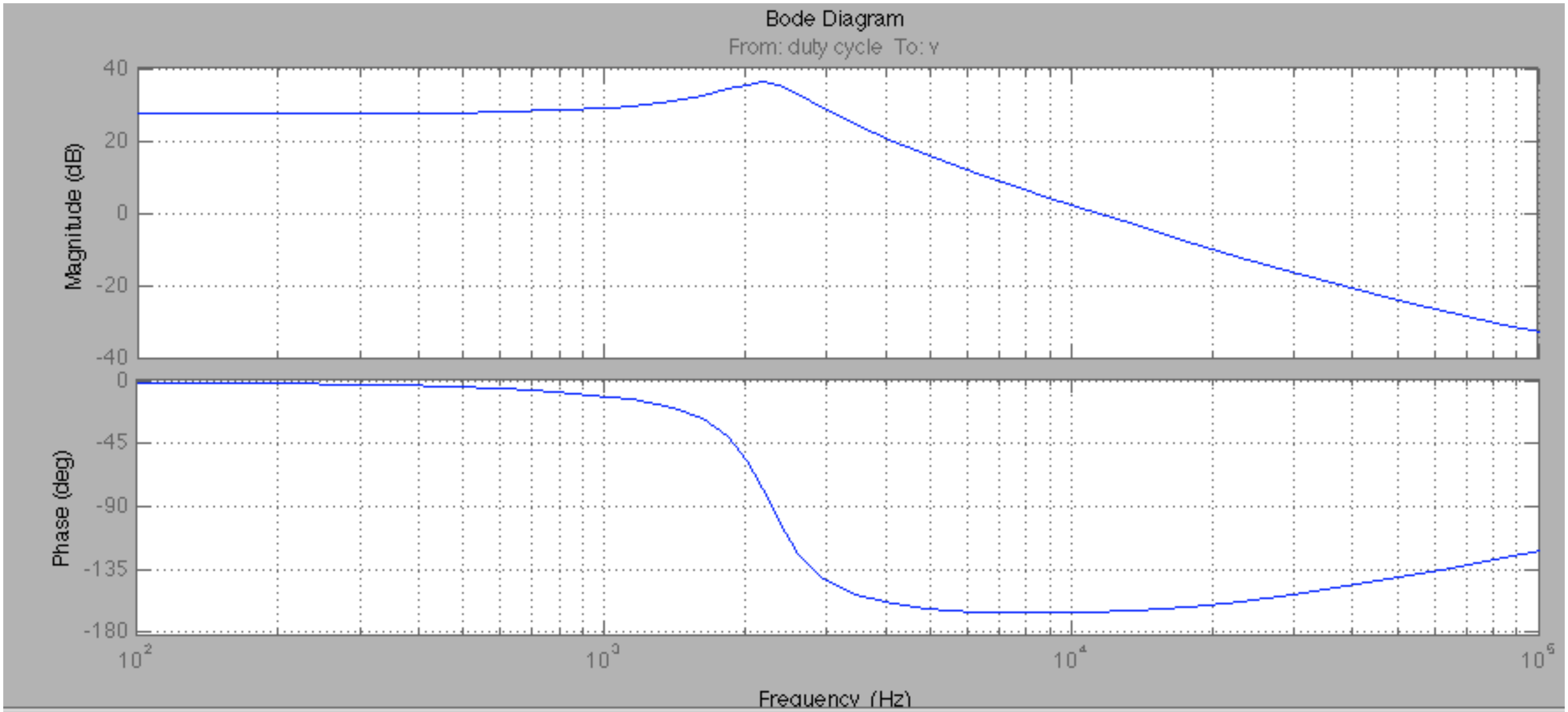
Generating a Bode Plot from Simulink, p. 2

```
%% 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}

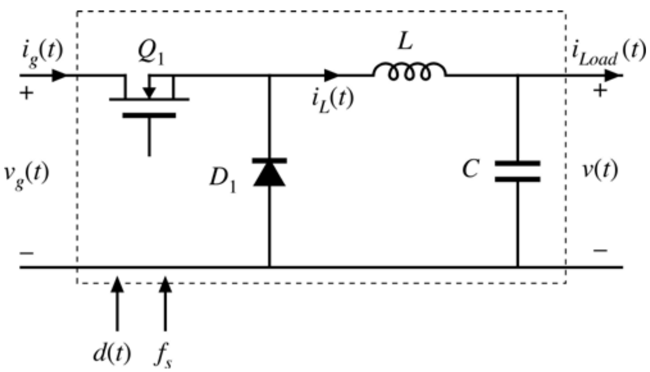
Generated by Simulink

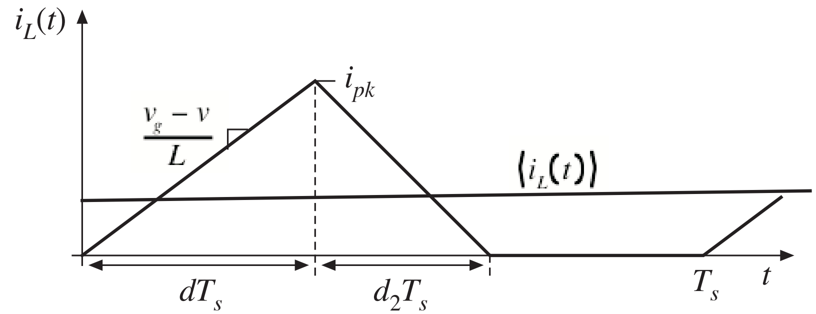


Synchronous buck example of previous slides

Modeling DCM

Buck example, Simulink model





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

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

