

ECEN5807 supplementary notes

Introduction to MATLAB/Simulink for switched-mode power converters

ECEN5807

Colorado Power Electronics Center
University of Colorado, Boulder

ECEN5807 supplementary notes


1.1 Getting started with MATLAB/Simulink

- Starting and running simulations in MATLAB/Simulink
- Constructing Simulink models
- Examples:
 - Open-loop synchronous buck converter model
Simulink file: **buck_open_loop.mdl**
 - Buck converter and PWM subsystem models
 - Closed-loop synchronous buck converter model with an analog controller
Simulink file: **buck_closed_loop.mdl**
 - Load transient model and simulations
Simulink file: **buck_closed_loop_load.mdl**

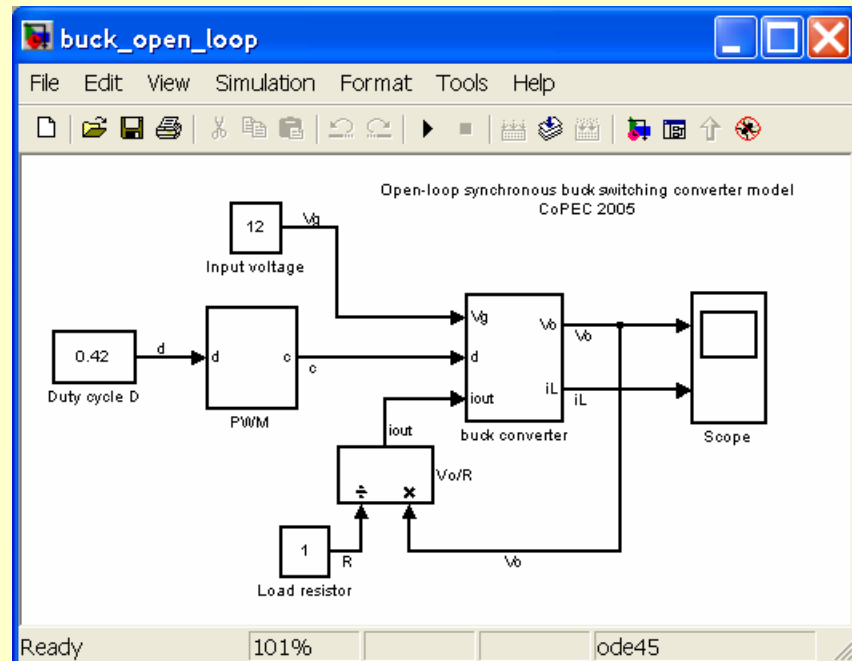
1.2 Digitally controlled buck converter: Simulink models and simulations

- System model
- A/D converter, discrete-time compensator, and DPWM models
Simulink file: **buck_closed_loop_discrete.mdl**

1.1 Starting MATLAB/Simulink

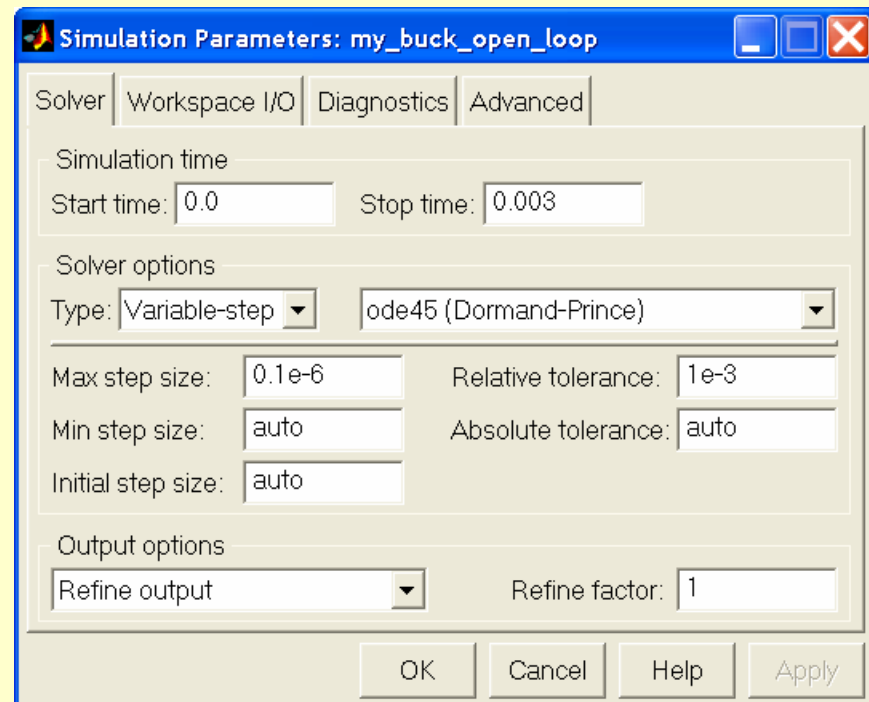
- Start **MATLAB**:
 (double-click on the MATLAB shortcut) 
- Open a file, in the MATLAB window menu:
 - Select file: **buck_open_loop.mdl**, then **Open**
- This opens a pre-configured Simulink model for an open-loop synchronous buck switching converter

- Converter parameters:
 - $L = 4.1 \mu\text{H}$, $R_L = 80 \text{ m}\Omega$
 - $C = 376 \mu\text{F}$, $R_{esr} = 5 \text{ m}\Omega$
 - $f_s = 100 \text{ KHz}$
 - $V_g = 12 \text{ V}$, $D = 0.42$
 - Load $R = 1 \Omega$



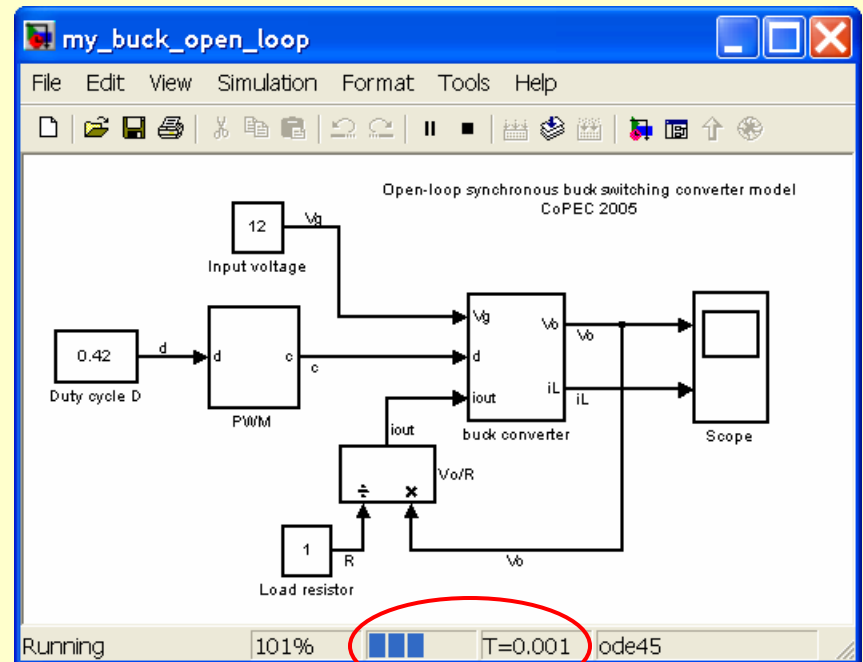
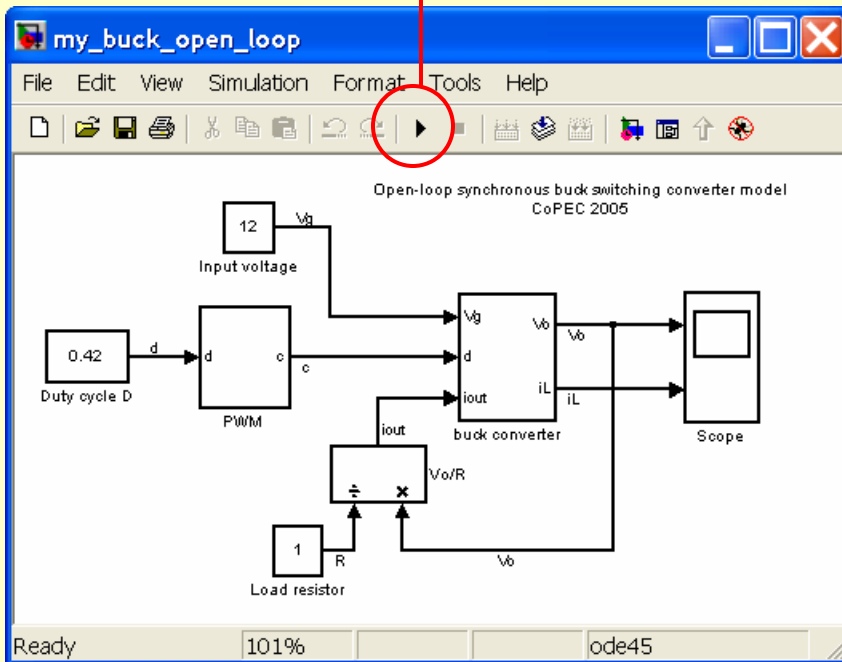
Transient Simulations in Simulink

- Make your copy of the Simulink model for further edits
 - In the current Simulink window:
 - **File** → **Save As** → **File Name:** enter **my_buck_open_loop.mdl**, then **Save**
- Check or adjust simulation parameters:
 - **Simulation** → **Simulation parameters**
- This opens a window to adjust simulation parameters such as Start Time, Stop Time, solver options, step size, etc.
- The default parameters and options are usually fine, except:
 - Enter appropriate **Stop time** (3 ms in this example)
 - Enter **Max step size** of about 1/100 of the switching period (0.1 μ s in this example)



Starting Transient Simulation

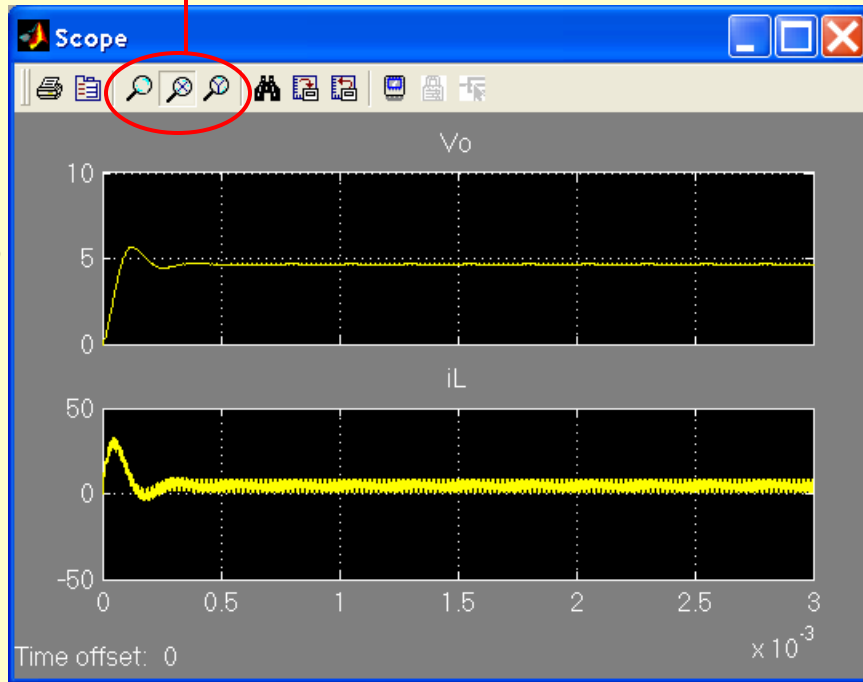
- In the current Simulink window:
 - **Simulation** → **Start** or click on the **Start button** in the toolbar



