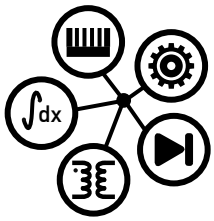

Embedded
Code Generation

DEMO MODEL

H-Bridge Converter

Current control of an H-bridge converter with embedded code generation for TI C2000 MCUs

Last updated in C2000 TSP 1.3.1

www.plexim.com

- Request a PLECS and PLECS Coder trial license
- Get the latest TI C2000 and RT Box Target Support Package
- Check the PLECS, RT Box and TI C2000 TSP documentation

1 Overview

This demo model features a current-controlled H-bridge circuit powering an inductive load. It provides an explanation of the typical workflow of the PLECS Embedded Coder, using Texas Instruments (TI) C2000 microcontrollers (MCUs). Combined with a PLECS RT Box, the performance of the MCU can be verified directly.

The model is split into two distinct subsystems called “Plant” and “Controller”. The plant contains an H-bridge converter with inductive load, and the controller employs a proportional-integral (PI) current control scheme. Each subsystem is deployed to a separate real-time target. The control logic in the controller subsystem is built and then flashed to a TI C2000 MCU. The plant subsystem is deployed on the PLECS RT Box for Hardware-in-the-loop (HIL) testing of the generated embedded code. The following sections provide a brief description of the model and instructions on how to simulate it.

The demo model is verified with a hardware prototype of the power circuit, comprising two phases of a BOOSTXL-DRV8305EVM BoosterPack [1] connected to an external RL load.

Note This model contains model initialization commands that are accessible from:

PLECS Standalone: The menu **Simulation + Simulation Parameters... + Initializations**

PLECS Blockset: Right click in the **Simulink model window + Model Properties + Callbacks + InitFcn***

---

2 Model

The top level schematic contains two separate subsystems representing the controller and plant models, as shown in Fig. 1. Both subsystems are enabled for code generation from the **Edit + Subsystem + Execution settings...** menu. This step is necessary to generate the model code for a subsystem via the PLECS Coder.

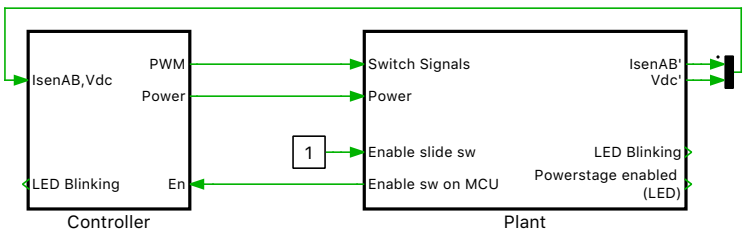

Figure 1: Top level schematic of the plant and the controller subsystems

2.1 Power Circuit

The power circuit, shown in Fig. 2, is supplied by a DC source voltage of $V_{\mathrm{dc}} = 24\,\mathrm{V}$. The H-bridge is composed of two IGBT Half Bridge power module components, powering an inductive load. The pulse-width modulated (PWM) switching signals are obtained from the PWM Capture block of the PLECS RT Box library. Further detail on the power module components and the sub-cycle averaging of PWM signals is described in [2]. The DC input voltage and output inductor current measurements are connected to Analog Out blocks from the PLECS RT Box library. The discretization step size of the plant subsystem is set to $2\,\mu\mathrm{s}$.