$$\text{with} \quad i_{pk} = \frac{\left(v_g - v\right)}{L}\, dT_S$$

$$\text{so} \quad \langle i_L(t) \rangle = \frac{1}{2}\, \frac{\left(v_g - v\right)}{L}\, dT_S \left(d + d_2\right)$$

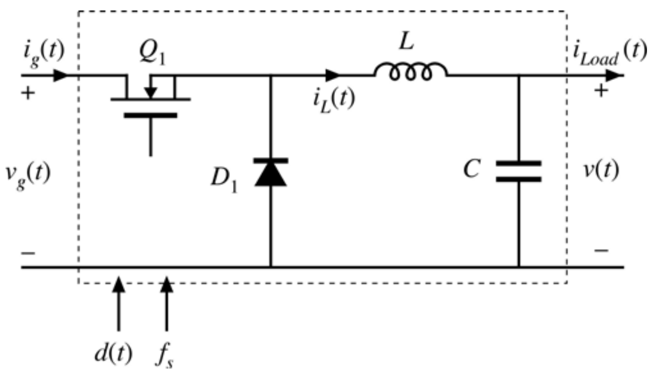
Treat the average inductor current as an independent state, and solve for d_2:

$$d_2 = \frac{2Lf_s \langle i_L(t) \rangle}{d\left(v_g - v\right)} - d$$

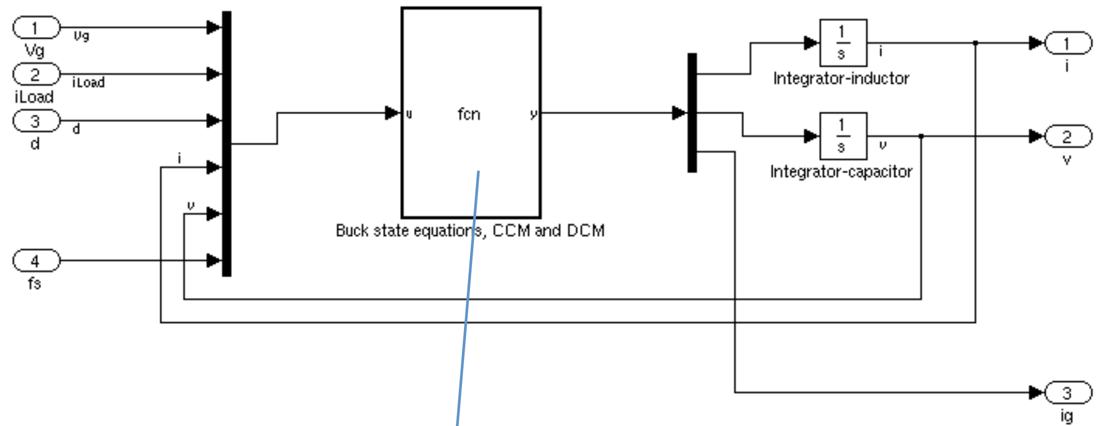
