



# Embedded Code Generation *Tutorial*

## Introduction to PLECS STM32 Code Generation

Tutorial Version 1.0

[www.plexim.com](http://www.plexim.com)

- ▶ Request a PLECS and PLECS Coder trial license
- ▶ Get the latest STM32 and RT Box Target Support Packages
- ▶ Check the PLECS, RT Box and STM32 TSP documentation

# Introduction

In this tutorial you will learn how to use STM32 microcontrollers (MCUs) with the PLECS Coder and the STM32 Target Support Package.

**Before you begin** this tutorial, please have available or install the following:

- Either PLECS Standalone or PLECS Blockset, with a valid license. A free trial license can be requested from the Plexim website: <https://www.plexim.com/trial>
- One PLECS Coder license. For a free trial, select the add-on product PCCT: PLECS Coder Trial
- The STM32 Target Support Package (TSP): Follow the step-by-step instructional video on installing and getting started with the STM32 TSP [1], or refer the Quick Start guide of the STM32 Target Support User Manual [2]
- One STM32 NUCLEO-G474RE board [3], referred to as “G474RE”, or “STM32 MCU”, or “STM32 board” hereafter
- Jumper wires
- Ensure that you have the reference files that you can compare with your own models at each stage of the exercise

## Exercise 1: Blinking an LED

In the first exercise you will create a simple model to blink the green LED “LD2” on the G474RE, which is connected to port A, pin 5 (PA5). The LED will be switched on for 0.5 seconds and then off for 0.5 seconds.

### Task 1.1: Create a PLECS Model

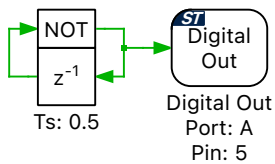


#### Your Task:

- 1 Open PLECS and create a new model file. From the PLECS Library Browser drag and drop a Subsystem block onto the main schematic and label it as “LED Blinker”.
- 2 Right-click on the “LED Blinker” subsystem and select **Subsystem + Execution settings...**. In the configuration window select the check box **Enable code generation** and **Apply** the changes. This converts the subsystem into an atomic subsystem (meaning all the components within the subsystem are executed together as a group), and allows for just the subsystem to be built on the target device, without having to build the whole PLECS model.
- 3 Delete any existing components within this “LED Blinker” subsystem (there are signal ports included by default), then drag and drop a Delay block onto its schematic, and set the Delay **Sample time** to 0.5. Next, drag and drop a Logical Operator block, and choose the NOT operator.
- 4 From the **STM32 Target** of the PLECS Library Browser, drag and drop a Digital Out block. Set the **Port** as A, and the **Pin number** as 5.
- 5 Then connect these three blocks as shown in Fig. 1.

### Task 1.2: Verify the Simulation offline in PLECS

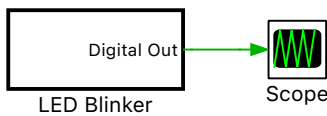
Before converting the “LED Blinker” subsystem directly into target specific code for the MCU, the PLECS model can be simulated offline on a computer.



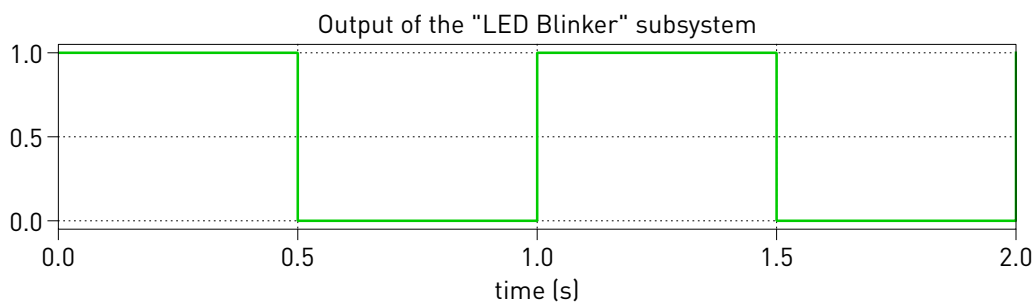
**Figure 1: Schematic to blink an LED on G474RE**



**Your Task:** Place a Scope at the top-level schematic and connect it to the Digital Out port, as shown in Fig. 2, and start a simulation (**Ctrl + T**). You should see a PWM waveform of 1 Hz with a 50 % duty cycle, as shown in Fig. 3.