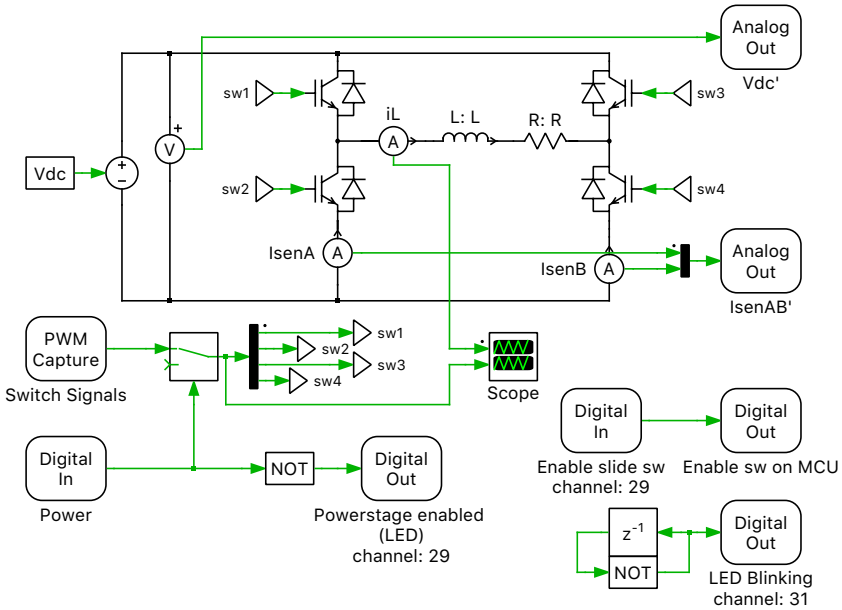

Figure 2: Power circuit of the H-Bridge with inductive load

Scaling Analog Outputs

The analog input and output voltage ranges of the PLECS RT Box target are limited between $-10\,\mathrm{V}$ and $10\,\mathrm{V}$, as set the **Target** tab of the **Coder options...** window.

However, since the Analog Out pins of the RT Box are connected to an embedded controller, these values are scaled and offset to be within $0\,\mathrm{V}$ to $3.3\,\mathrm{V}$, to satisfy voltage limits of the MCU analog-to-digital converters (ADCs).

Since the plant model is based on a power circuit prototype as described in Section 4, the input voltage and the shunt currents are scaled according to the voltage and current sense circuits of the BOOSTXL-DRV8305EVM BoosterPack.

Voltage Sensing

A similar voltage sensing circuit as described in [1] is shown in Figure 3.

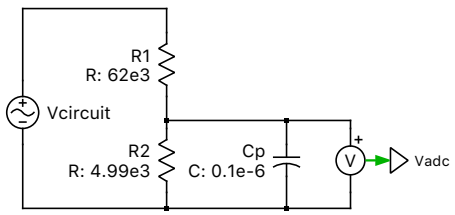

Figure 3: Schematic of a voltage sensing circuit

The equations below show the response of the voltage sensing circuit. A capacitor, C_p can be selected to create a pole in the sensing circuit to attenuate the switching frequency ripple on the sensed voltage. The pole frequency is denoted as f_p below.

$$\frac{\mathrm{V_{adc}}}{\mathrm{V_{circuit}}} = \frac{\mathrm{R_2}}{(\mathrm{R_1}+\mathrm{R_2}) + s\mathrm{R_1R_2C_p}},\ \mathrm{C_p} = \frac{\mathrm{R_1}+\mathrm{R_2}}{2\pi \mathrm{f_p R_1 R_2}}$$

The voltages are scaled with a gain of,

$$\mathrm{Vm.K} = \frac{\mathrm{R}_2}{\mathrm{R}_1 + \mathrm{R}_2} = \frac{4990}{62000 + 4990}$$

Current Sensing

As described in [1], low-side shunt currents are sensed for both of the half bridges. These currents are sensed and amplified with a scaling factor of $Im \cdot K$ and offset with offset factor of $Im \cdot O$ using a current shunt amplifier.

$$\mathrm{Im.K} = 0.007 * 10$$

$$\mathrm{Im.O} = \frac{3.3}{2}$$

2.2 Controls

The controller subsystem is shown in Fig. 4. The "Right Leg Duty-Cycle" subsystem determines the duty cycle required to maintain an average of $12\,\mathrm{V}$ at the right leg H-bridge output while accounting for variations in the sensed input voltage, V_{dc}.