Note that $d_2 = 1 - d$ in CCM, and $d_2 < 1 - d$ in DCM

Combined CCM-DCM model

Buck converter



Simulink model



State equations:

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

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

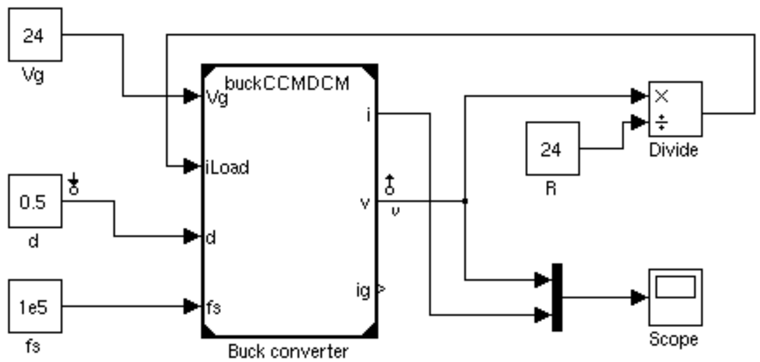
$$\langle i_g(t)\rangle = \frac{d}{d + d_2}\, i$$

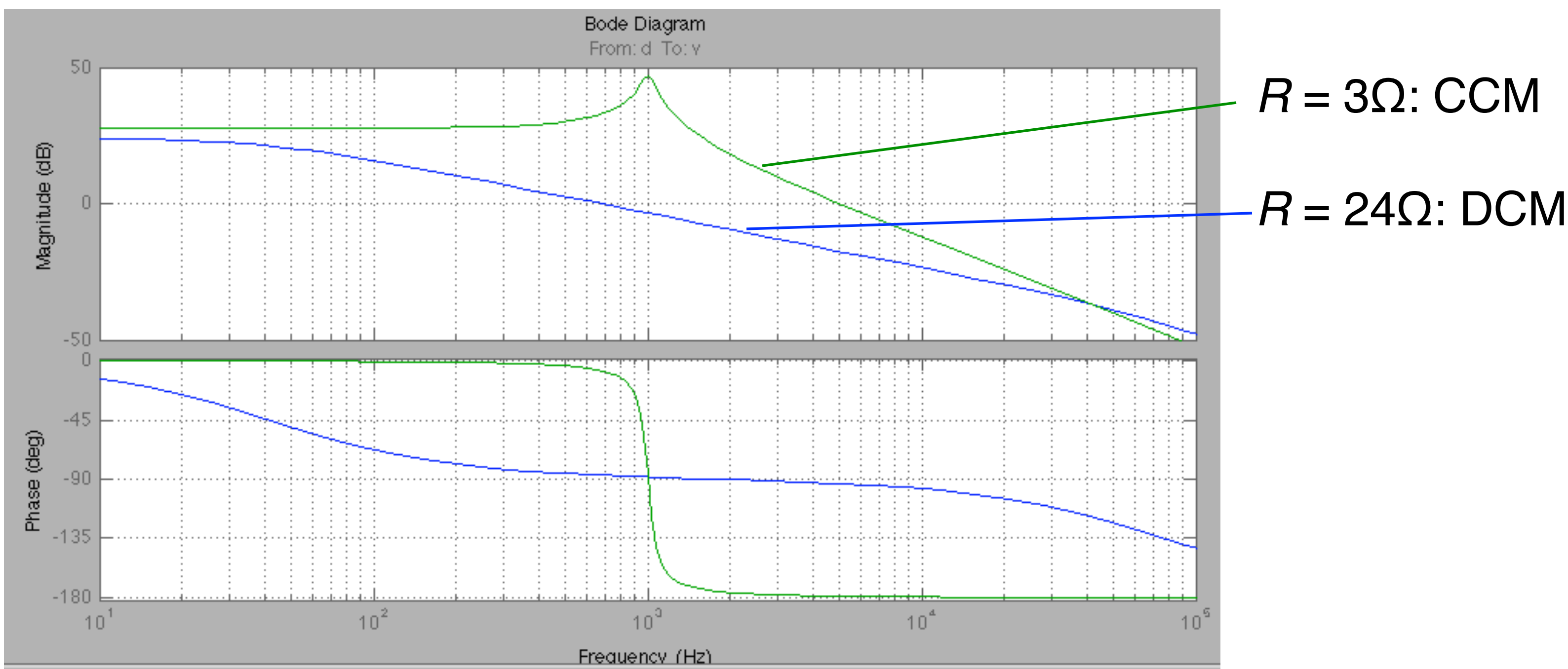
$$d_2 = \min\left(d', \frac{2Lf_s i}{d\left(v_g - v\right)} - d\right)$$

