RT Box

Tutorial

Introduction to the RT Box using PLECS

Step-by-step introduction to the RT Box architecture, real-time workflow, inputs and outputs, and model structure

Tutorial Version 1.1

www.plexim.com

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

1 Introduction

In this tutorial you will learn step-by-step how to run basic PLECS models on the PLECS RT Box. The tutorial is designed for users making the transition from a PLECS model to a real-time simulation on the RT Box using PLECS Standalone. The specific learning goals for the tutorial are to:

- Gain deeper understanding of the RT Box hardware, workflow, and the RT Box Target Support library components.
- Learn how to update PLECS Scopes with data from a real-time simulation, change parameters during a real-time simulation, and gather information about your simulation from the RT Box's Web Interface.
- Learn how to structure your model to facilitate the transition from a PLECS model to a real-time simulation.

Before you begin This tutorial requires an RT Box and a PLECS Coder license. Refer to the Quick Start section of the RT Box User Manual [1] if you are configuring the RT Box for the first time.

The tutorial is designed so that it can be completed with an RT Box, a loopback cable to drive the RT Box analog and digital inputs from the RT Box outputs, and an RT Box interface card.

2 RT Box Architecture

The PLECS RT Box is a state of the art real-time simulator based on the Xilinx Zynq system-on-chip (SoC) that contains a dual-core ARM CPU and an FPGA. The real-time simulation is performed on a CPU core while the FPGA acts as an interface to the RT Box inputs and outputs (I/O). The CPU core has the flexibility to run C code representing a physical model or control system. The same code used for a PLECS simulation on your host PC can also be used to simulate the model on the RT Box. This approach ensures a consistent modeling framework for offline and real-time simulations.

The FPGA serves as a high-resolution, low-latency interface between the simulation running on the CPU and the RT Box analog and digital I/O. The RT Box is equipped with 16 analog input and 16 analog output channels. All analog I/Os feature 16-bit resolution and a sampling frequency up to 2 MHz. In addition, the RT Box has 32 digital input and 32 digital output channels. Each digital I/O can be configured to generate PWM signals or capture PWM signals with a time resolution of 7.5 ns. Digital I/Os can also be used as general purpose signals.

Refer to the RT Box User Manual for a more detailed overview on the RT Box architecture and principles of operation [1].

3 Real-Time Workflow

The real-time workflow is designed to easily transition from a PLECS model to a real-time simulation on the RT Box without having to develop and maintain separate models. A typical real-time workflow in PLECS consists of the following key steps, summarized in Figure 1:

1. **Create a PLECS model**: The workflow begins with a PLECS model. The PLECS model should perform as expected prior to configuring the model for real-time simulation. It is generally beneficial to develop models representing the real-time system deployed on the RT Box as well as external hardware connected to the RT Box. This allows offline and real-time models to be benchmarked later in the workflow. The model deployed to the RT Box can include components from any PLECS domain, with the exception of unsupported components listed in the Code Generation chapter of the PLECS Manual [2].
2. **Add and configure RT Box library components:** The next step is to add RT Box specific components from the library browser to map simulation signals to I/Os on the front panel of the RT Box. The signal paths from the RT Box to any physical hardware must be defined and analog I/O

scale and offset values should be calculated. At this stage it is also helpful to structure the model so the portion of model deployed to the RT Box is contained within one sub-system.

3 **Run an offline simulation:** All RT Box I/O components have representative offline models. Simulate the entire model offline to ensure that all scaling and signal connections in the model are correct. Physical connections should also be checked at this stage.

4 **Select a discretization step size:** To simulate a PLECS model in real-time, the model must be discretized to run at a fixed execution rate using a fixed-step solver. The PLECS Coder transforms the continuous state-space equations of the model into a discrete state-space representation compatible with a fixed-step solver. The PLECS Coder then translates the model into C code designed to execute on a generic or real-time target. A discretization step size must be specified before generating code. Due to the fast time constants inherent in power conversion systems, the discretization step size typically is on the order of microseconds. The ideal step size is a compromise between system model fidelity and accuracy of the simulation results.

5 **Run a CodeGen simulation:** The impacts of model discretization can be evaluated prior to a real-time simulation by using the CodeGen mode. The CodeGen mode utilizes the PLECS Coder to create generic C code that can execute on your host PC in PLECS. The generic C code represents the discretized model that can be benchmarked against a continuous state-space model. Comparing the discretized and continuous simulation results will show how discretization impacts the model fidelity and indicate if the chosen discretization step size is appropriate. At this stage the discretization step size should be tuned to achieve the desired model fidelity.

6 **Build and upload the model the RT Box:** The PLECS Coder then translates the model into C code specifically targeted to run on the RT Box in real-time. The model code is automatically generated, compiled, and uploaded onto the RT Box with a single-click. The simulation begins automatically if there are no errors. If there are errors, the RT Box Web Interface provides essential debugging information in the application log. At this stage it may be necessary to further optimize the model and refine the discretization step size based on the real-time performance and RT Box processor load.