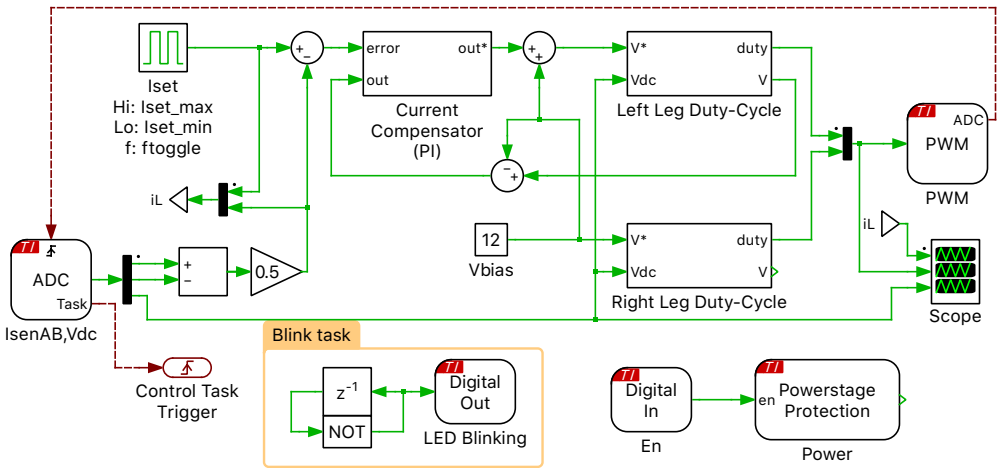

Figure 4: Controller of the H-Bridge circuit

The "Left Leg Duty-Cycle" subsystem determines the modulation index of left leg of the H-bridge based on a proportional-integral (PI) controller, as shown in Fig. 5. The sensed inductor current is compared to a setpoint that is toggled between $-3\,\mathrm{A}$ and $3\,\mathrm{A}$. This error is used for current compensation by the digital PI controller, which is equipped with anti-windup logic.

A few of the concepts described in this Section 2.2, related to the controller development, are based on reference [3].

Plant transfer function

To set the PI controller gain parameters a plant transfer function $P(s)$ is needed. $P(s)$ relates the change of the voltage across the inductor (L) and resistor (R), V_{RL} (the input variable), to the response of the inductor current I_{L} (the output variable):

$$P(s) = \frac{I_{\mathrm{L}}}{V_{\mathrm{RL}}} = \frac{1/R}{1 + s\,L/R} = \frac{K_1}{1 + s\,T_1}, \quad \text{where } K_1 := 1/R \text{ and } T_1 := L/R$$

Time constants

For reducing the order of complex systems, it is often useful to distinguish between dominant time constants and small time constants. Typically, the dominant time constants are a part of the plant transfer function, while the control system introduces several small time constants (e.g. sensors, actuators, sampling, calculation delays, fast inner control loops).

If the largest small time-constant is at least 4 times smaller than the smallest dominant time constant, i.e.,

$$min(T_{\mathrm{dominant}}) \geq 4 \cdot max(T_{\mathrm{small}})$$

then it is possible to make an important simplification for modeling of the system, which is to define one equivalent small time constant which is the sum of all small time constants in the system.

If the controller sampling frequency ($1/T_{\mathrm{s}}$) is same as the switching frequency ($1/T_{\mathrm{sw}}$), the small time constants present in the H-bridge converter model for this specific implementation are:

- small time constant for control calculation, T_{calc}, is $\frac{1}{2}\,T_{\mathrm{sw}}$
- small time constant for PWM output generation, T_{pwm}, is $\frac{1}{2}\,T_{\mathrm{sw}}$
- small time constant for conversion of continuous parameters to discrete, T_{sh}, is $\frac{1}{2}\,T_{\mathrm{sw}}$ (explained in the section below)

The equivalent small time constant, T_{Σ}, is

$$T_{\Sigma} = T_{\mathrm{calc}} + T_{\mathrm{pwm}} + T_{\mathrm{sh}}$$

$$D_{\Sigma}(s) = \frac{1}{1 + sT_{\Sigma}}$$

Calculation of control parameters

The control parameters of the PI controller (K_{p} and K_{i}) are calculated using the Magnitude Optimum Criterion. The system's open-loop transfer function $H_{\mathrm{OL}}(s)$ is given by the product of transfer functions from the controller, plant and time delays:

$$H_{\mathrm{OL}}(s) = \frac{1 + sT_{\mathrm{n}}}{sT_{\mathrm{i}}} \cdot \frac{K_1}{1 + s\,T_1} \cdot \frac{1}{1 + sT_{\Sigma}}, \quad \text{where } K_{\mathrm{p}} = \frac{T_{\mathrm{n}}}{T_{\mathrm{i}}} \text{ and } K_{\mathrm{i}} = \frac{1}{T_{\mathrm{i}}}$$

The controller parameter T_{n} is chosen, such that the pole of the plant transfer function is canceled, i.e. $T_{\mathrm{n}} = T_1$. After pole-zero cancellation, the closed-loop transfer function is a second order system. The remaining parameter T_{i} is calculated from setting the damping factor (ζ) of the second order system to $1/\sqrt{2} = \zeta$, resulting in $T_{\mathrm{i}} = 2K_1T_{\Sigma}$