Current simulation time and progress are shown here

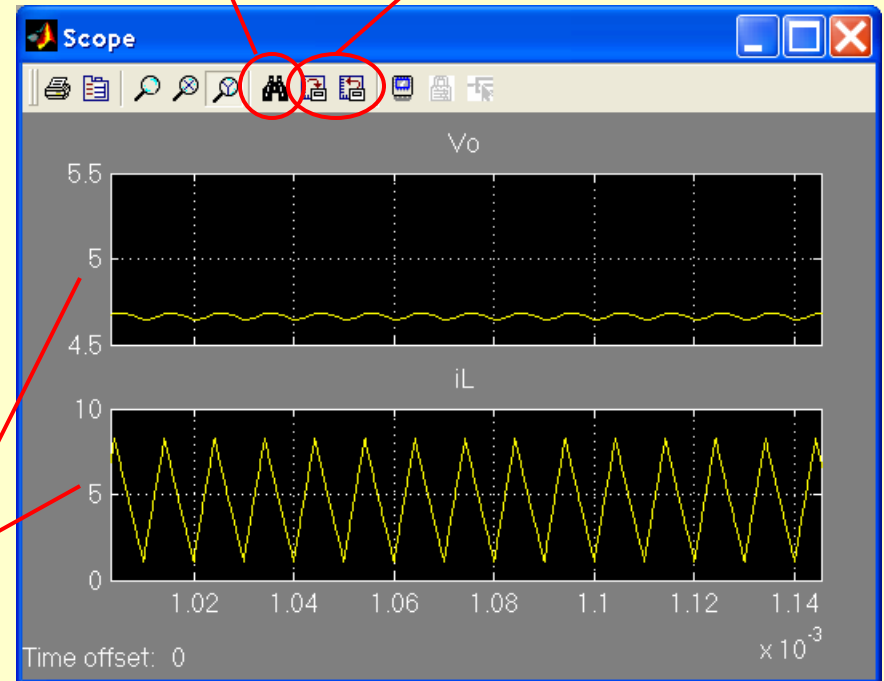
View Simulation Results

- In the **my_buck_open_loop** Simulink window, double-click the **Scope** block
- Use rectangular box, X-axis, or Y-axis Zoom tools to view waveforms details



Autoscale fits the entire waveform into the window

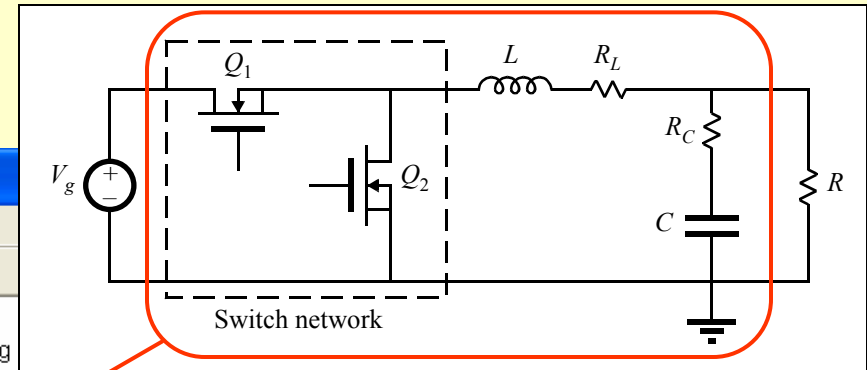
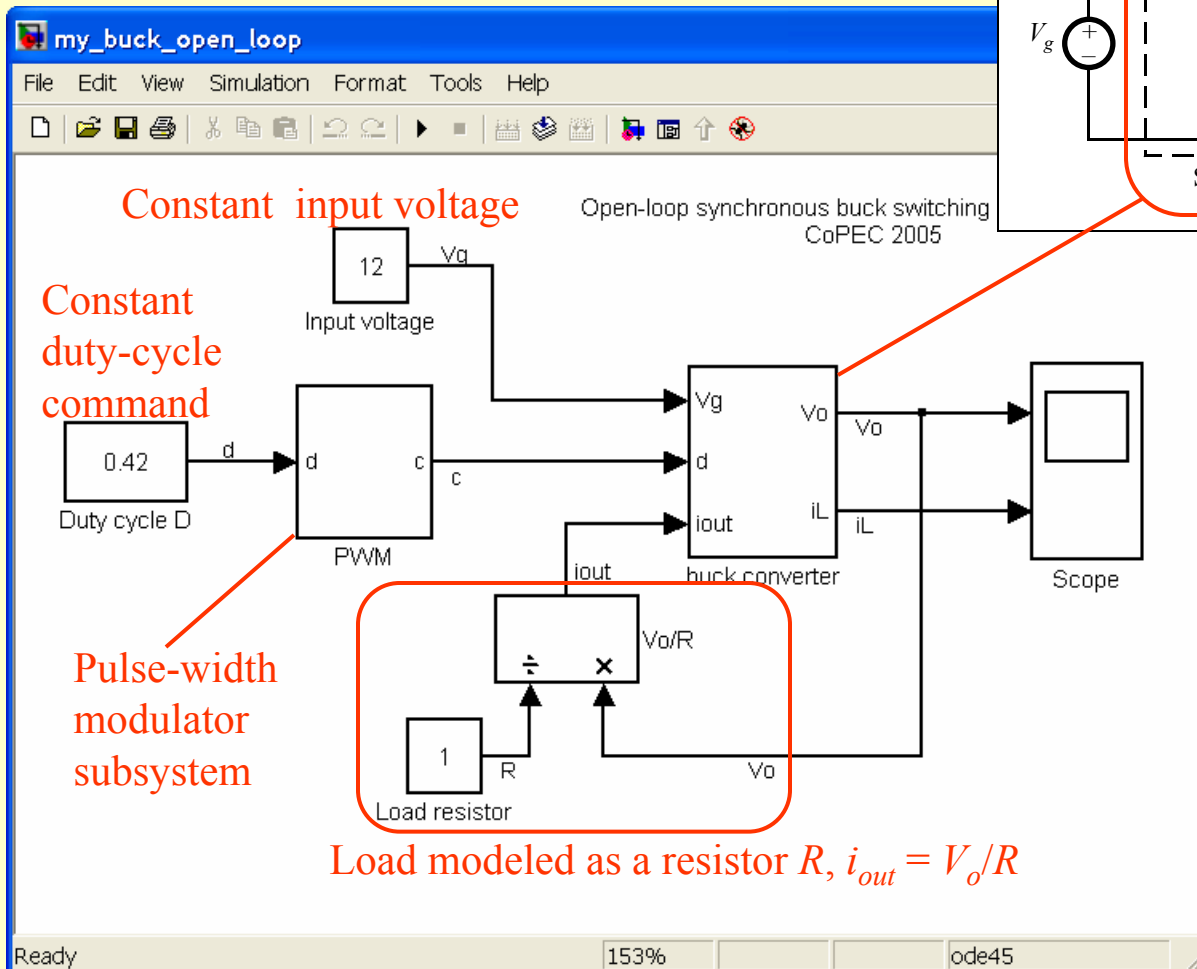
Current axes settings can be Saved or Restored



To enter a specific vertical axis range, right-click on the waveform, select **Axes properties...** then enter **Y-min** and **Y-max**, click **OK**

Construction of Simulink Models

Top-level system model

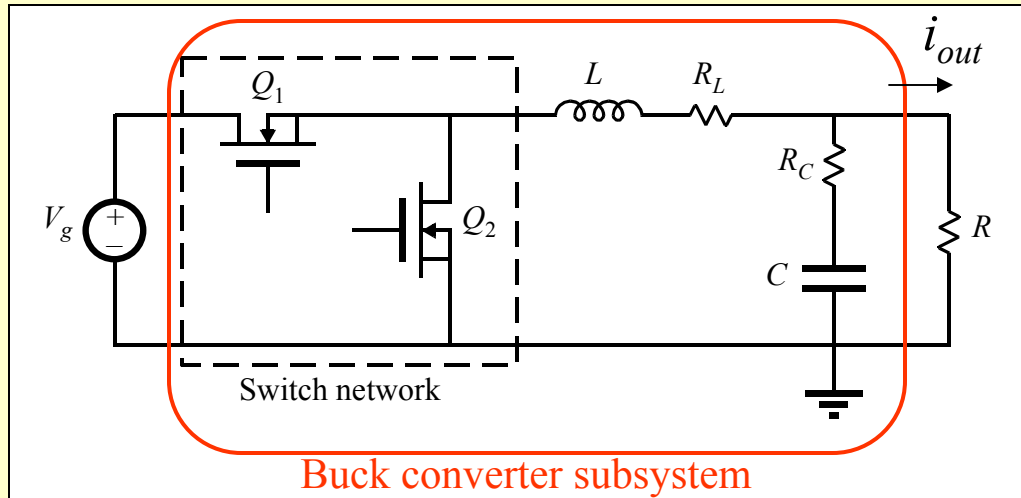


Buck converter subsystem

Simulink models are block-diagrams consisting of

- “Sources” (such as **Constant V_g** block)
- “Sinks” (such as **Scope**) and
- Various functional blocks, including subsystems

Buck Converter Subsystem



- System equations:

