Embedded
Code Generation

DEMO MODEL

Buck Converter BOOSTXL

Rapid control prototyping with embedded code generation for TI C2000 MCUs

Last updated in C2000 TSP 1.5.3

1 Overview

This demo model shows how to control a real synchronous buck converter using a reference design kit from Texas Instruments [1]. The TI C2000 MCU is directly programmed from PLECS using the code generated from the controller subsystem. The buck converter supports a dynamic load and can be operated with open-loop or closed-loop control. Different TI C2000 LaunchPads can be used to run the generated code.

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 model, as shown in Fig. 1. The 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.

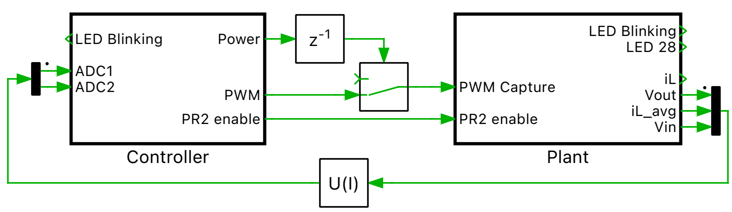

Figure 1: Top-level schematic of the plant and the controller subsystem

2.1 Power Circuit

The power circuit in Fig. 2 corresponds to the schematic of the BOOSTXL-BUCKCONV BoosterPack from TI. It is supplied by a DC source voltage of $V_{\mathrm{DC}} = 9\,\mathrm{V}$.

Resistor R4 is used to sense the current of inductor L1. The transfer characteristic of the op-amp U2 is implemented using the Gain block U2. The RC low-pass filter of the “iL_avg” signal is modeled as a first-order transfer function. A static load PR1 is permanently connected between the buck converter output and ground. A dynamic load PR2 can be switched on or off by controlling the switch Q1. When PR2 is turned on, the output load increases, which causes the inductor current to stay positive and non-zero. This capability provides an active load feature at run time to support transient performance testing.

Since the plant model is based on a reference design from TI, the input voltage, output voltage and load inductor current are scaled according to the voltage and current sense circuits. Input and output voltage sensing are both based on resistive voltage dividers.

2.2 Controls

The controller subsystem is shown in Fig. 3. The key interface signals between the BOOSTXL-BUCKCONV, RT Box and the supported TI LAUNCHXL evaluation boards are described in Table 1.