Transformation of continuous control parameters to discrete

The digital implementation of a PI regulator can be written as

$$y[k] = K_{\mathrm{pz}}\,e[k] + K_{\mathrm{iz}} \sum_{i=0}^{k} e[i]$$

If the sample time (T_{s}) is sufficiently less than the smallest time constant of the PI regulator, i.e.

$$T_{\mathrm{s}} \leq \frac{T_{\mathrm{n}}}{2}$$

then the sum $\sum_{i=0}^{k} e[i]$ can be approximated by an equivalent transfer function:

$$\sum e[i] \approx \frac{1 + sT_{\mathrm{s}}/2}{sT_{\mathrm{s}}}$$

With this, the following relationship between the continuous and discrete PI regulator can be obtained:

$$K_{\mathrm{pz}} = \frac{T_{\mathrm{n}} - T_{\mathrm{s}}/2}{T_{\mathrm{i}}}$$

$$K_{\mathrm{iz}} = \frac{T_{\mathrm{s}}}{T_{\mathrm{i}}}$$

However, the sample & hold at the input of a digital regulator adds an additional delay which must be taken into account during the continuous time design:

$$SH(s) = \frac{1}{1 + sT_{\mathrm{s}}/2}$$

Anti-windup

When the actuator of a control system reaches its limits, or saturates, the plant ceases to respond to changes in the control output and the control system cannot reduce the control error. Under these conditions it is important to prevent integrators in the controller from reaching extreme values, referred to as windup. One approach to prevent windup is to keep the integrator, *I* in Fig. 5, at the same value as the saturated output, *y'*. This ensures that when the error (eventually) decreases below zero, changes in the integrator output (*i*) will not be limited by controller saturation.

In saturation:

$$I = y'$$

$$i = 0$$

Therefore, the anti-wind up corrective gain (K_{c}) can be calculated as:

$$i = K_{\mathrm{iz}} \cdot e - K_{\mathrm{c}}(I + e(K_{\mathrm{pz}} + K_{\mathrm{iz}}) - y') = 0$$

$$i = K_{\mathrm{iz}} \cdot e - K_{\mathrm{c}}(y' + e(K_{\mathrm{pz}} + K_{\mathrm{iz}}) - y') = 0$$

$$K_{\mathrm{c}} = \frac{K_{\mathrm{iz}}}{K_{\mathrm{pz}} + K_{\mathrm{iz}}}$$

The overall structure of the PI controller including anti-windup is given in Fig. 5.

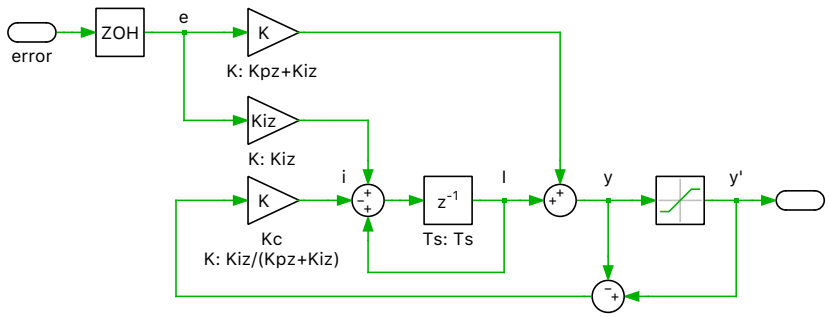

Figure 5: Proportional-Integral controller with anti-windup

Configuring TI C2000 Target library components

The controller in Fig. 4 contains several components from the TI C2000 Target library. The gate signals are generated by the PWM component, where the left and right leg duty cycles are inputs to the PWM block. The duty cycles are in the range of [0, 1] for generating gate signals. The **Carrier type**, **Carrier frequency** and **Blanking time** parameters can be configured from the **Main** tab of the PWM block parameters window. Please note that the input to the ePWM module generates a complementary PWM pair, i.e. ePWMxA and ePWMxB will be opposite polarity excluding the blanking time.

The measurements of the input voltage and inductor current are introduced to the model environment from the ADC block of the TI C2000 Target component library. Scaling and offset factors are provided to each channel via the parameter window of the ADC block in order to convert the detected analog voltage into values with physical units to be used for the control algorithm. The **ADC unit** and the **Analog input channel** parameters can be modified according to the available resources of different MCUs.