$$\frac{di_L}{dt} = \frac{1}{L} (V_g \cdot d - i_L R_L - v_o)$$

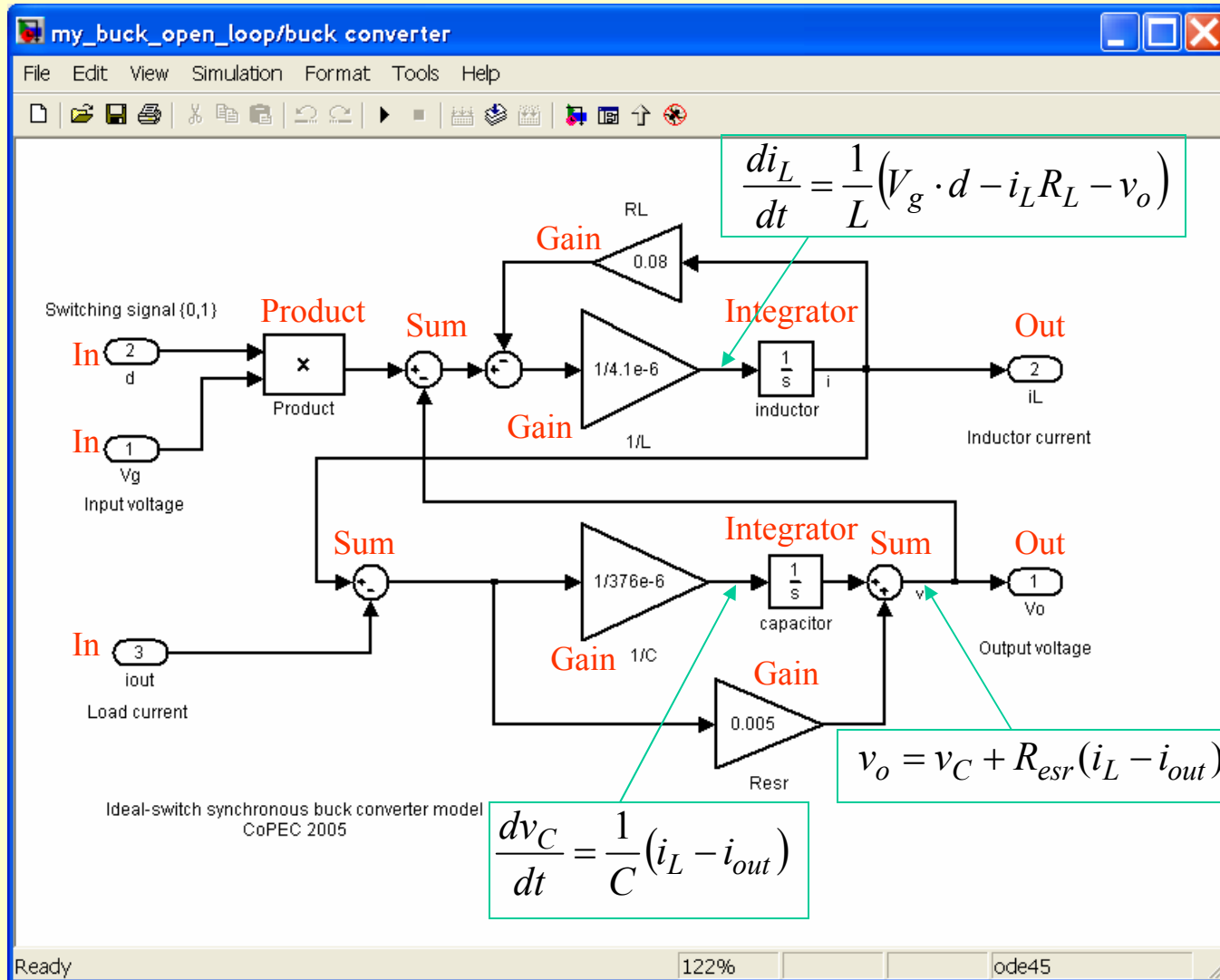
$$\frac{dv_C}{dt} = \frac{1}{C} (i_L - i_{out})$$

$$v_o = v_C + R_{esr} (i_L - i_{out})$$

- Double-click on the **buck converter** subsystem block to view a Simulink implementation of the system equations

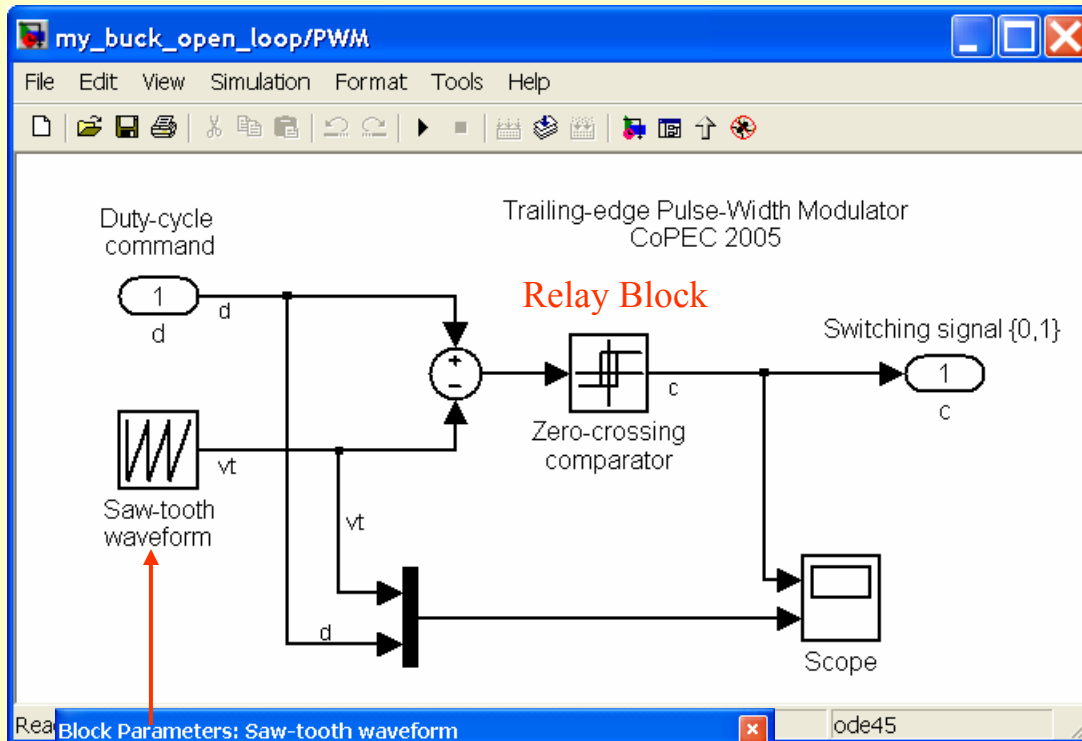
- Inputs:
 - Input voltage V_g
 - Switching signal $d = \{0, 1\}$
 - Load current i_{out}
- Outputs:
 - Output voltage V_o
 - Inductor current i_L

Buck Converter Subsystem



On this slide, the subsystem model is annotated with the system equations and Simulink block names, such as **Product**, **Gain**, **Integrator**, etc., shown in red

PWM Subsystem



Block Parameters: Saw-tooth waveform

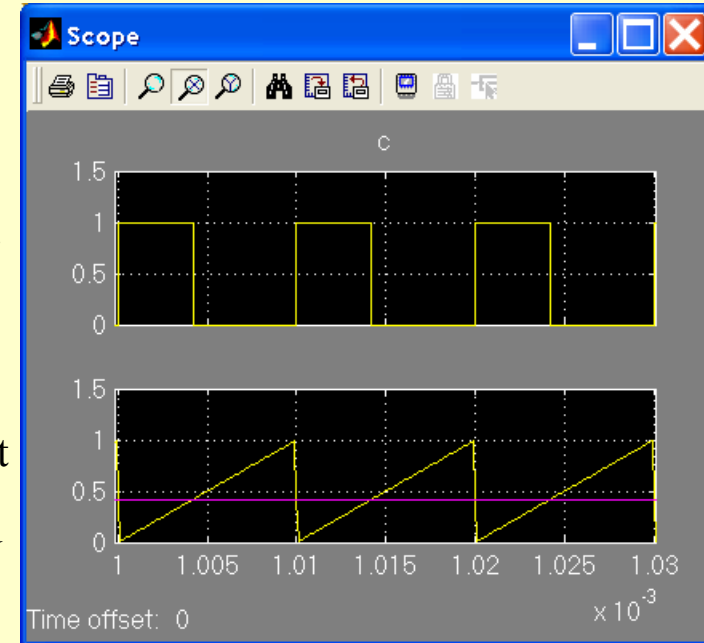
Repeating table (mask) (link)
Output a repeating sequence of numbers specified in a table of time-value pairs. Values of time should be monotonically increasing.

Parameters

Time values:
[0 0.001e-5 1e-5]

Output values:
[1 0 1]

OK Cancel Help Apply



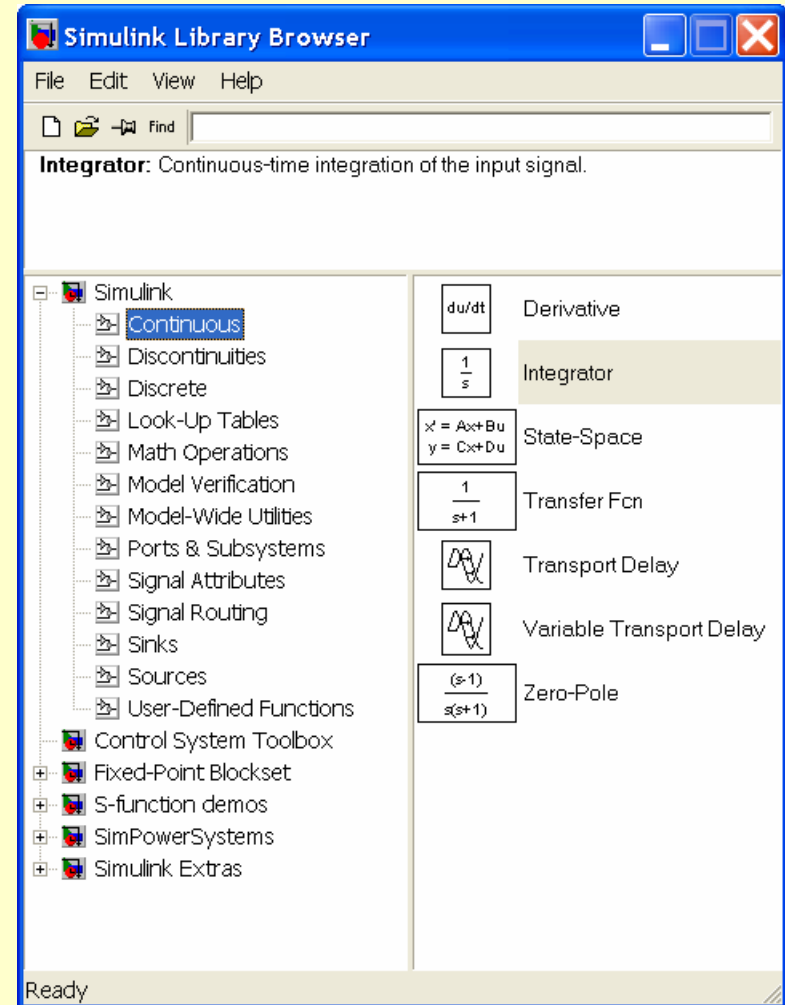
- In the my_buck_open_loop window, double-click on the PWM block to open the subsystem window
- Double-click on the **Saw-tooth waveform** block to view or change the block parameters
- Note that **Time values [0 0.001e-5 1e-5]** and the corresponding **Output values [1 0 1]** define the saw-tooth waveform in the PWM (switching frequency is: $1/1e-5 = 100 \text{ KHz}$)

Constructing a Closed-Loop Model

- In this step, the objective is to construct and simulate a closed-loop voltage regulator using a simple continuous-time integral compensator
- Save `my_buck_open_loop.mdl` as `my_buck_closed_loop.mdl`
- In the `my_buck_closed_loop` window, click on the Library Browser button