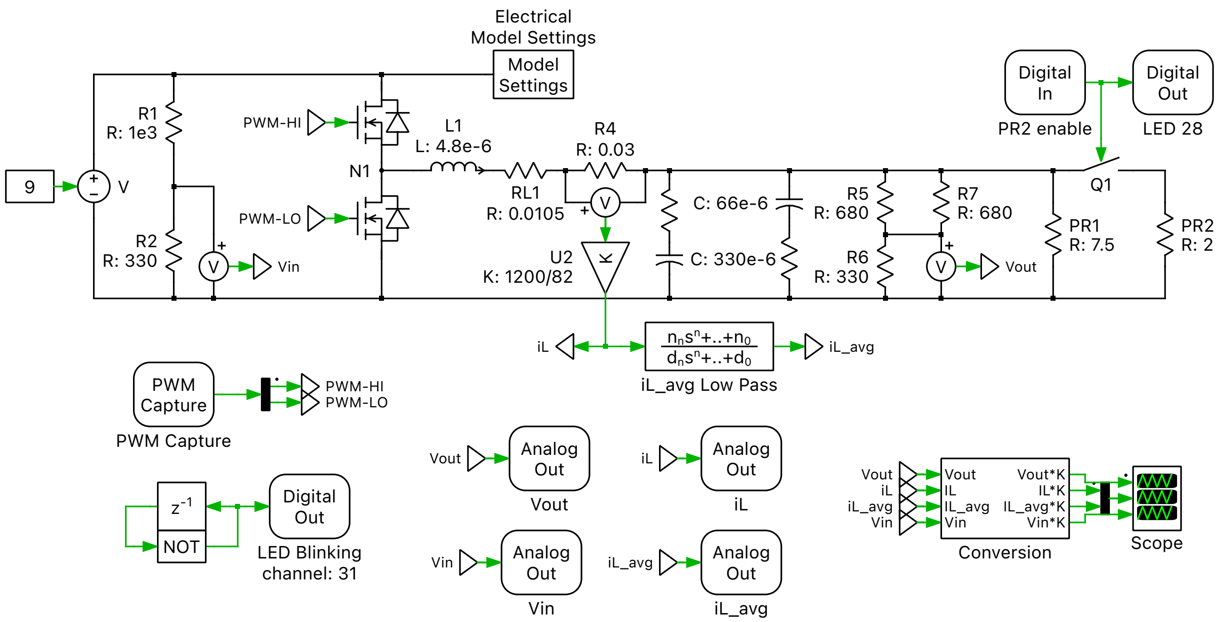

Figure 2: Power circuit of the buck converter with dynamic load

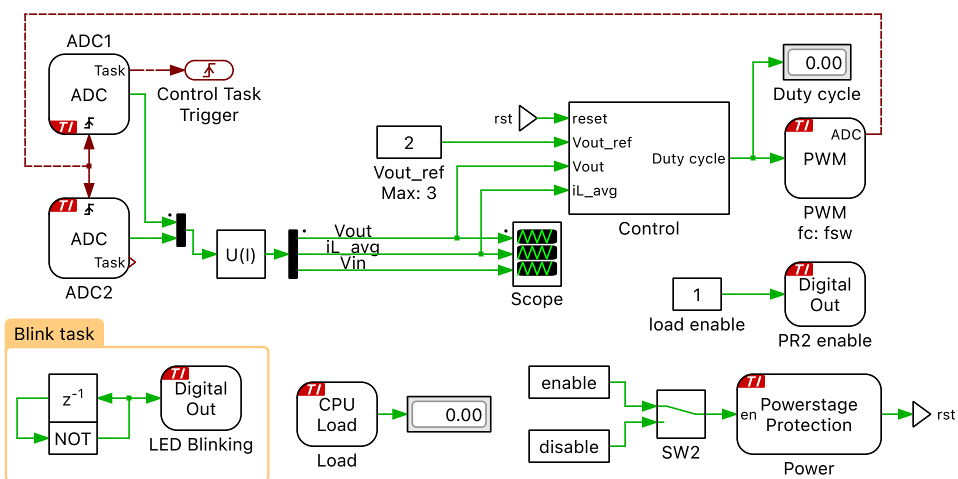

Figure 3: Controller of the BOOSTXL-BUCKCONV BoosterPack

Open loop

When the configuration of the Control subsystem is set to “Open_loop”, the buck converter output voltage is controlled using a pre-determined duty cycle without a feedback loop. The ADC measurements are only used for user observation. The changing rate of the duty cycle is limited using a Rate Limiter block and the maximum value is limited by a Saturation block.

Closed loop

By selecting the “Closed_loop” configuration the output voltage of the buck converter is regulated using outer-voltage and inner-current feedback loops. The controller implementation is shown in Fig. 4. The voltage compensator regulates the DC bus voltage by comparing the sensed DC output signal against a reference. The output of the voltage loop generates the current reference and is compared against the sensed inductor current. This generated error signal is then used as the input to the current compensator, whose output, the modulation index, m, is then fed to the PWM modulator. Both the

Table 1: Key signal connections between LaunchPad, BOOSTXL-BUCKCONV and RT Box

BOOSTXL-BUCKCONV	Description	LAUNCHXL-Pin	RT Box Channel	F28379D	F28377S	F280049C
H3[1] PWM-HI	High-side drive signal for buck	J8[80]	DI16	EPWM4A	EPWM2A	EPWM1A
H3[2] PWM-LOW	Low-side drive signal for buck	J8[79]	DI17	EPWM4B	EPWM2B	EPWM1B
H3[5] DO-Load	Enable for active load	J8[76]	DI20	GPIO10	GPIO18	GPIO2
H2[7] Vout	Output voltage feedback	J7[67]	AO12	ADCINC4	ADCINA3	ADCINC3
H2[9] iL	Inductor current feedback	J7[69]	AO14	ADCINA4	ADCINA4	ADCINA3
H2[4] iL_avg	Filtered inductor current feedback	J7[64]	AO9	ADCINC5	ADCINA2	ADCINB6
H2[8] Vin	Input voltage feedback	J7[68]	AO13	ADCINB4	ADCINB3	ADCINC5

voltage and current compensators use the Continuous PID block from the PLECS component library. The switching frequency is 200 kHz.