In this model, the interrupt sequence of the embedded application is defined explicitly by connecting trigger signals between the PWM generator, the ADC, and the Control Task Trigger. Trigger signals are shown as red dashed lines. From the **Events** tab of the PWM block parameter window, the **ADC Trigger** parameter is configured as `Overflow`. This means the first ePWM module associated with this block generates a start-of-conversion signal for the ADCs whenever the carrier value reaches its maximum. Then the **Trigger source** parameter of the ADC block is set to "Show trigger port". Next the trigger port of the ADC block is connected to the ADC trigger from the PWM block.

The control task is executed after the last conversion on ADC module. This is configured by connecting the Task output of the ADC to the Control Task Trigger block from the TI C2000 Target component library.

The ADC and Task trigger schemes are shown in Fig. 6. In this model, the lower switch will be closed when ePWMxB is high. The ADC will measure the shunt current only when there is current flowing through the low side switch.

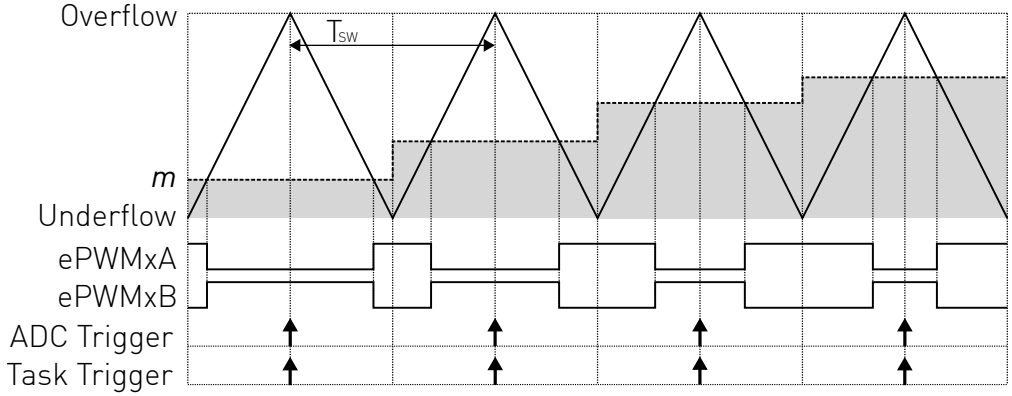

Figure 6: PWM carrier and ADC & Task trigger schemes

In order to enable or disable PWM signals during runtime, DIP switch "DI-29" on the RT Box LaunchPad Interface board is used. This input signal "DI-29" is connected to the Digital in block labeled "Enable slide sw" in the "Plant" subsystem, which is then routed as the input of the Powerstage Protection block on the "Controller" subsystem through the RT Box LaunchPad Interface board. The Powerstage Protection block implements a finite state machine to enable or disable all PWM outputs on the target MCU. A logic low to high transition enables the PWM outputs, while a high to low transition disables them. For more details, please browse the **Help** section of this block.

When the power stage is enabled a digital output, configured in the **Powerstage enable GPIO number** of the Powerstage Protection block, is toggled. This signal is connected to the Digital In block labeled "Power" in the "Plant" subsystem. This allows the captured PWM signals to pass to the gates of the inverter bridge modeling a gate driver enable circuit. The red LED "DO-29" on the LaunchPad Interface board will turn on, visually indicating the switching signals are connected to the gate drivers.

3 Simulation

In addition to running a simulation of this demo model in offline mode on a computer, the "Controller" subsystem can be directly converted into target specific code for the TI LaunchPads. The model is configured by default for a TI 28069 LaunchPad [4], but other LaunchPad targets are supported as explained later in this section.

Follow the instructions below to upload the “Controller” subsystem to a TI MCU.

- Connect the MCU to the host computer through a USB cable.
- From the **System** tab of the **Coder + Coder options...** window, select “Controller”.
- Next, from the **Target** tab, select the appropriate target from the dropdown menu. Then under the **General** sub-tab, select the desired **Build type**.
- Then, to `Build and program` the MCU directly from PLECS, choose either `Run from Flash` or `Run from RAM` as the **Build configuration** to program the MCU either to flash memory or to RAM respectively, then select `LaunchPad` as the **Board** type, and click **Build**.

If programmed correctly, LED “D9” (or the LED corresponding to GPIO “DO_DSP_LED” listed in the model initialization commands) should blink.