**Figure 2: Top-level schematic**



**Figure 3: Results of the "LED Blinker" subsystem offline on PLECS**

### Task 1.3: Configure the Coder Options



#### Your Task:

- 1 From the PLECS schematic, browse to the **Coder + Coder options...** menu option. In the **Coder Options** window, select "LED Blinker" in the **System** list.
- 2 Then, from the **Scheduling** tab, set the **Discretization step size** to 1e-4. This parameter specifies the base sample time used to discretize the continuous state variables of the control blocks. Then choose the **Floating point format** as float.
- 3 From the **Target** tab, choose the **Target** as STM32G4x. Then under the **General** sub-tab, choose the **Chip** as G474RE.
- 4 Leave the other parameters as their default values. Save the model at a location of your choice.

### Task 1.4: Flash the STM32 MCU

Convert this PLECS model into target specific code for the STM32 G474RE MCU.



### Your Task:

- 1 Connect the desired MCU to the host computer through a USB cable.
- 2 To deploy the MCU target directly from PLECS: from the **Target** tab of the **Coder Options** window, choose the desired **Programming interface** from the dropdown menu and click **Build**. The default programming interface is OpenOCD.

If programmed correctly, the LED on the STM32 board should blink.

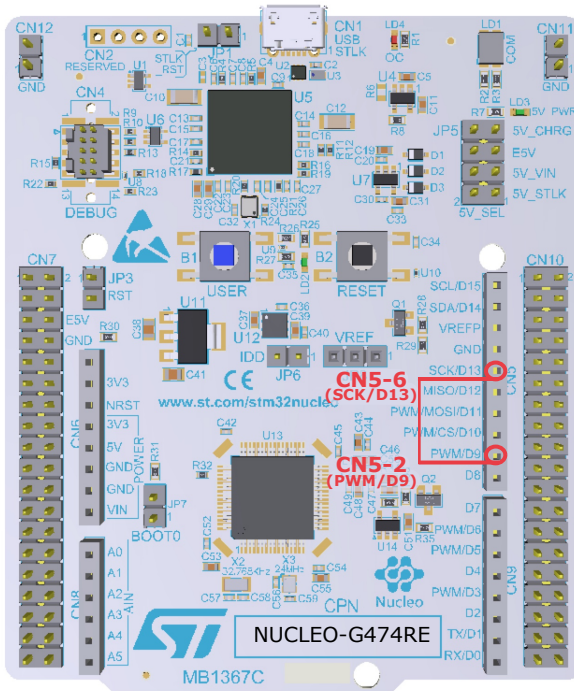


At this stage, your model and simulation results should be the same as the reference `stm32_intro_1.plecs`.

## Exercise 2: Configuring a PWM Output

In this exercise, let's control the same LED "LD2" from Exercise 1:, but using a PWM block. The PWM block generates a single or complementary PWM pair on one or more PWM resources. A modulation index must be provided as an input to this block.

LED "LD2" on the G474RE (PA5, labeled "SCK/D13" on CN5-6, or CN10-11) is not connected to an advanced-control timer (TIM) used to generate PWM signals on an STM32 MCU [4]. Therefore, we will first generate PWM signals onto an available PWM output (PC7, labeled "PWM/D9" on CN5-2, or CN10-19), and connect it to the LED using a jumper wire, as shown in Fig. 4.



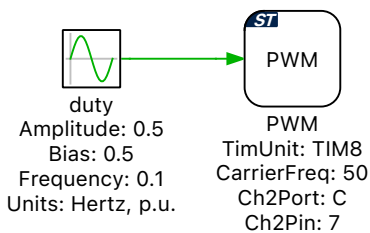
**Figure 4: Use a jumper wire to connect the PWM output (CN5-2) to LED "LD2" (CN5-6)**