The closed-loop controller has been designed for continuous conduction mode (CCM), with load PR2 enabled. The controller parameters are calculated according to [5]. A detailed explanation of the controller parameter calculations is provided with the PLECS demo model: Three-Level Boost PFC Converter.

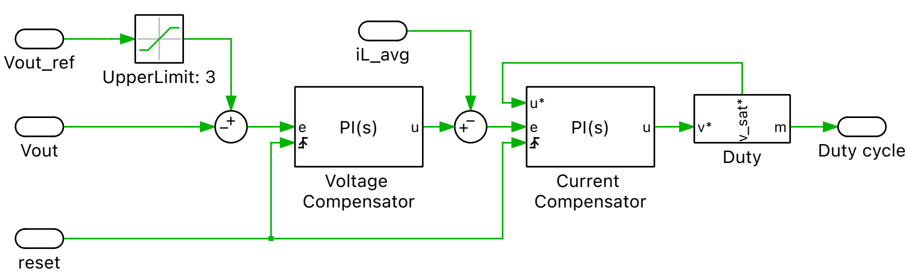

Figure 4: Closed-loop controller implementation

Configuring TI C2000 Target library components

- **Control Task Trigger** 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 as seen in Fig. 3. From the **Events** tab of the PWM block parameter window, the **ADC trigger** parameter is configured as `Overflow`, and the **ADC trigger divider** is set to 3 for the TI2837x, and 6 for the TI28004x MCU. This means that the ADC start-of-conversion is triggered after set number of `Overflow` events. Fig. 5 shows the corresponding PWM carrier, ADC and task trigger, and PWM outputs for an **ADC trigger divider** set to 3.

  By default, the C2000 target support package will only update the ePWM duty cycle register on PWM underflow and overflow events to prevent data corruption. In Fig. 5, note the delay between the task trigger and the instant when the duty cycle, m, is updated in the ePWM module. The control task is triggered by the ADC end-of-conversion, then the modulation index will update on the next underflow event after all ADC channels are converted and the control task is completed.