For advanced users who are familiar with Code Composer Studio, there is an option to `Generate code into CCS project`. Locate the appropriate `cg` folder from the CCS project (refer to [8] for step-by-step instructions), enter its path into the **CCS project directory** field and click **Build**. The code of the “Controller” subsystem will be automatically generated. Then, proceed to build and debug the project as a normal CCS project.

Note If using the RT Box LaunchPad Interface board, make sure that the **RST** jumper is open throughout the simulation.

Prior to controlling a real power stage with the programmed MCU, it is highly recommended to first verify the behavior of the controller using a PLECS RT Box and perform a hardware-in-the-loop (HIL) test. A typical hardware configuration is shown in Fig. 7, where the evaluation kit, a TI 28069 LaunchPad (the red board), is connected to the RT Box via an RT Box LaunchPad Interface (the green board).

Figure 7: Hardware setup of the HIL verification

Follow the instructions below to run a real-time model on the RT Box.

- From the **System** tab of the **Coder + Coder options...** window, select “Plant”. Click the **Target** tab and select a target device. Then click **Build** to deploy the model to the target RT Box.
- Once the model is uploaded, from the **External Mode** tab of the **Coder options...** window, **Connect** to the RT Box and **Activate autotriggering** to observe the test results in real-time.

If programmed correctly, the LED corresponding to “DO-31” of the RT Box LaunchPad Interface board should blink.

Toggle the switch “DI-29” on the RT Box LaunchPad Interface board from low to high to enable the MCU, as explained at the end of Section 2.2. When the power stage is enabled, the LED corresponding

to “DO-29” of the LaunchPad interface board should turn on. Observe the real-time waveforms in the Scope of the “Plant” subsystem.

Note At this stage, verify that the LED corresponding to “DO-29” on the RT Box LaunchPad Interface board is turned on.

Toggling the switch “DI-29” on the RT Box LaunchPad Interface board from high to low should disable all the gating signals. “DO-29” of the LaunchPad Interface board should turn off. Toggling “DI-29” back to high will enable the PWM outputs once again.

In order to tune the parameters of the control program in the MCU and observe any intermediate values, follow the instructions below to connect to the external mode of the TI MCU.

- First, **Disconnect** the “Plant” subsystem from the **External Mode** of the PLECS RT Box, if connected.
- Then, from the **System** menu on the left hand side of the **Coder + Coder options...** window, select “Controller”.
- Next, from the **External Mode** tab, select the appropriate **Target device** and click **Connect**.
- Then, **Activate autotriggering** to observe the test results in the “Controller” subsystem Scope.

The inductor current measurements can be viewed using the scope found in the “Plant” subsystem as well as the scope in the “Controller” subsystem. The inductor current reference is toggled between $-3\,\mathrm{A}$ and $3\,\mathrm{A}$ using the “Iset” (Pulse Generator) component of the “Controller” subsystem. These reference values can be changed on the fly, in real-time, since the “Iset” component has been added to the "Exceptions" list found in the **Parameter Inlining** tab of the **Coder options...** window, prior to building the model.

The step response of the inductor current in real-time is shown in Fig. 8.