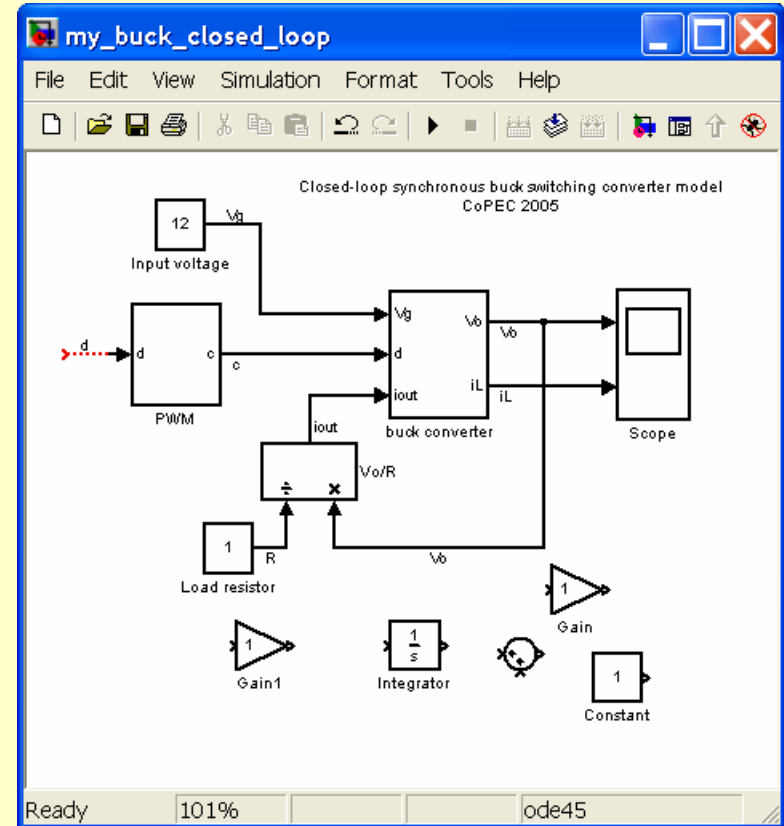
to open the Simulink Library Browser window



Simulink Library Browser

Constructing the Closed-Loop Model, continued

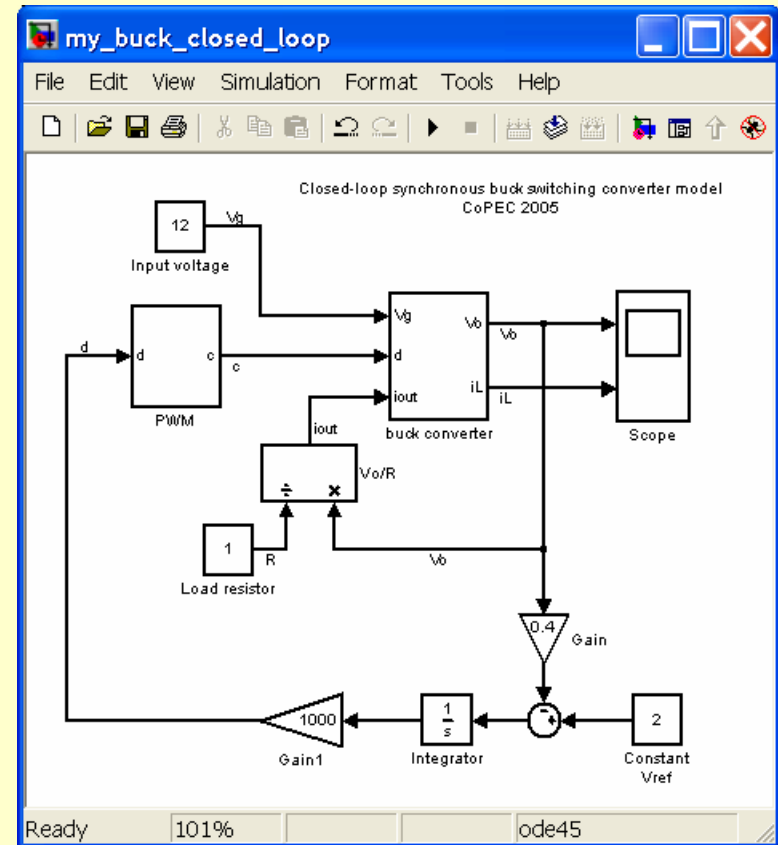
- In the Simulink **Library Browser** window, select
 - Simulink → Continuous → Integrator**
- Drag an **Integrator** block to the **my_buck_closed_loop** window, click the left mouse button to place the integrator
- Similarly, add the following Simulink blocks to **my_buck_closed_loop**:
 - Two **Gain** blocks
(**Simulink → Math → Gain**)
 - A **Sum** block
(**Simulink → Math → Sum**)
 - A **Constant** block
(**Simulink → Sources → Constant**)
- Delete the **Duty Cycle D (Constant)**: select the **Duty Cycle D** block and press **Delete** key



my_buck_closed_loop after the edits listed on this page

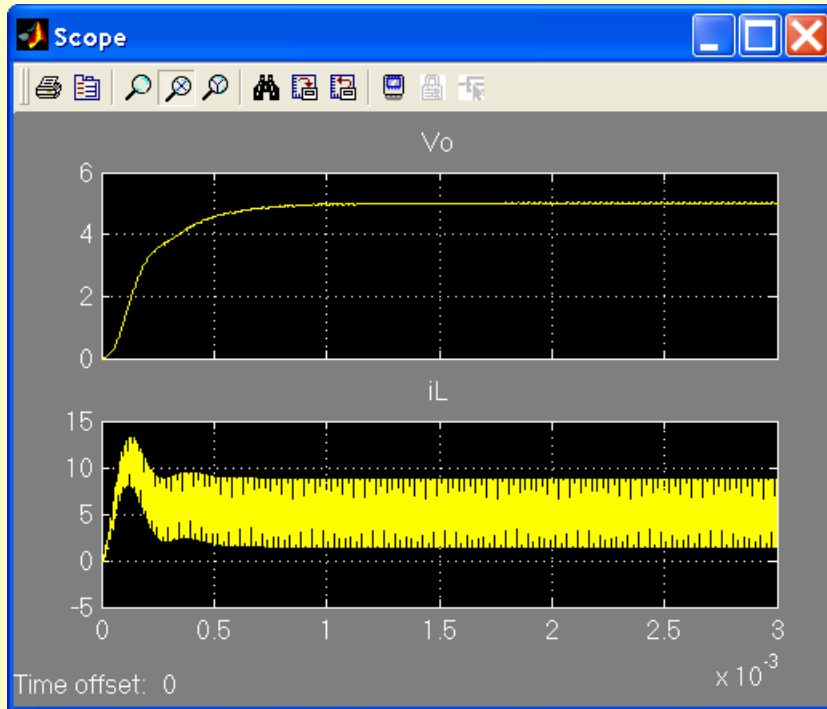
Constructing the Closed-Loop Model, completed

- Change the orientation of the blocks for easier wiring:
 - Select the **Gain** block, click the right mouse button, select **Format** → **Rotate block**
 - Similarly, using **Rotate block** or **Flip block**, change the orientation of the **Integrator**, **Sum**, **Gain1** and **Constant** blocks
- Double-click on the **Sum** block to change the input for the sensed output voltage to minus (–); reorder the symbols +, –, and | as desired
- Wire the blocks to construct the closed-loop model
- Set the model parameters (double click the block and edit the default values):
 - **Gain** = 0.4 (gain H of the voltage divider sensing the output voltage)
 - **Gain1** = 1000 (gain of the integral compensator); you may need to resize the block to show the parameter value: select then drag a corner to resize the block
 - **Constant** = 2 (constant $V_{ref} = 2$, so that in steady-state $V_o = V_{ref}/H = 5$ V)
- Run a simulation to verify that the output voltage comes to $V_{ref}/H = 5$ V in steady state

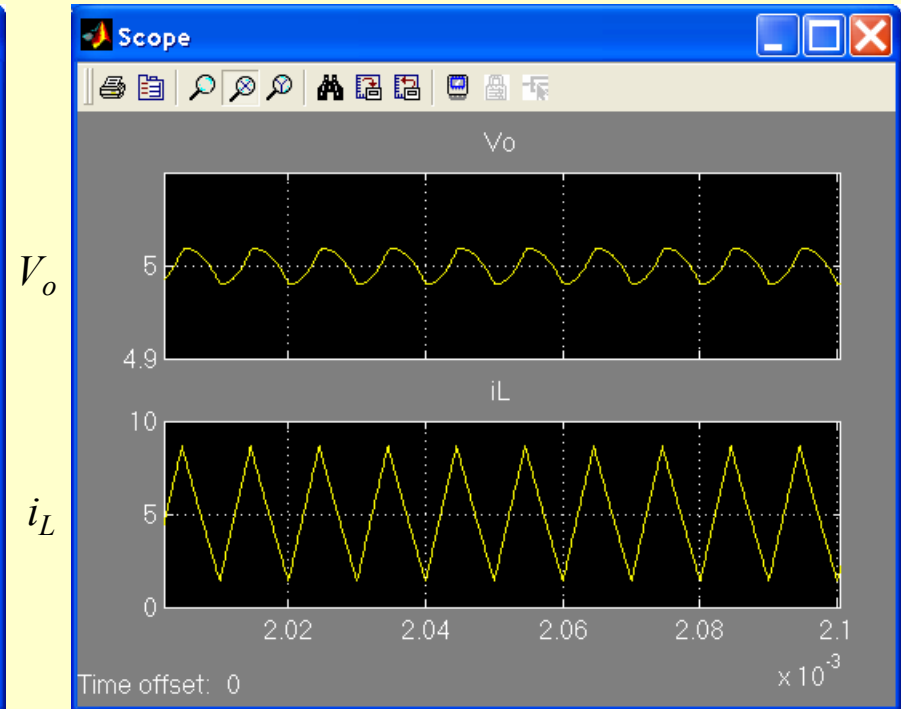


my_buck_closed_loop after the edits listed on this page

Closed-Loop Simulation Results



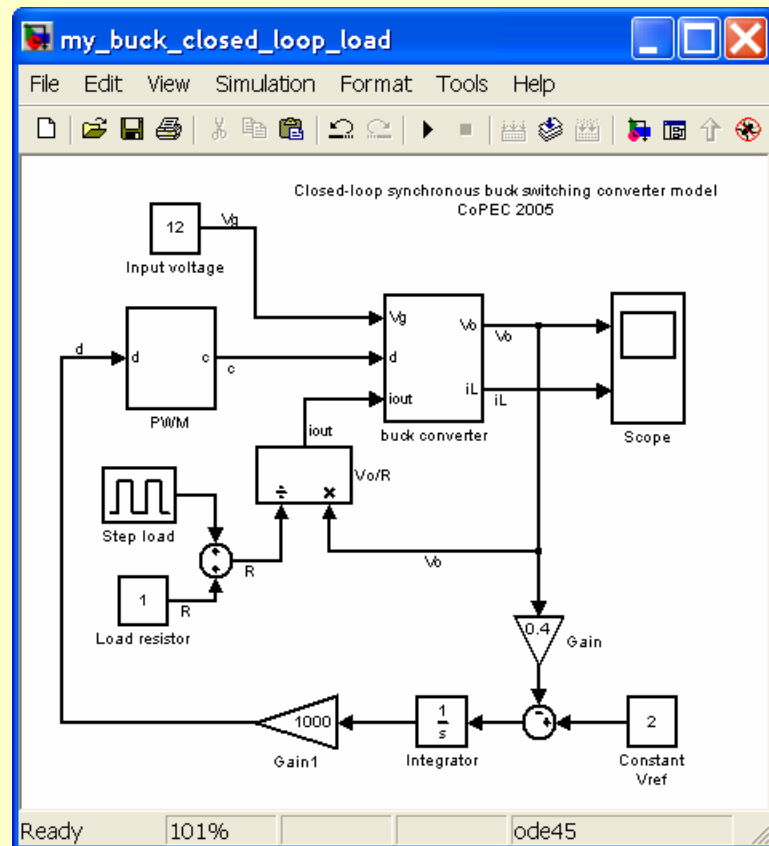
The output voltage V_o and the inductor current i_L during a start-up transient in the closed-loop buck converter with the continuous-time integral compensator