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

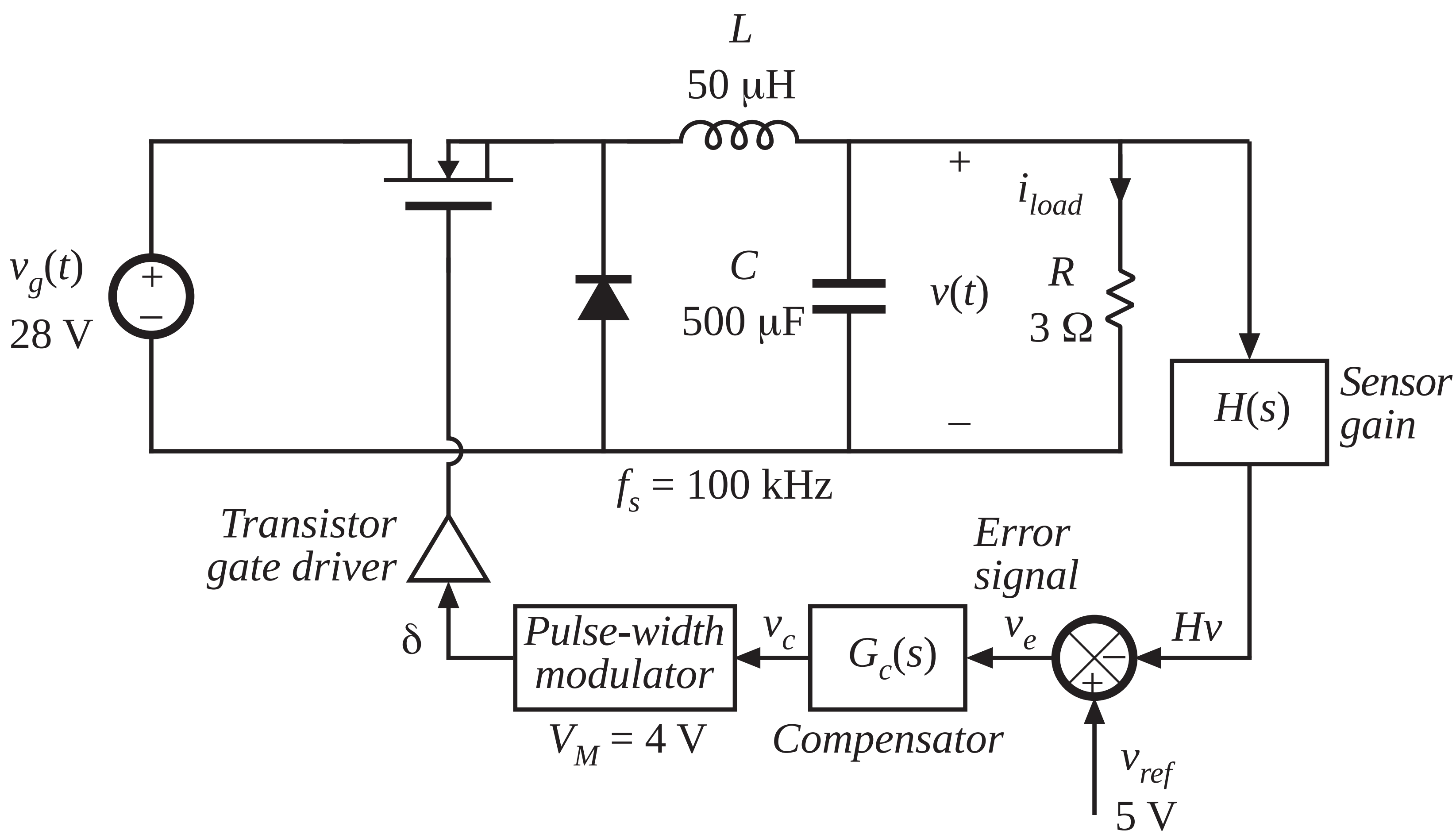
Buck control-to-output transfer function

CCM vs. DCM



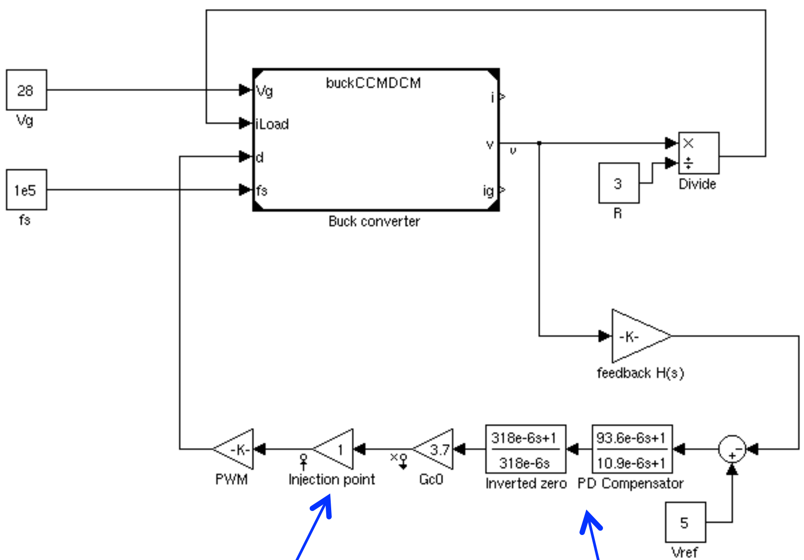


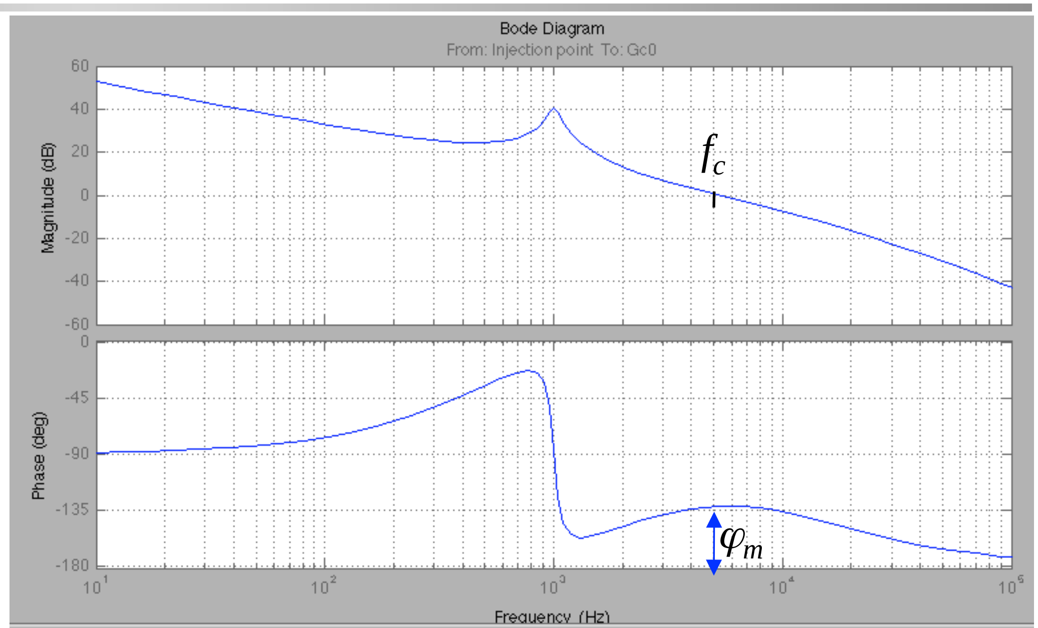
9.5.4. Design example



Closed-loop buck converter

Simulink frequency domain simulation, averaged model





Loop gain: Bode plot

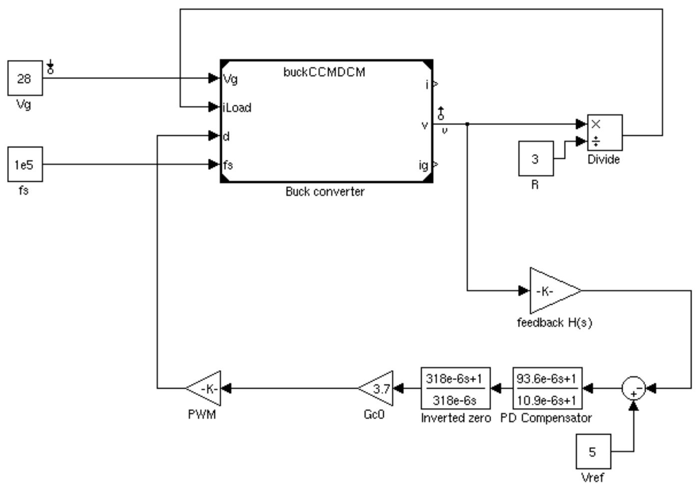