7 **Connect to the RT Box using the External Mode to perform tests:** Once the real-time code is running on the RT Box the user can enter the External Mode to update Scopes in the PLECS application with real-time waveforms, change certain simulation parameters, and capture real-time data for post-processing.

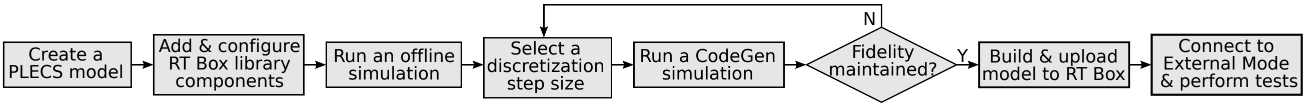
Create a PLECS model
Add & configure RT Box library components
Run an offline simulation
Select a discretization step size
Run a CodeGen simulation
Fidelity maintained?
N
Y
Build & upload model to RT Box
Connect to External Mode & perform tests

Figure 1: Typical RT Box workflow

4 RT Box Web Interface

The RT Box Web Interface displays essential diagnostic information. You can access the Web Interface from the **Coder Options** window. After selecting an RT Box target device, click on the blue monitor icon to open the Web Interface. Here you will see the model name and the sample step size of the model running on the RT Box. The current cycle time shows the how long it takes the RT Box to execute a model step. Always make sure that the cycle time is less than the sample time.

You can upload models and start or stop the RT Box from the **Application** tab. Pre-compiled RT Box models can be uploaded to the RT Box by navigating to the active model directory, opening the folder containing the generated code, and then uploading the file with an `.elf` extension. Note the RT Box Web Interface can also be accessed through your web browser and without an active PLECS license. This means you can load and execute pre-compiled `.elf` files to the RT Box without opening the PLECS software.

The **Front panel** tab window provides pinout information for the analog and digital connectors on the front panel and the current state of the status LEDs. Click the image of any front panel connector to see a table showing the function of each connector pin.

The **Diagnostics** tab captures the application log including errors, if any. Should the red **Error** LED illuminate the application log will contain diagnostic information about the model run-time error. The **Info** tab shows the host name, IP and MAC addresses, the serial number and firmware version.

5 Analog Inputs and Outputs

In this section you will create a simple model using the Analog In and Analog Out components from the RT Box Target Support library. The Analog In and the Analog Out blocks are used to map analog signals from the RT Box front panel to values in a real-time simulation. You will manipulate the scaling and offset parameters to reshape the analog waveforms. You also will observe the impacts of the analog output voltage range setting and model discretization on analog input and output signals.

Analog output channels are often used in hardware-in-the-loop (HIL) simulations. One use of an Analog Out block is to represent a voltage or current measurement of a power converter simulated on the RT Box. Each analog output has offset and scaling parameters. The offset and scaling parameters can behaviorally model the gain and DC offset of a sensing circuit.

Analog inputs are essential to sense voltages and currents when using the RT Box to control a power converter in Rapid Control Prototyping (RCP) applications. Analog inputs also have scaling and offset parameters and are equipped for both single-ended and fully differential inputs. More detailed specifications on the RT Box analog input characteristics are available from the RT Box User Manual [1].

This exercise also highlights the External Mode. The External Mode establishes a communication link between the RT Box and the PLECS user interface. While connected in External Mode the PLECS simulation scopes and displays can be updated with real-time measurements. Certain parameters can also be changed in real-time while in the External Mode, as explained in Section 6.

Your Task:

1 Use your loopback cable to connect the RT Box analog outputs to the analog inputs. Now the analog input channel 0 is physically connected to the analog output channel 0.

2 Create a new model and place an Analog Out block and an Analog In block from the RT Box target library onto your schematic. Connect a Sine Wave Generator with an amplitude of 1 and a frequency of 50 Hz to the Analog Out component. Set the analog output and analog input to channel 0. Leave all other Sine Wave Generator and Analog Out component values as their defaults. Include a Scope component from the PLECS library to view the signals. Use a Multiplexer component so you can see the generated sine wave and the analog input measurement in the same Scope.

Your schematic should look like Figure 2. Run the simulation and save the model.

3 From the **Coder** drop-down menu select the **Coder options...** dialog. In the **Coder Options** window select the **General** tab and enter a discretization step size of $1e{-}4$ seconds.

In the **Target** tab select PLECS RT Box as the target and click the binocular icon to locate an available RT Box. Note that the analog output voltage range for all channels can be selected from this tab. Leave all other options as the defaults and click the **Build** button at the bottom of the window to start the build process.