Details of the steady-state output voltage V_o and the inductor current i_L in the closed-loop buck converter with the continuous-time integral compensator

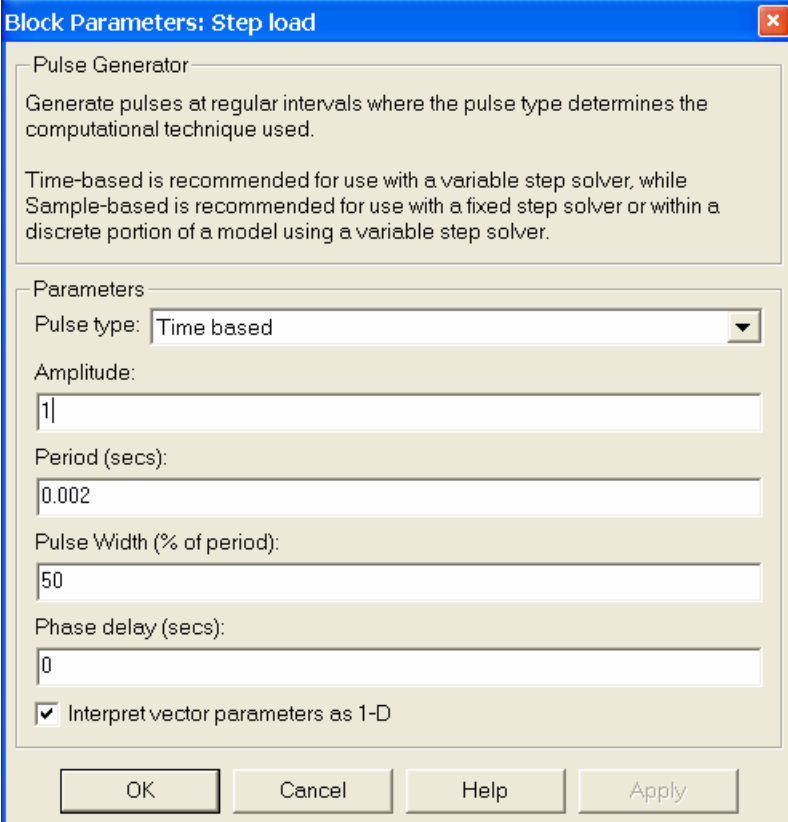
Adding a Step Load Transient Model

- In this step, the objective is to add a step load transient to the closed-loop converter model
- In the Simulink Library Browser window, select **Simulink** → **Sources** → **Pulse Generator**
- Place a **Pulse Generator** block and another **Sum** block in the **my_buck_closed_loop** window
- Click on the block name to rename the **Pulse generator** to **Step load**
- Wire the blocks as shown in the diagram
- Set the parameters of the **Step load** block as shown on the next page
- Save the system model as **my_buck_closed_loop_load.mdl**



Step Load Transient Parameters

- The objective is to set the parameters of the **Step load** pulse generator block to step the total load resistance from $2\ \Omega$ to $1\ \Omega$ and back, corresponding to a 50% to 100% load transient
- Double click on the **Step load** block to open the **Block Parameters** window
- Set **Amplitude** to 1, the **Period** to 2 ms, and the **Pulse Width** to 50%
- With these parameters, the **Step load** block periodically adds the resistance of $1\ \Omega$ to the constant load resistance of $1\ \Omega$. As a result, the total load resistance is $2\ \Omega$ from 0 to 1 ms, $1\ \Omega$ from 1 ms to 2 ms, back to $2\ \Omega$ from 2 ms to 3ms, etc.
- Note that in this model the step load change occurs instantaneously



Block Parameters: Step load

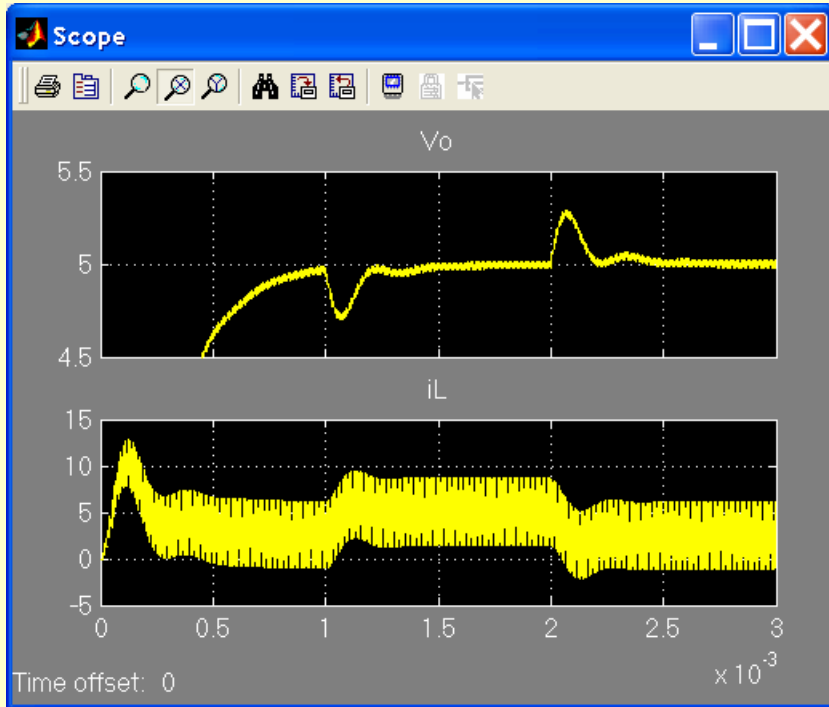
Pulse Generator
 Generate pulses at regular intervals where the pulse type determines the computational technique used.
 Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters
 Pulse type: Time based
 Amplitude: 1
 Period (secs): 0.002
 Pulse Width (% of period): 50
 Phase delay (secs): 0
 Interpret vector parameters as 1-D

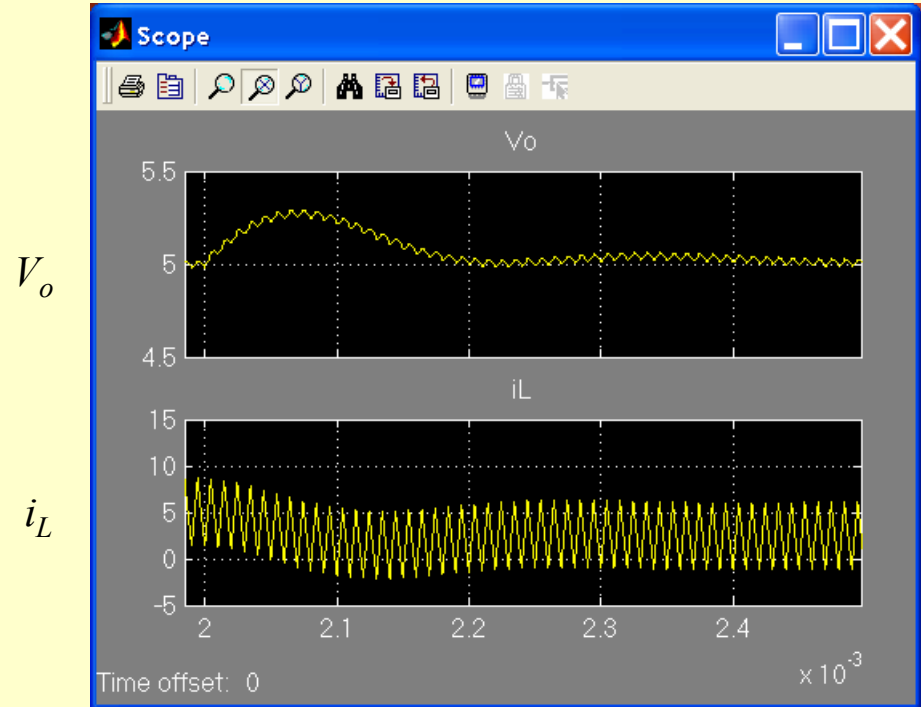
OK Cancel Help Apply

Step Load Transients

Review the step-load transient simulation results



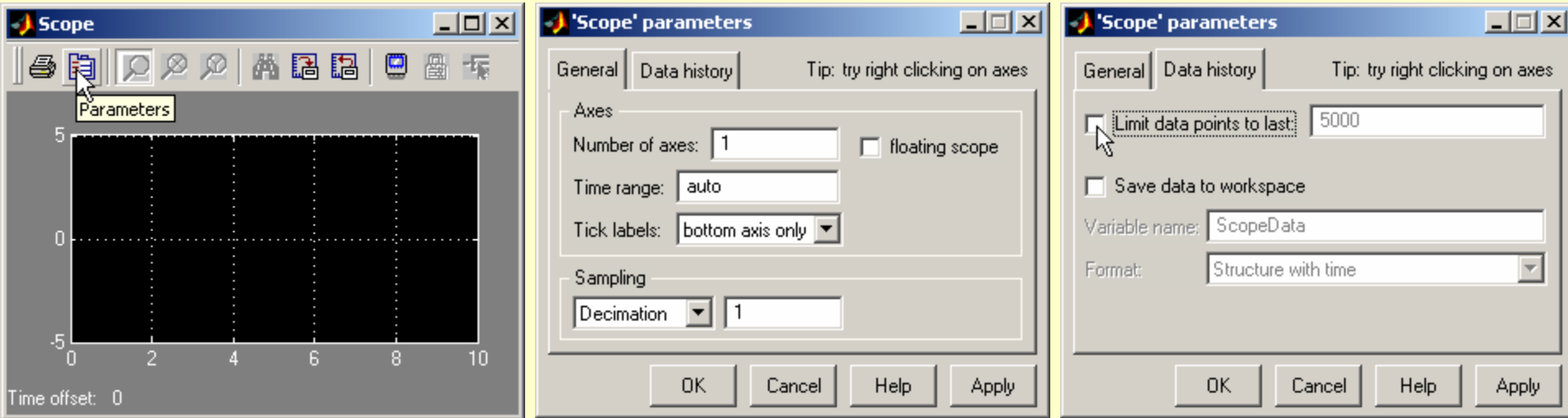
Start-up, 50-to-100% and 100-to-50% load transient responses in the closed-loop converter with the simple continuous-time integral compensator



Details of the 100-to-50% load transient response in the closed-loop converter with the simple continuous-time integral compensator

Optional Exercises

1.1.1 Add a **Scope** block (**Simulink** → **Sinks** → **Scope** in the Library Browser) to observe the duty-cycle command d and the switching signal c waveforms.



Note: in Scope window, click on the **Parameters** button to change the **Number of axis** to 2 in the **General** tab. Also, uncheck **Limit data points to last** in the Data history tab to allow the Scope to display long waveforms

1.1.2 In the **buck_closed_loop** models, the output of the **Integrator** block can be arbitrarily large. If the compensator output “winds-up” far away from the 0-to-1 duty-cycle command range, during start-up or over-loads, the output voltage may have large overshoots or undershoots before returning to regulation. Double-click on the Integrator block to add realistic saturation limits.