## Task 2.1: Modify the PLECS Model



### Your Task:

- 1 In the same PLECS model file from Exercise 1, place a second Subsystem block and name it “LED Dimmer”. Again delete any existing components inside and enable it for code generation by right-clicking on the subsystem and in the configuration window of **Subsystem + Execution settings...**, selecting the check box **Enable code generation**.
- 2 From the **STM32 Target** component library, drag and drop a PWM block into the “LED Dimmer” subsystem.
- 3 From the I/O assignment table of [4], we can see that PC7 is connected to TIM8, channel 2. Therefore from the **Main** tab of the PWM block parameters window, choose the **TIM unit** as TIM8. Set the **Carrier frequency** as 50 Hz.
- 4 From the **Channel 2** tab, choose the **Mode** as Single output, and set the **Port** as C, and the **Pin** as 7.
- 5 Disable all other channels. Leave all the other parameters as their default values.
- 6 Then connect the PWM block to a Sine Wave Generator of **Amplitude** 0.5, **Bias** 0.5, **Frequency** 0.1 Hz, and **Units** Hertz, p.u. as shown in Fig. 5.



**Figure 5: Schematic to dim an LED on G474RE using a PWM block**

## Task 2.2: Verify the Simulation offline in PLECS

During an offline simulation, the PWM block behaves as a normal PWM generation block. Right-click on the PWM block and browse to **Subsystem + Look under mask** to explore the offline “Schematic” implementation.

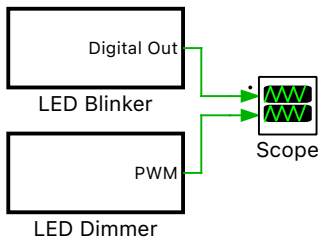


**Your Task:** Insert a new plot to the Scope of the top-level schematic by right-clicking on the plot window or from the Scope’s File menu, and connect the newly created Scope port to the PWM output port (see Fig. 6). Start a new simulation and you should see the PWM waveform, as shown in Fig. 7. This modulation waveform will gradually increase and decrease the brightness of the LED.

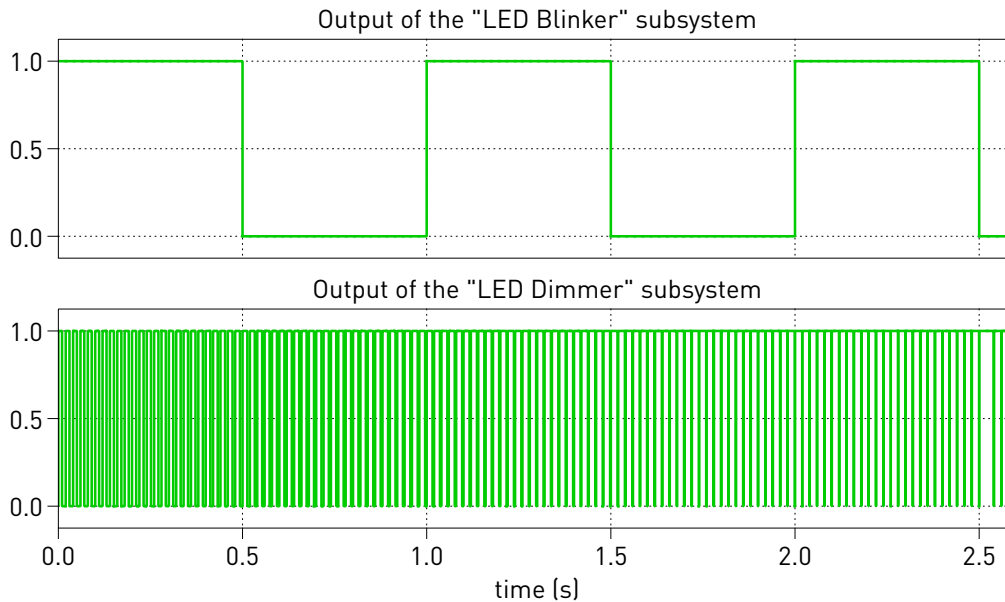
## Task 2.3: Make the required hardware connections



**Your Task:** Using a jumper wire, connect the pin PA5 (labeled “SCK/D13” on CN5-6, or CN10-11) corresponding to the LED “LD2” to pin PC7 (labeled “PWM/D9” on CN5-2, or or CN10-19) corresponding to the PWM output (see Fig. 4).