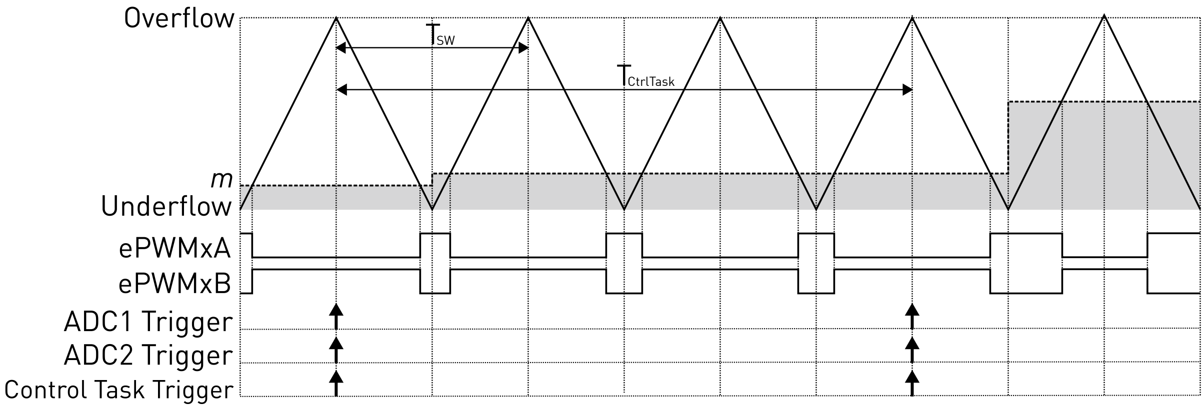

Figure 5: PWM carrier, ADC and task interrupts

- **Powerstage Protection** The Powerstage Protection block implements an interlock, which is a safety mechanism used to enable or disable all the PWM outputs on the target device. The PWM outputs are disabled unless there is a logical low-to-high transition on the input signal, labeled "en". This prevents the PWM signals from becoming active as soon as the code is executed on the target, thereby ensuring safe operation. For more details, please browse the Help section of this block. The LED "D9" or the "LED4" (corresponding to GPIO "DO_DSP_LED2" listed in the **Model initialization commands** window of **Simulation Parameters... + Initializations** tab from the **Simulation** menu) indicates the status of the Powerstage (when the LED is on it means the PWM outputs are active).

  In addition, a protection function is implemented within the Powerstage Protection block. An analog trip value of 6 A is defined using the **Coder + Coder options + Target tab + Protections** parameter. When the inductor current exceeds this value the trip signal A is triggered. A reaction to this trip signal A is defined in the **Protection** tab of the Powerstage Protection block.

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 I/O configuration of all the peripheral blocks (ADC, PWM etc.,) are configured for a TI 28379D [2] LaunchPad by default, but the demo model also supports code generation for TI 28377S [3] and 280049C [4] LaunchPads. The I/O configuration can be adapted from the **Model initialization commands** window (**Ctrl + E**); the value of `type_evm` must be changed to choose the desired target. You must also configure the corresponding **Target** in the **Coder Options** window accordingly.

The real-time simulation can be performed with the BOOSTXL-BUCKCONV hardware.

Configuring the TI C2000 target hardware

Before connecting the LaunchPad to the BOOSTXL-BUCKCONV hardware, configure the LaunchPad as follows:

- Install the power jumpers (JP1, JP2 and JP3).
- Install jumper on JP4 to make power available on connector side 2 and uninstall jumper on JP5 to provide 3.3 V.
- Uninstall jumper JP6 (only TI 28379D) to use the 5 V supply from the USB port.

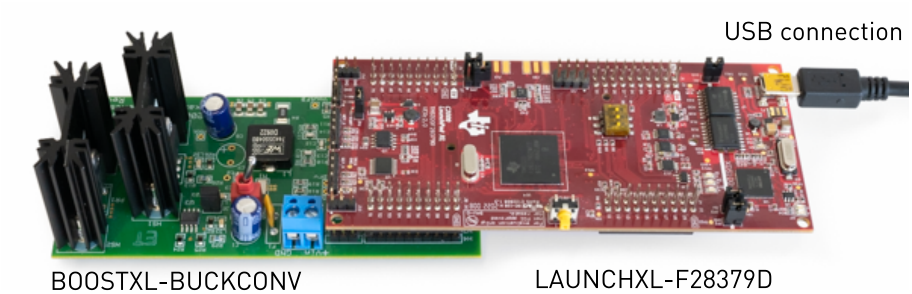

Figure 6: Hardware setup of the BOOSTXL and TI C2000 Launchpad

Configuring the BOOSTXL-BUCKCONV hardware

- Install a jumper between J1[1] and J1[2] to select signal H3[2] as the source for PWM-LO.
- Install a jumper on J2 to provide power to the MOSFET driver of the active load.
- Turn the toggle switch **SW1** *OFF* (pointing away from the heatsinks) to isolate JP1 from the plant.
- Insert the lead wires of the de-energized 9 V DC power supply into terminal block JP1.

Note Wrongly configured jumpers can damage the BOOSTXL-BUCKCONV board. Strictly follow the procedure given in this documentation to configure the LaunchPad and the BOOSTXL-BUCKCONV hardware.

Once the LaunchPad and the BOOSTXL-BUCKCONV are configured as stated above, the LaunchPad can be placed on the BOOSTXL-BUCKCONV hardware, as shown in Fig. 6.