Exercises

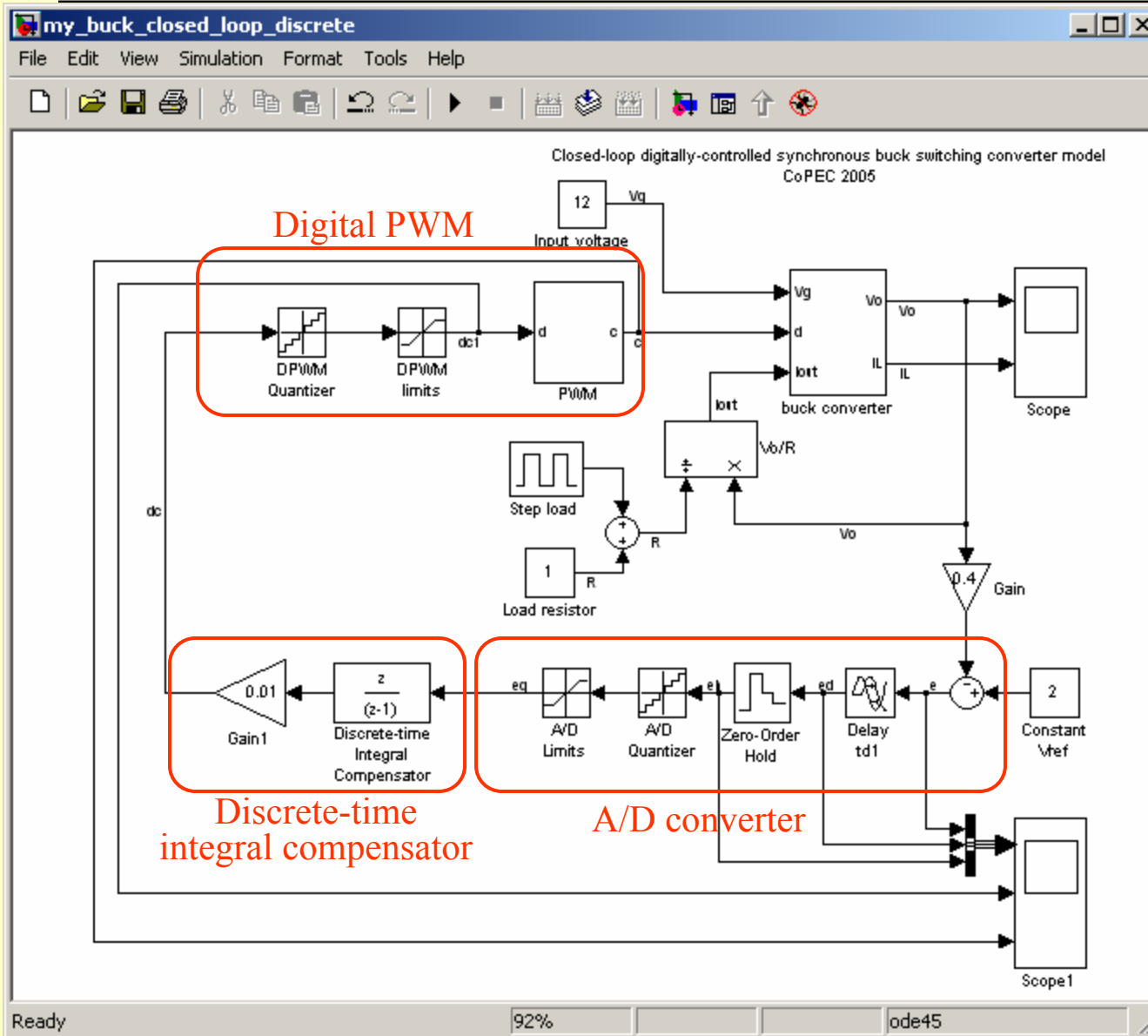
- 1.1.3 Add a **Saturation** block (**Simulink** → **Discontinuities** → **Saturation** in the Library Browser) to model limits D_{min} and D_{max} for the duty-cycle command
- 1.1.4 Change the step-load-transient model to test the closed-loop converter response under 0-to-100% (i.e. 0-to-5 A) load transients.
- 1.1.5 Add a soft-start feature to the model. Hint: a **MinMax** block (**Simulink** → **Math Operations** → **MinMax** in the Library Browser) can be used to select the minimum of two signals. Connect one of the MinMax block inputs to the duty-cycle command from the compensator, and connect the other input to a slow ramp generator (**Simulink** → **Sources** → **Ramp** in the Library Browser). In this implementation, the output voltage will still overshoot because of the duty-cycle saturates at D_{max} , which is greater than the steady-state value. Can you improve the soft-start model?
- 1.1.6 Change the model to test for a transient response to a step in the input voltage.

* Note: in MATLAB 6.1, and earlier versions, the “**Discontinuities**” section of the Simulink library was called “**Nonlinear**”

1.2 Digitally-Controlled Buck Converter: Simulink Models and Simulations

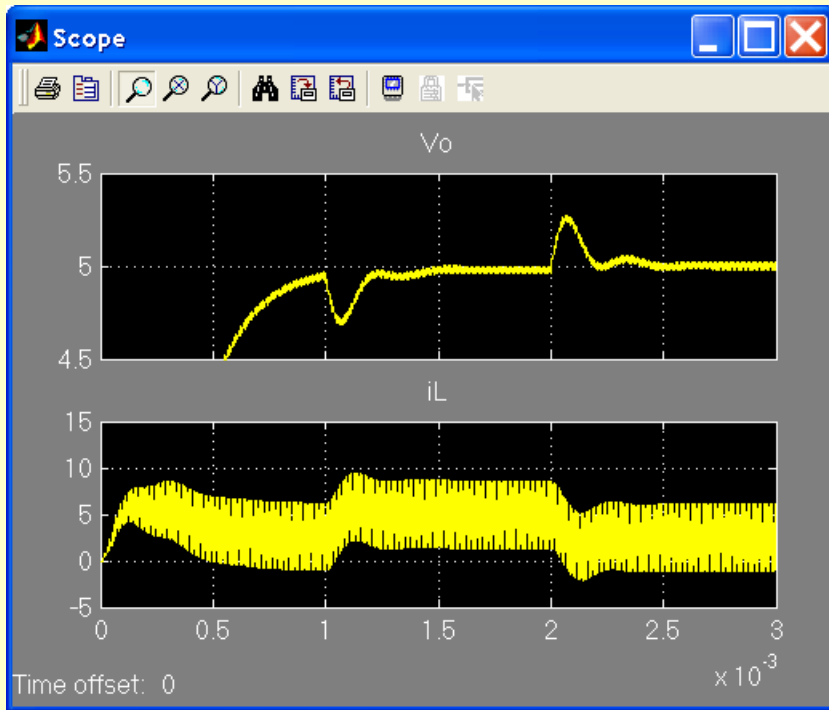
- The objective of this part is to develop and explain details of a Simulink model for a digitally-controlled buck converter, including Simulink models for:
 - A/D converter
 - Discrete-time compensator
 - Digital PWM
- The buck converter model and the parameters are the same as in Section 1.1 (same parameters as in the Simulink file: **buck_closed_loop_load.mdl**)
 - $L = 4.1 \mu\text{H}$, $R_L = 80 \text{ m}\Omega$
 - $C = 376 \mu\text{F}$, $R_{esr} = 5 \text{ m}\Omega$
 - $f_s = 100 \text{ KHz}$
 - $V_g = 12 \text{ V}$
 - Maximum load current: 5 A
- Open the file **buck_closed_loop_discrete.mdl** and save the system model as **my_buck_closed_loop_discrete.mdl**

Digitally Controlled Buck Converter Simulink Model

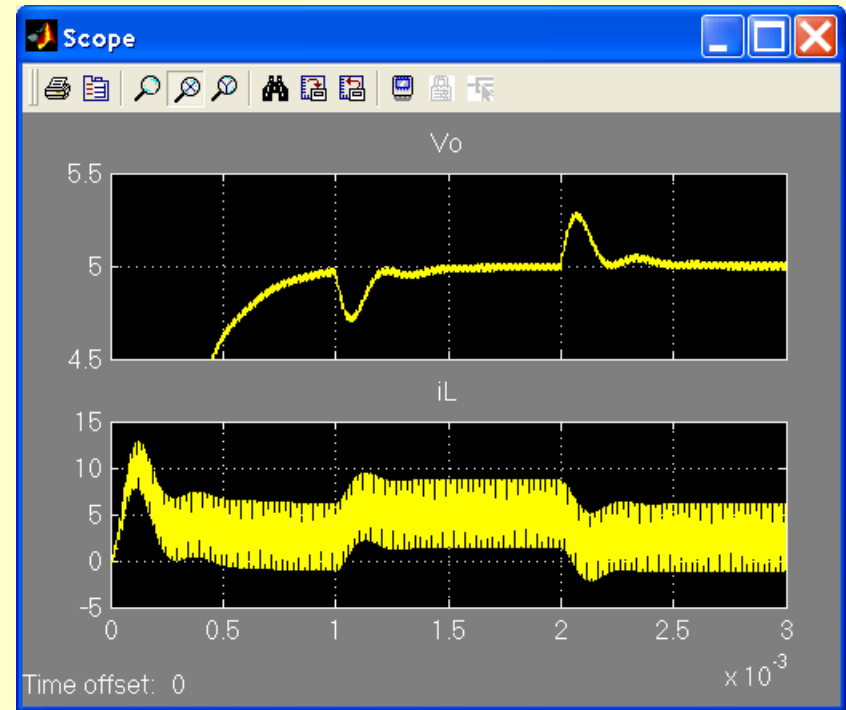


- The **buck converter** block and the step load transient model are the same as in the continuous-time **buck_closed_loop_load** system
- Note the parts of the system that model the digital controller including:
 - A/D converter
 - Discrete-time integral compensator, and
 - Digital PWM
- Run a simulation and double-click on the **Scope** block to observe the output waveforms
- The output voltage and inductor current waveforms are shown on the next page, in comparison with the waveforms obtained from the continuous-time example in **buck_closed_loop_load**

Start-Up and Step-Load Transient Waveforms



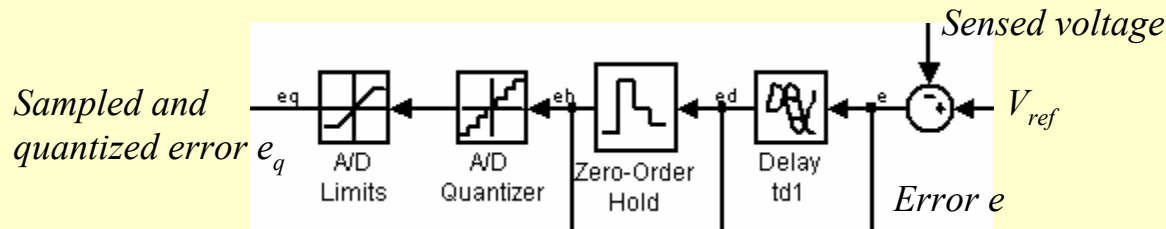
Scope waveforms in the digitally controlled converter model **buck_closed_loop_load_discrete**