**Figure 6: Top-level schematic**



**Figure 7: Results of the "LED Dimmer" subsystem offline on PLECS**

## Task 2.4: Configure the Coder Options and Flash the STM32 MCU



**Your Task:** Back in the **Coder Options** window, choose "LED Dimmer" under the **System** list on the left. Then configure the code generation parameters and flash the STM32 MCU as instructed previously in Sec. Exercise 1:.

If programmed correctly, the brightness of the LED on the STM32 board should increase and decrease gradually.



At this stage, your model and simulation results should be the same as the reference `stm32_intro_2.plecs`.

*Optional:* Change the duty cycle input and/or the **Carrier Frequency** of the PWM block and explore its impact on the LED's brightness.

## Exercise 3: Connecting to the External Mode

In this exercise, we will learn how to connect to the External Mode. Once the generated code is running on the STM32 target, External Mode can be used to update Scopes in the PLECS application with real-time waveforms.

### Task 3.1: Modify the PLECS Model

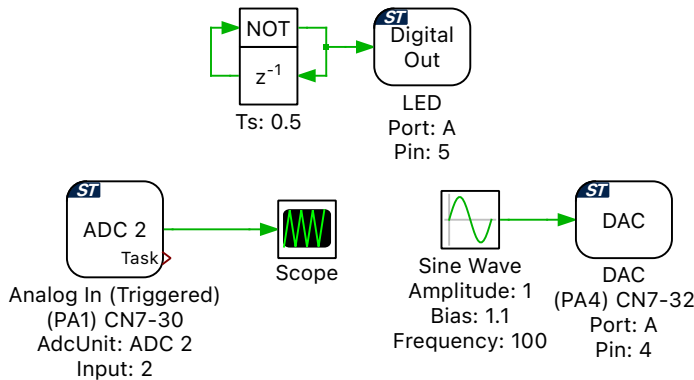
You will now create a simple loopback model, as shown in Fig. 8, with DAC and Analog In (Triggered) blocks from the **STM32 Target** library.

By externally connecting a DAC block to an analog input via a jumper wire, the generated analog signals can then be sensed via an Analog In (Triggered) block and used in the model environment. Optionally, scaling and offset factors can be configured for each channel via the parameter windows of the ADC and DAC blocks.



#### Your Task:

- 1 Continuing with the same PLECS model file from Exercise 2, create a copy of the “LED Blinker” subsystem and rename it as “Analog Loopback”.
- 2 Leave the circuit to blink the LED as is, and within this “Analog Loopback” subsystem, drag and drop DAC and Analog In (Triggered) blocks from the **STM32 Target** library.
- 3 DAC block parameters:
  - Select the **DAC selection** as Port and Pin, and set the **Port** as A and the **Pin number** as 4.
  - Connect the DAC block to a Sine Wave Generator of **Amplitude** 1, **Bias** 1.1, **Frequency** 100 Hz, and **Units** Hertz, p.u. as shown in Fig. 8.
- 4 Analog In (Triggered) block parameters:
  - Select the **ADC unit** as ADC2, and set the **Analog input channel** as 2.
  - Set the **Trigger source** to Determine automatically.
- 5 Leave all the other parameters not defined above as their default values.
- 6 Connect the output of the Analog In (Triggered) block to a Scope, as in Fig. 8.



**Figure 8: Schematic of the Analog Loopback subsystem**

### Task 3.2: Verify the Simulation offline in PLECS



**Your Task:** At the top-level schematic, connect the DAC output port to the Analog In (Triggered) input port, as in Fig. 9. Then start a new simulation and you should see the appropriate Sine waveform within the Scope of the Analog Loopback subsystem.



**7**



### Task 3.4: Configure the Coder Options and Flash the STM32 MCU



#### Your Task:


- 1 In the **Coder Options** window, choose “Analog Loopback” under the **System** list on the left. Then configure the code generation parameters as instructed previously in Sec. Exercise 1:.
- 2 Next, from the **Target** tab, browse to the **External Mode** sub-tab. Choose the **External mode** communication option as JTAG. The **Target buffer size** specifies how much target memory should be allocated to buffering signals for the External Mode. Leave it as 1000.
- 3 Click on **Build** to build and program the MCU.

### Task 3.5: Connect to the External Mode

Once the generated code is running on the STM32 target, the user can enter the External Mode to update Scopes in the PLECS application with real-time waveforms.