Program the MCU

Program the “Controller” subsystem to the LaunchPad following the instructions below:

- 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. In this setting **TI28379S** also corresponds with the TI 28379D processor. Then under the **General** sub-tab, select the desired **Build type**.
- To `Build and program` the MCU directly from PLECS, choose `Run from RAM` as the **Build configuration** to program the MCU to flash to RAM, then select `LaunchPad` as the **Board** type, and click **Build**.

If programmed correctly, LED “D9” or "LED4" (corresponding to GPIO “DO_DSP_LED1” listed in the model initialization commands) should blink.

External Mode

The PWM signals are disabled by default for protection. To enable them, the MCU must first be conneced to the external mode using the instructions below.

- First, 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.

Enable the Powerstage

Now, from the Scope window of the “Controller” subsystem, observe that the real-time value of the input voltage is approximately 9 V DC. All the other measurements should be close to zero. If this is the case, follow the instructions below to enable the powerstage.

- Toggle the switch **SW1** on the BOOSTXL-BUCKCONV *ON* (pointing towards the heatsinks) to power the powerstage.
- To enable the PWM signals, toggle the manual switch **SW2** from the Controller subsystem from enable to disable, and back to enable. The LED corresponding to the GPIO “DO_DSP_LED2” listed in the model initialization commands should turn *ON*.

In the default configuration of the model the output voltage is controlled to 2 V and the load “PR2” is enabled. The real-time voltage and current measurements should be as shown in Fig. 7.