Injection point for measurement of loop gain $T(s)$

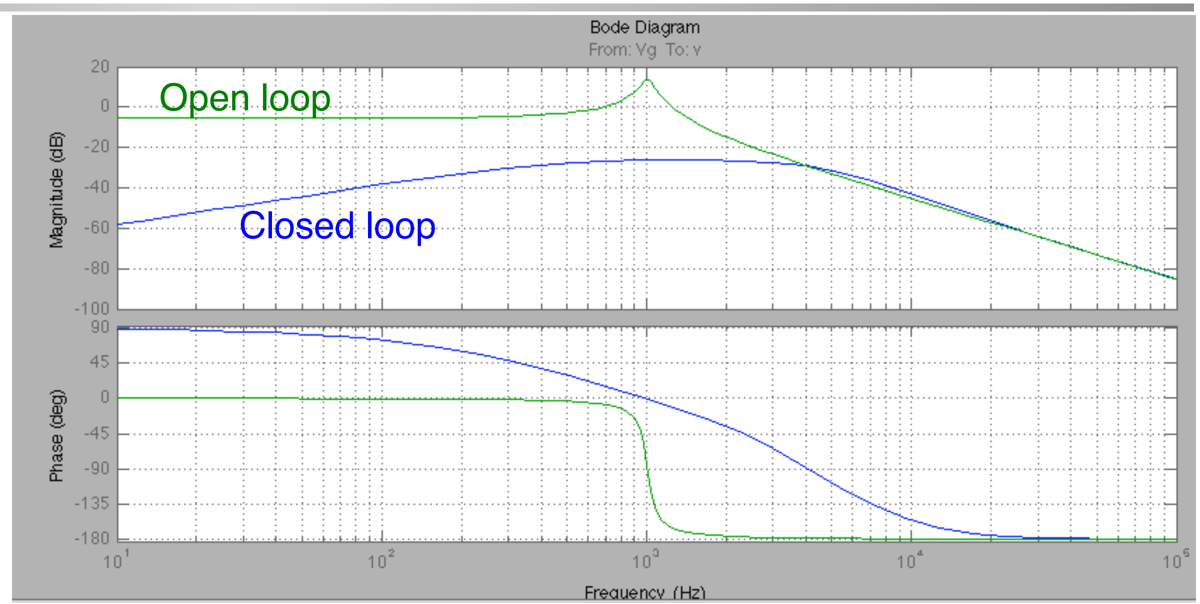
Transfer function blocks:
Implementing the PID compensator

Closed-loop line-to-output transfer function

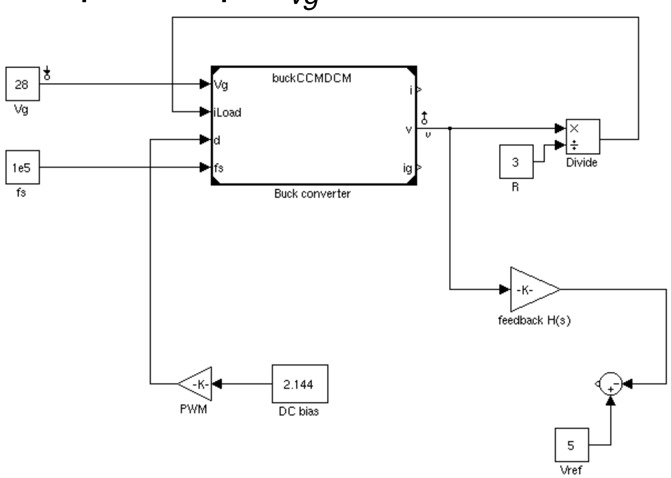
Simulink frequency domain simulation

Closed loop G_{vg}





Open loop G_{vg}



Script that generates both plots

```
%% Bode plotter using linearization tool
% requires simulink control design toolbox
mdl = 'BuckFeedbackExampleCh9PIDGvg'; % set
io = getlinio(mdl)  % get i/o signals of md
op = operspec(mdl)
op = findop(mdl,op)  % calculate model oper
lin = linearize(mdl,op,io)  % compute state
mdl = 'BuckFeedbackExampleCh9GvgOpenLoop';
io = getlinio(mdl)  % get i/o signals of md
op = operspec(mdl)
op = findop(mdl,op)  % calculate model oper
lino = linearize(mdl,op,io)  % compute stat
ltiview(lin,lino)  % send linearized model
```