#### Your Task:

- 1 First, from the **System** list on the left-hand side of the **Coder Options** window, select the desired MCU.
- 2 Next, from the **External Mode** tab, click the  icon to edit the embedded target. Select the **Device type** of Serial over GDB and enter the **Device name** of localhost.
- 3 Next, click **Connect** and then **Activate autotriggering** to observe the test results in the subsystem Scope.



At this stage, your model should be the same as the reference `stm32_intro_3.plecs`. The captured real-time PWM and ADC waveforms should be the same as those shown in Fig. 11.

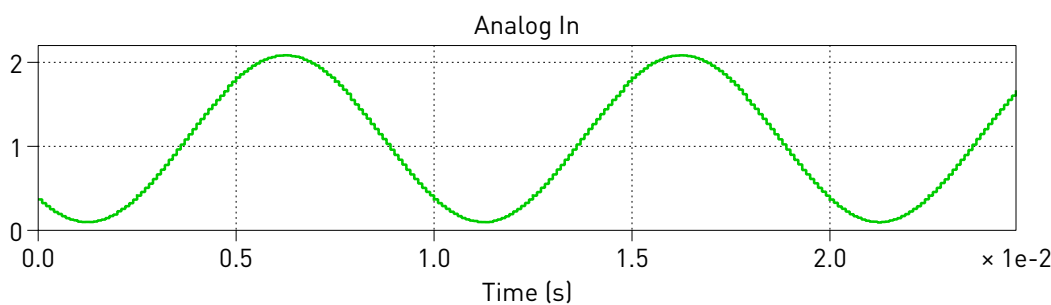


Figure 11: Captured real-time ADC signals on Nucleo-G474RE

## Exercise 4: Parameter Inlining

Certain parameter values on the target device can be changed in real time if the component is added to the **Exceptions** list found in the **Parameter Inlining** tab of the **Coder Options** window prior to building the model.

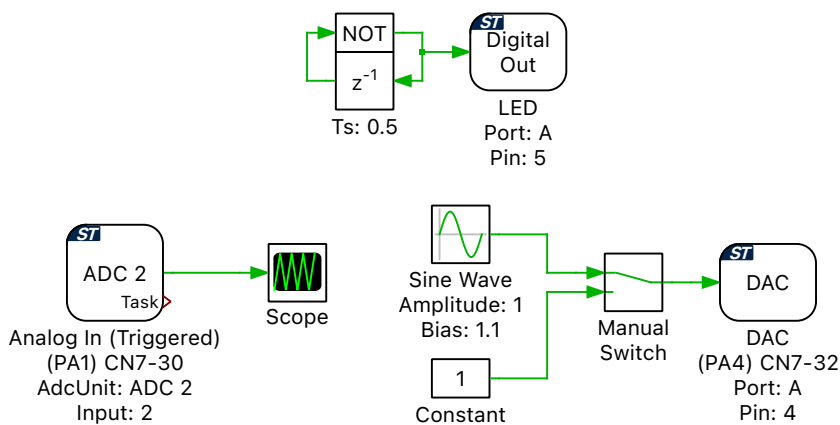
In this exercise, you will make a few parameters tunable. This means that these parameters can be changed when connected to the target device via the External Mode. Changes in the parameters will be reflected in the Scope traces once they take effect.

#### Task 4.1: Modify the PLECS Model



##### Your Task:

- 1 Disconnect the “Analog Loopback” subsystem from Exercise 3 from External Mode, if connected.
- 2 Continuing with this same PLECS model file from Exercise 3, edit the “Analog Loopback” subsystem by adding a Constant and Manual Switch blocks, as shown in Fig. 12.