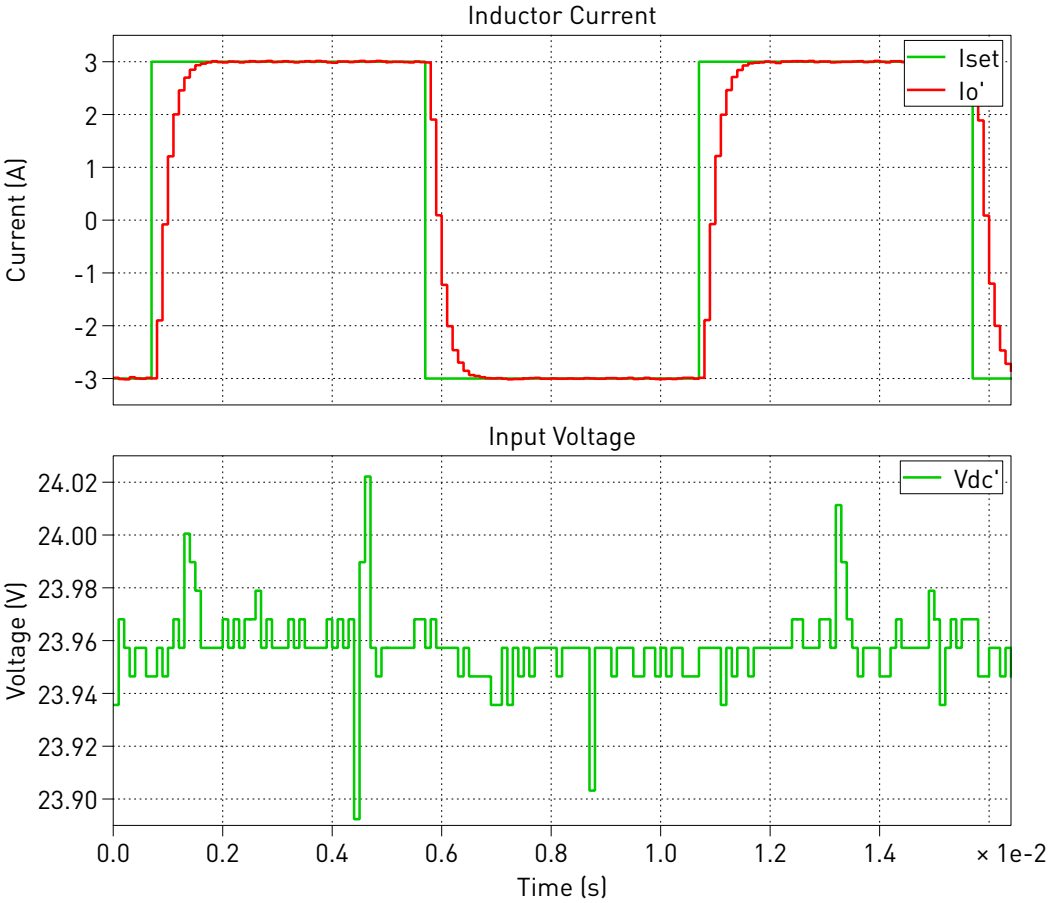

Figure 8: Inductor current and input voltage measurements in real-time using the PLECS RT Box 1

A trigger control for a desired **Trigger channel** can be set from the **External Mode** tab.

Please note that the I/O configuration of all the peripheral blocks (ADC, PWM) are configured by mapping to the TI 28069 LaunchPad [4]. For a TI MCU other than the TI 28069 LaunchPad, the I/O configuration has to be adapted. In addition to the TI 28069 LaunchPad, this demo model also supports code generation for other TI C2000 MCUs, such as the TI 28377S [5], 28379D [6] and 280049C LaunchPads [7]. From the **Model initialization commands** window of **Simulation Parameters... + Initializations** tab from the **Simulation** menu, change the value of `type_evm`, to choose the desired target. You must also configure the corresponding **Target** in the **Coder Options** window accordingly.

4 Power Circuit Prototype

A hardware prototype of the power circuit, comprising two phases of a BOOSTXL-DRV8305EVM BoosterPack [1] connected to an external RL load, is used to verify the circuit. The GPIOs listed in the demo model for the 28069 LaunchPad are fully compatible with the BoosterPack.

After deploying the code and connecting to the 28069 LaunchPad target, as described in 3, the current and voltage measurements can be viewed using the scope found in the “Controller” subsystem, as shown in Fig. 9.