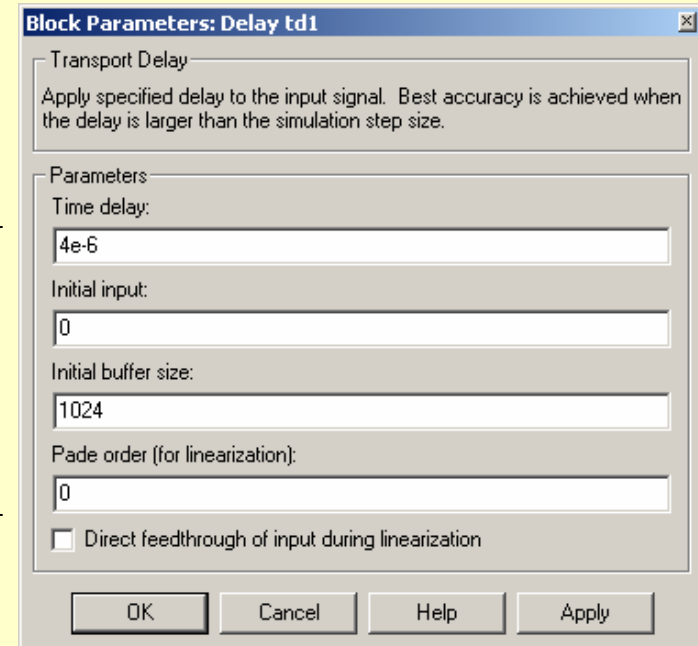
Scope waveforms in the analog controlled converter model **buck_closed_loop_load**

- Note that the transient waveforms are almost the same except for a slight difference in the inductor current waveform during start up. Where does the difference come from? Hint: add a **Scope** block to observe the sampled and quantized error signal e_q after the **A/D Limits** block during the start-up transient.
- Next, we examine details of the A/D converter, Discrete-time compensator and Digital PWM models

A/D Converter Model: Delay t_{d1}



- **Delay td1** block is a **Transport Delay** block (Simulink → Continuous → Transport Delay from the Library Browser). This block models the total time between sampling the error signal e and updating the duty cycle command d_c at the beginning of the next switching period. This delay must be long enough to include the A/D conversion time, as well as processing and computation delays in the compensator.
- Double click on the **Delay td1** block to view/change the delay t_{d1} (4 μ s in this example)
- Usually there is no need to change the default values of other parameters in this block. Optional: click on the **Help** button to see more details about the block
- Double-click on the **Scope1** block and zoom in on the waveforms following the step-load transient at 1 ms. See the annotated waveforms on the next page.



Waveform details in the digital controller

Error signals:

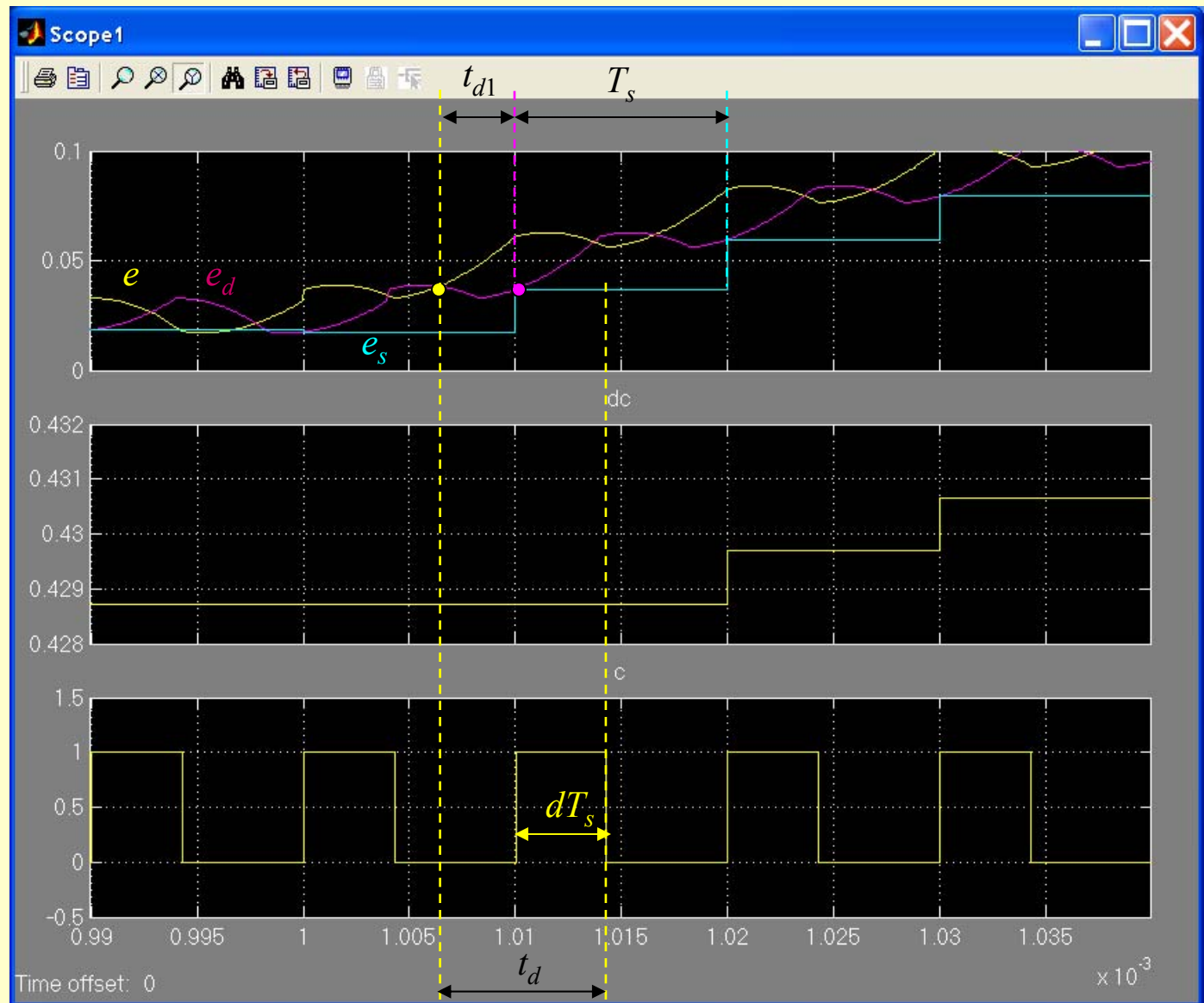
$$e = V_{ref} - V_o$$

$$e_d = e \text{ delayed by } t_{d1}$$

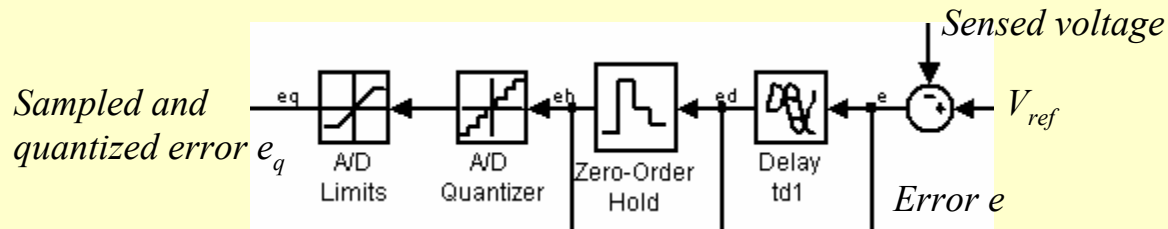
$$e_s = e_d \text{ after Zero-Order Hold}$$

Duty-cycle command d_c

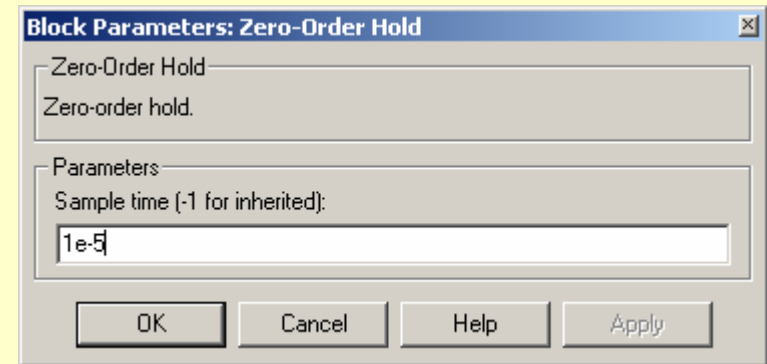
Switching signal c



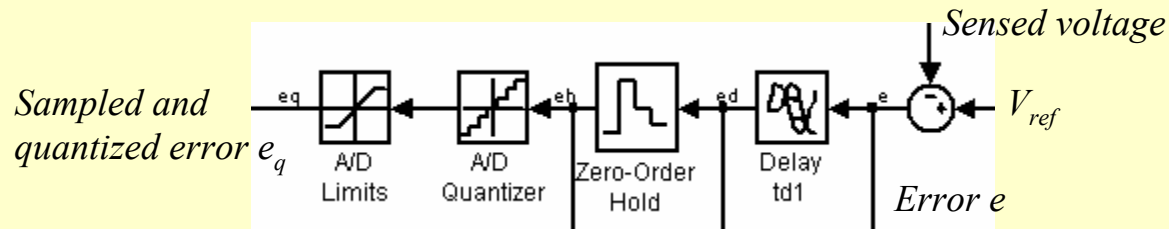
A/D Converter Model: Zero-Order Hold



- The **Zero-Order Hold** block (**Simulink** → **Discrete** → **Zero-Order Hold** in the Library Browser) samples the error signal, *i.e.* converts the signal from continuous time to discrete time
- Double click on the block to view the **Sample time** ($T_s = 10 \mu\text{s}$ in this example)
- Note that the **Sample time** is the same as the switching period T_s defined by the period of the sawtooth waveform in the **PWM** subsystem
- Observe the **Scope1** waveform e_h after the **Zero-Order Hold** block (shown earlier)
- Notice that the sampling of the delayed error signal e_d occurs at the beginning of each switching period, *i.e.* at $0, T_s, 2T_s$, etc. It coincides with the rising edge of the switching signal c .
- It is important to note that the Zero-Order Hold is added to the Simulink model only for the purpose of explicitly modeling the sampling effect. The system dynamic model does not include a ZOH transfer function

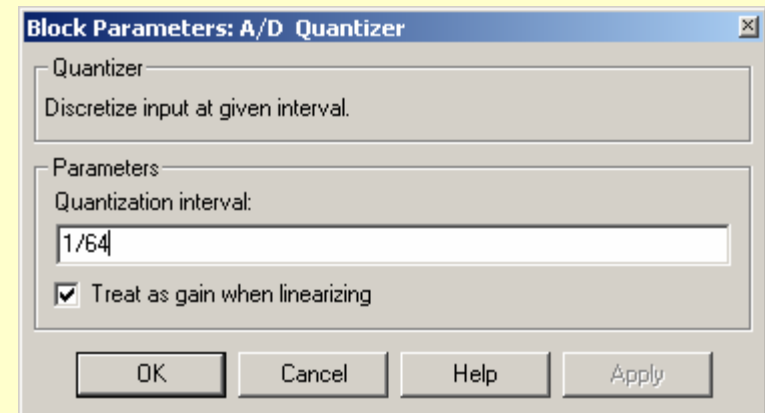


A/D Converter Model: A/D Quantizer

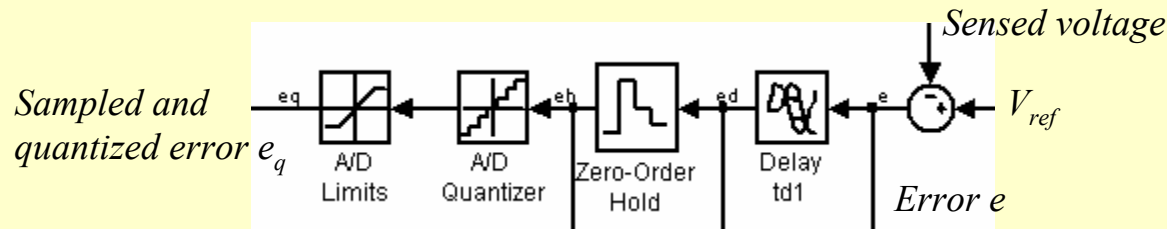


- **A/D Quantizer** block is a **Quantizer** block (Simulink → Discontinuities → Quantizer from the Library Browser).
- Double click on the block to view/change the **Quantization interval** ($q_{A/D} = 1/64$ V in this example)
- The **Quantization interval** equals the LSB value (in Volts) of the A/D converter. In this example, the A/D converter has 7-bit resolution over 2 V $\{-1V$ to $+1V\}$ voltage range. Hence, the **Quantization interval** is