**Figure 12: Schematic of the edited Analog Loopback subsystem**

#### Task 4.2: Configure Parameter Inlining, Flash the STM32 MCU & Connect to the External Mode



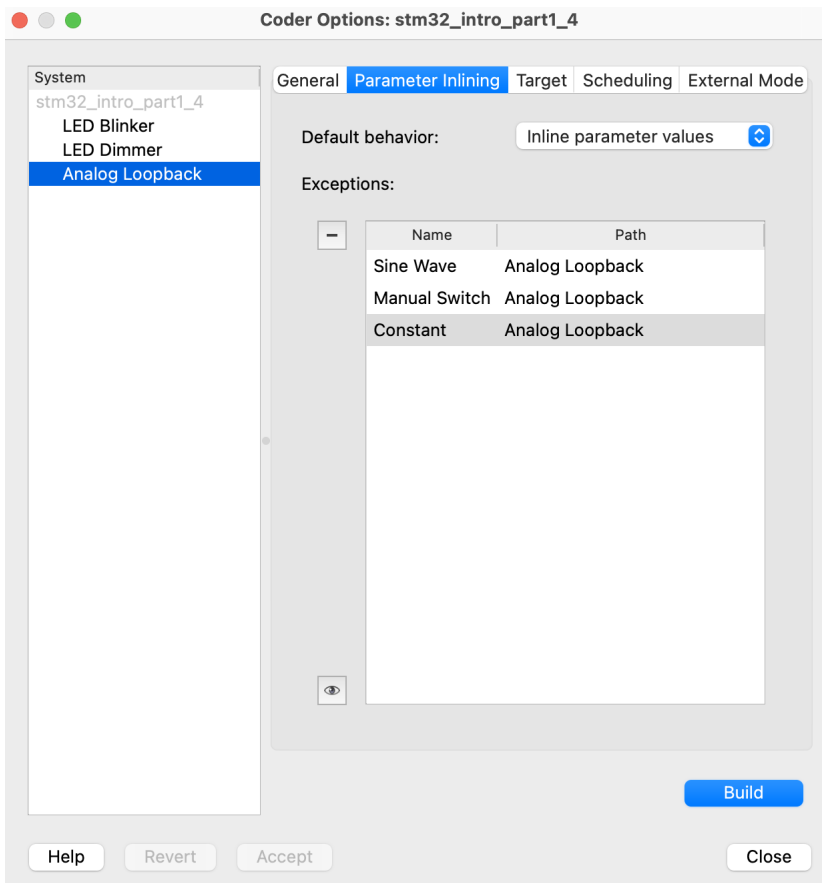
##### Your Task:

- 1 Navigate to the **Parameter Inlining** tab of the **Coder Options** window, and drag and drop the “Sine Wave”, “Constant” and “Manual Switch” blocks into the **Exceptions** list window, as shown in Fig. 13.
- 2 Choose the **Default behavior** as Inline parameter values.
- 3 **Build** the “Analog Loopback” subsystem onto the STM32 MCU, and **Connect** to the External Mode as instructed in Sec. Exercise 3:.
- 4 Now you should be able to easily switch between generating sine waves out of the DAC block to a constant value, and also vary the **Value** of the Constant, and **Amplitude** and **Bias** of the Sine Wave, and observe these changes in real time on the Scope.



At this stage, your model should be the same as the reference `stm32_intro_4.plecs`.

*Optional:* The STM32 MCU can only transmit or receive values between 0 to 3.3 V. Any values outside of this range will be cropped. Using the scaling and offset parameters of the DAC and Analog In blocks, try to transmit and receive values that are less than 0 V, or greater than 3.3 V.



**Figure 13: Parameter Inlining configuration**

## References

- [1] Getting Started with the STM32 Target Support Package:  
<https://www.plexim.com/support/videos/coder-stm32-getting-started>
- [2] STM32 Target Support User Manual: <https://plexim.com/sites/default/files/stm32manual.pdf>
- [3] STMicroelectronics, NUCLEO-G474RE: <https://www.st.com/en/evaluation-tools/nucleo-g474re.html>
- [4] STM32G4 Nucleo-64 boards (MB1367) User manual:  
[https://www.st.com/resource/en/user\\_manual/um2505-stm32g4-nucleo64-boards-mb1367-stmicroelectronics.pdf](https://www.st.com/resource/en/user_manual/um2505-stm32g4-nucleo64-boards-mb1367-stmicroelectronics.pdf)

## Revision History:

Tutorial Version 1.0      First release

## 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
	<a href="http://www.plexim.com">http://www.plexim.com</a>	Web

### *Embedded Code Generation Tutorial*

© 2002–2023 by Plexim GmbH

The software PLECS described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from Plexim GmbH.

PLECS is a registered trademark of Plexim GmbH. MATLAB, Simulink and Simulink Coder are registered trademarks of The MathWorks, Inc. Other product or brand names are trademarks or registered trademarks of their respective holders.