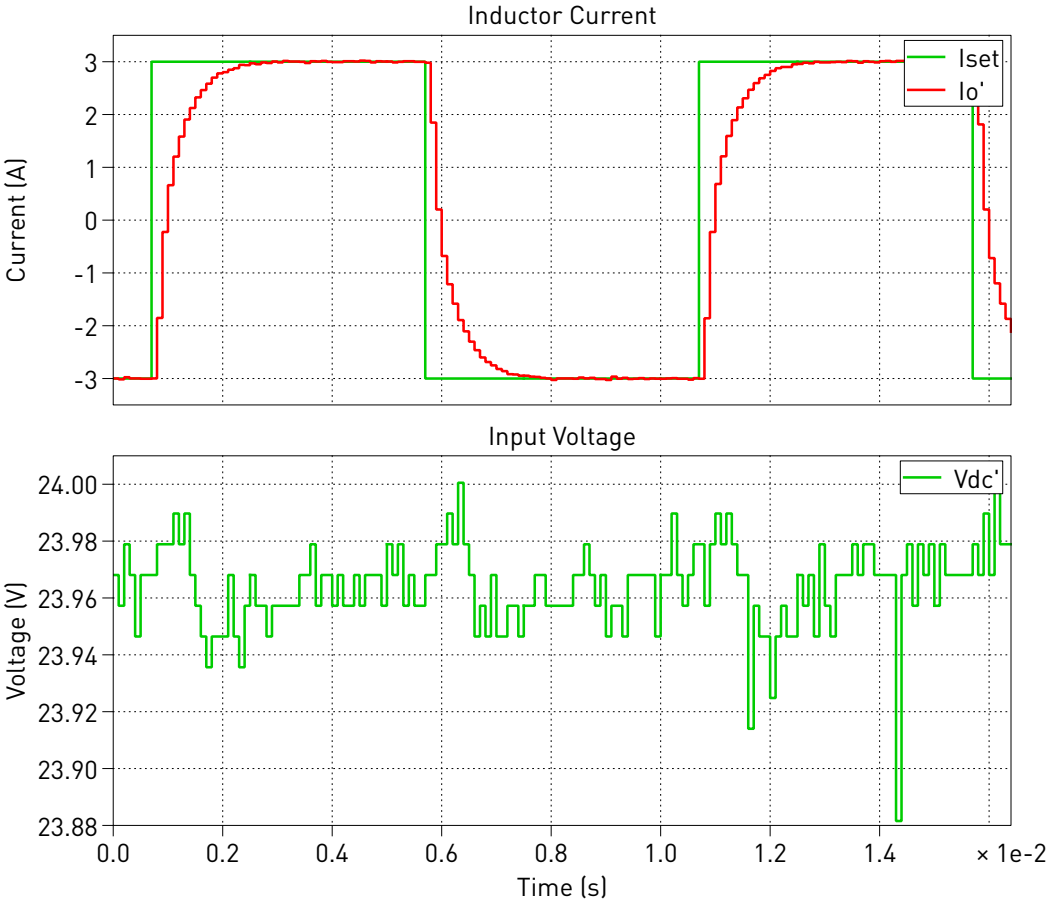

Figure 9: Inductor current and input voltage measurements of the power circuit prototype

Notice how the real-time simulation results using the PLECS RT Box match the results with the hardware prototype.

5 Conclusion

This model demonstrates an H-bridge converter with a discrete current controller that supports embedded code generation for TI C2000 MCUs. It can be run in both offline mode, as well as in real-time. The model has also been verified with a power circuit prototype. The model also demonstrates the **Parameter Inlining** feature using a current controller reference that can be changed in real-time.

References

[1] TI DRV8305N 3-Phase Motor Drive BoosterPack Evaluation Module
URL: http://www.ti.com/tool/BOOSTXL-DRV8305EVM.

[2] J. Allmeling, and N. Felderer, “Sub cycle average models with integrated diodes for real-time simulation of power converters,” *IEEE Southern Power Electronics Conference (SPEC), 2017*.

[3] Conception de SystÃÍmes automatiques, Hansruedi BÃijhler, Presses Polytechniques Romandes, Lausanne 1988, ISBN 2-88074-149-1

[4] TI C2000 Piccolo MCU F28069M LaunchPad Development Kit,
URL: http://www.ti.com/tool/LAUNCHXL-F28069M.

[5] TI C2000 Delfino MCUs F28377S LaunchPad Development Kit,
URL: http://www.ti.com/tool/LAUNCHXL-F28377S.

[6] TI C2000 Delfino MCUs F28379D LaunchPad Development Kit,
URL: http://www.ti.com/tool/LAUNCHXL-F28379D.

[7] TI C2000 Piccolo MCU F280049C LaunchPad Development Kit
URL: http://www.ti.com/tool/LAUNCHXL-F280049C.

[8] PLECS TI C2000 Target Support User Manual,
URL: https://www.plexim.com/download/documentation.

Revision History:

C2000 TSP 1.1	First release
C2000 TSP 1.3.1	Turn on Assertions in the two IGBT Half Bridge power modules

How to Contact Plexim:

☎	+41 44 533 51 00	Phone
	+41 44 533 51 01	Fax
✉	Plexim GmbH Technoparkstrasse 1 8005 Zurich Switzerland	Mail
@	`info@plexim.com`	Email
	`http://www.plexim.com`	Web

Embedded Code Generation Demo Model