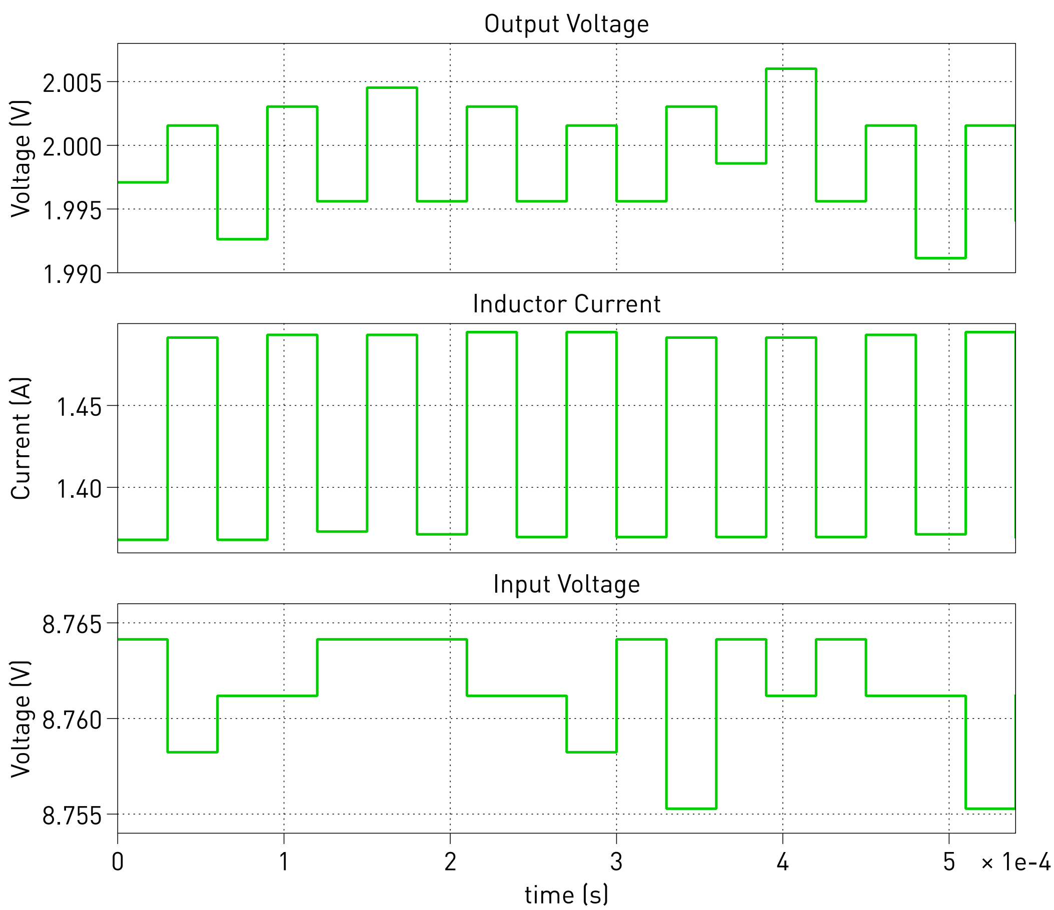

Figure 7: Real-time voltage and current values as sensed on the TI 28379D MCU

Parameter Inlining

Since the “Vout_ref” parameter has been added to the exceptions list in the the **Parameter Inlining** tab of the **Coder + Coder options...** window, this value can be adjusted on the fly. Changes in the parameters will be reflected in the Scope traces once they take effect. Note that the maximum value of “Vout_ref” is set to 3 for protection.

By default the model runs in closed-loop operation with load PR2 enabled. To run the model in open-loop operation or to disable load PR2, first disable the powerstage. Then, modify the “Controller” subsystem as desired and set “Vout_ref” to 0. Then re-program the MCU, enable the powerstage and slowly increment the value of “Vout_ref”.

4 Conclusion

This model demonstrates how to control a real power stage with a TI C2000 MCU with code automatically generated from PLECS. The combined plant and control model can run in an offline simulation or in real-time. The control code generated for the TI C2000 MCU from PLECS can control the real hardware.

References

[1] TI Digital Buck Converter BoosterPack BOOSTXL-BUCKCONV,
URL: https://www.ti.com/tool/TIDM-DC-DC-BUCK?keyMatch=TIDM-DC-DC-BUCK

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

[3] TI C2000 Delfino MCUs F28377S LaunchPad Development Kit
URL: https://www.ti.com/lit/pdf/sprui25

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

[5] Conception de systèmes automatiques, Hansruedi Bühler, Presses Polytechniques Romandes, Lausanne 1988, ISBN 2-88074-149-1

5 Appendix: Real-time Simulation with the RT Box

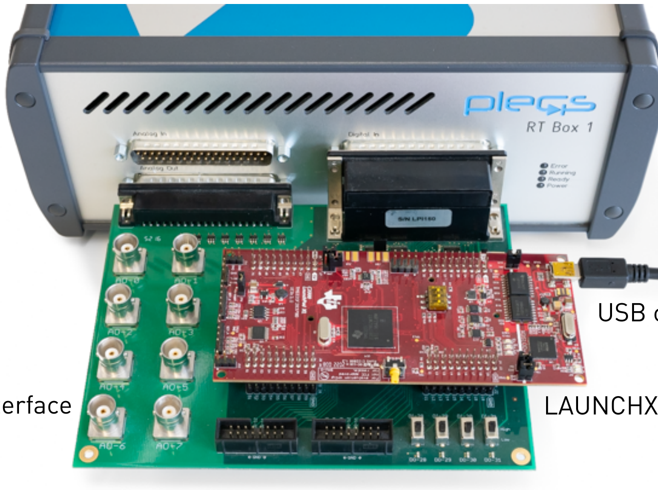

Figure 8: Hardware setup of the HIL verification

The BOOSTXL-BUCKCONV hardware can optionally be emulated on the PLECS RT Box to perform a hardware-in-the-loop (HIL) test. The "Plant" subsystem is configured to be built on the RT Box as shown in Fig. 2. A typical HIL testing configuration with the RT Box is shown in Fig. 8, where the LaunchPad (LAUNCHXL) is connected to the RT Box via an RT Box LaunchPad Interface board.

Revision History:

C2000 TSP 1.5.3 First release

How to Contact Plexim:

☎ +41 44 533 51 00 Phone

+41 44 533 51 01 Fax

✉ Plexim GmbH Mail
Technoparkstrasse 1
8005 Zurich
Switzerland

@ info@plexim.com Email

http://www.plexim.com Web

Embedded Code Generation Demo Model