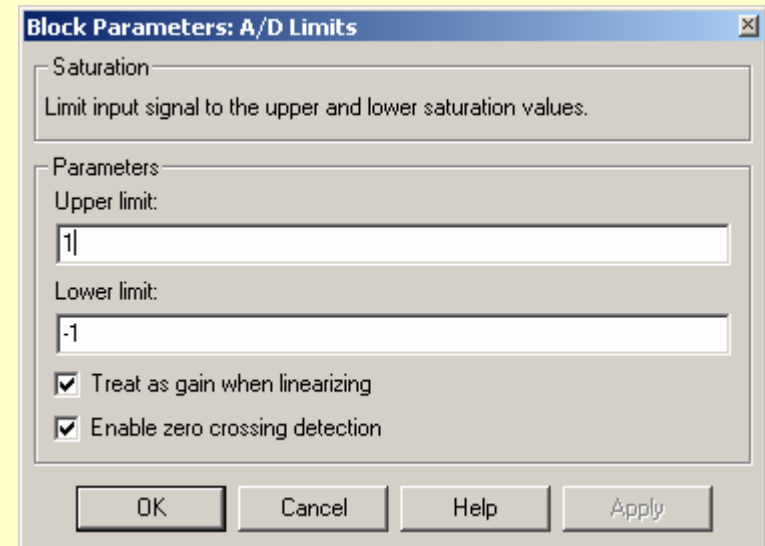
$$q_{A/D} = 2/2^7 = 1/2^6 = 1/64 = 15.6 \text{ mV}$$
- With the box **Treat as gain when linearizing** checked, the “gain” of the Quantizer block in a linearized model is 1. Otherwise, a “small-signal” gain equal to zero is assumed. Optional: click **Help** for more details



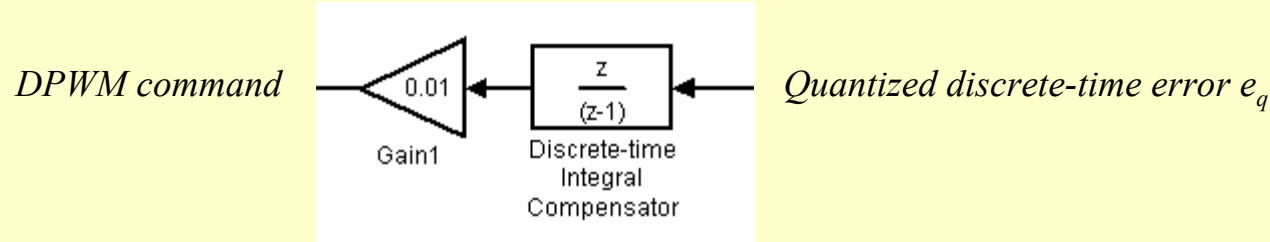
A/D Converter Model: A/D Limits



- **A/D Limits** block is a **Saturation** block (Simulink → Discontinuities → Saturation from the Library Browser).
- Double click on the block to view/change the **Upper limit** (+1 V in this example) or the **Lower limit** (–1 V in this example)
- This block models the conversion range (or window) of the A/D converter
- **Optional exercise 1.2.1:** add a **Scope** block to observe the error signals e_h and e_q before and after the **A/D Quantizer** and the **A/D Limits** blocks:
 - Note the effect of **A/D Limits** during the start-up transient; experiment with making the A/D conversion range smaller
 - Note that in steady state the quantized error e_q is exactly zero



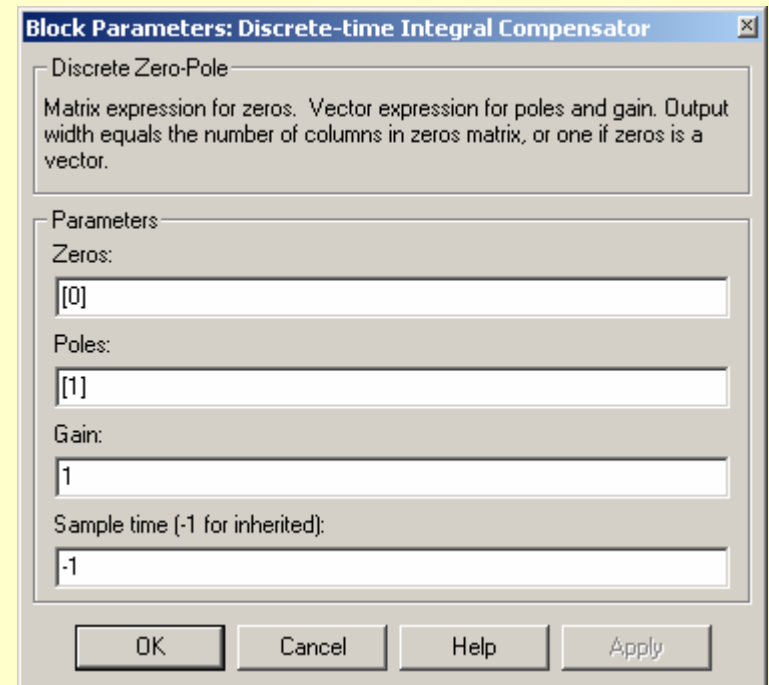
Discrete-time Integral Compensator



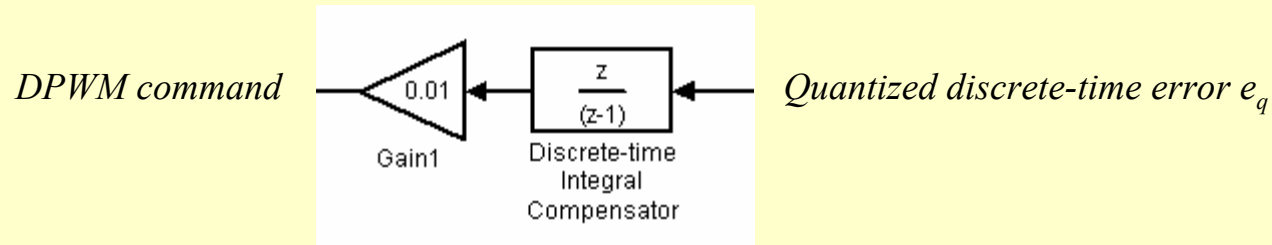
- The **Discrete-time Integral Compensator** block is a **Discrete Zero-Pole** block (Simulink → **Discrete** → **Discrete Zero-Pole** in the Library Browser)
- Double click on the block to view the block Parameters
- The block implements a discrete system transfer function in the following factored pole/zero form:

$$H(z) = K \frac{Z(z)}{P(z)} = K \frac{(z - Z_1)(z - Z_2) \dots (z - Z_m)}{(z - P_1)(z - P_2) \dots (z - P_n)}$$

- The Parameters are specified as follows:
 - **Zeros:** [Z1 Z2 Zm] ([0], in this example)
 - **Poles:** [P1 P2 Pn] ([1] in this example)
 - **Gain:** K (1 in this example)
 - **Sample time** (“-1 for inherited” means that the sample time is inherited from the **Zero-Order Hold** block in this example)
- Note that a discrete-time integrator with no delay is implemented in this example

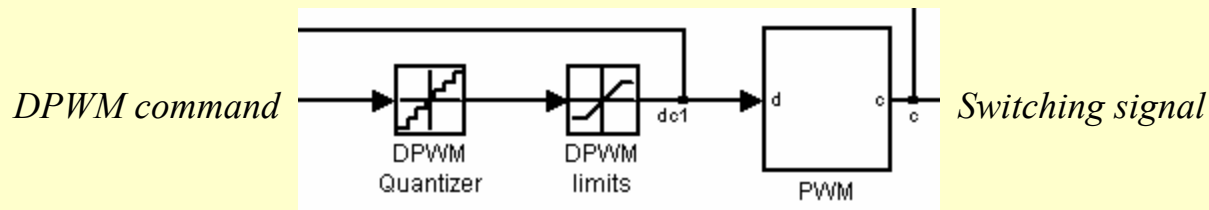


Discrete-time Compensator



- There are a number of other ways to implement a discrete transfer function (see **Simulink** → **Discrete** blocks in the Library Browser), such as one or a combination of the following blocks:
 - **Discrete Transfer Fcn** block
 - **Discrete Filter** block
 - **Discrete-Time Integrator** block
 - A combination of **Unit Delay**, **Gain** and **Sum** blocks
- **Exercise 1.2.2:** the parameter of the **Gain1** block is set to 0.01 to match the gain of the discrete-time integral compensator to the gain of the continuous-time integral compensator in the **buck_closed_loop** model. Show that this is true.
- **Exercise 1.2.3:** implement the compensator using the **Discrete-Time Integrator** block from the Library Browser. What should **Gain1** be in this case? Verify your implementation by simulation. Is this implementation completely equivalent to the implementation shown above? Why not?
- **Exercise 1.2.4:** implement the discrete-time integral compensator using a **Unit Delay** block and a **Sum** block. Verify your implementation by simulation.

Digital PWM



- Digital PWM model includes a **DPWM Quantizer** block (a **Quantizer** block), a **DPWM limits** block (a **Saturation** block) and the analog **PWM** subsystem block
- Double-click on the DPWM Quantizer to view/change the **Quantization interval** parameter, *i.e.* the LSB value of the duty cycle. In this example, the DPWM resolution is 10 bits over the 0-to-1 range, so that the **Quantization interval** is $q_{DPWM} = 1/2^{10} = 1/1024$
- Double-click on the **DPWM limits** block to view/change:
 - **Upper limit** (*i.e.* the maximum duty cycle), 0.8 in this example, or
 - **Lower limit** (*i.e.* the minimum duty cycle), 0.0 in this example
- **Exercise 1.2.5:** change the DPWM resolution to 7 bits and run a simulation. Low-frequency oscillations can be observed in the output voltage. Why? What is, approximately, the frequency of the oscillations? Compare this frequency to the corner frequency f_o of the buck converter LC filter.

Exercises

- 1.2.7** In the **buck_closed_loop_discrete** models, the output of the integral compensator can be arbitrarily large. If the compensator output “winds-up” far away from the 0-to-1 duty-cycle command range, during start-up or over-loads, the output voltage may have large overshoots or undershoots before returning to regulation. Change the model to add realistic saturation limits for the integral compensator. Hint: do Exercise **1.2.4** first.
- 1.2.6** Change the step-load-transient model to test the closed-loop converter response under 0-to-100% (i.e. 0-to-5 A) load transients.
- 1.2.7** Add a soft-start feature to the model.
- 1.2.8** Change the model to test for a transient response to a step in the input voltage.