When the model has successfully been uploaded to the RT Box target and the simulation is running the blue **Running** LED on the RT Box front panel will be illuminated.

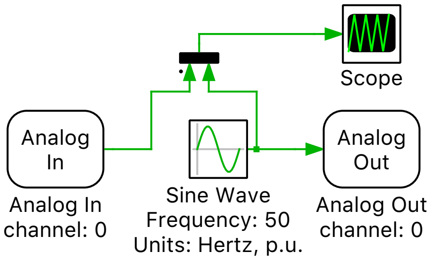

Figure 2: Generating and measuring analog waveforms with a loopback cable

4 Navigate to the **External Mode** tab of the **Coder Options** window. Click **Connect** and then the **Activate autotriggering** buttons. Open the Scope on your PLECS schematic and you will notice that it is periodically updating with real-time data. From the **Coder Options** window you may increase or decrease the Number of samples or the Decimation setting to capture multiple cycles of the sine wave.

The scope update can synchronize the data capture to a specific trigger event. From the **External Mode** tab of the **Coder Options** window select the Sine Wave Generator as the Trigger channel.

On the vertical axis of the scope window there is a small square indicating the trigger level and delay. If the level or delay are outside of the current axis limits, a small triangle will be shown instead. Adjust the trigger level from the scope window by vertically dragging the square icon with the left mouse button pressed. The trigger delay can be adjusted in from the scope by holding the **Shift** key and then horizontally dragging trigger icon with your mouse. Both the trigger level and trigger delay can be set from the **External Mode** window. You can also adjust the trigger update rate in this window.

? Do the PLECS Scope and oscilloscope waveforms agree? Review the settings specified in the **Target** tab of the **Coder Options** window and explain why the waveforms differ.

A Note the analog output voltage range will be restricted between 0 to 5 V from the settings specified in the **Target** tab of the **Coder Options** window. The waveform on the oscilloscope will show significant clipping during the negative half cycle of the sine wave. The PLECS Scope and an external oscilloscope will not strictly show the same waveform.

At this stage your model should be the same as the reference model *introduction_rtbox_analog_1.plecs*.

5 Change the scale and offset parameters of the Analog Out component to transform the 50 Hz sine wave input to an analog output with a peak-to-peak voltage of 5.0 V and an offset of 2.5 V.

Next, determine Analog In scaling and offset parameters that will reshape a 0-5 V analog input voltage to a -1 to 1 V range.

The resulting analog input signal will then have the same magnitude and offset as the Sine Wave Generator output; the two signals can now be compared directly.

Build and upload the model to the RT Box with a $1\mathrm{e}{-3}\mathrm{s}$ discretization step size. Reduce the number of external mode samples to 300. Observe how Sine Wave Generator output correlates with the sensed analog input voltage.

? What is the delay between the generated sine wave and the measured signal in seconds? You may use the Scope cursors to determine the delay. How does the delay compare to the $1\mathrm{e}{-3}\mathrm{s}$ discretization step size?

A You will see a delay of 2e−3 s seconds or a two time step delay.

? Does the discretization step size limit the frequency content of analog output signals?

A Yes, the discretization step size not only limits the frequency content of the analog output signal, but all signals within the simulation. For a continuous periodic signal, the more discrete time steps per period results in a higher fidelity representation of the signal. The maximum frequency that can be represented is limited by the Nyquist rate. If the Nyquist rate is violated then aliasing will occur. For example, a 1100 Hz sine wave with a 1e−3 s discretization time step will result in a 100 Hz signal both in the simulation and at the analog output pin.

? Is the discretization of analog signals represented in the PLECS Scope by default?

A The discretized nature of a signal is not always represented in the PLECS Scope. One way to show the discrete nature of the real-time waveform is to change the signal type setting. To do this, in the PLECS Scope activate the cursors. When the cursors are activated, the data view window appears if it was not already open. A small icon that represents the signal type is shown next to the signal name in the data view window. Double-click on the signal type icon and change the representation to discrete.

🏁 At this stage your model should be the same as the reference model *introduction_rtbox_analog_2.plecs*.

6 Digital Inputs, Outputs, and PWM Signals

The RT Box Target Support library includes two different classes of digital input and output signals. The Digital In and Digital Out components represent general purpose I/Os that are updated once per model time step. These signals are often used to convey status information between the real-time simulation and an external system, for example to model an emergency shutdown signal or a sensor failure alarm. These components are not suitable to sense or generate PWM signals as their time resolution is limited to the discretization time of the model. A PWM signal with a 100 µs switching period and a 1 µs model step can represent only 100 different duty cycles.

PWM Capture and PWM Out library components decouple the PWM resolution from the simulation time step by utilizing the 7.5 ns time resolution of the FPGA. The PWM Out block is used in RCP applications to generate a configurable PWM signal on an RT Box digital output. The input to the PWM Out block is a modulation index that is updated every model step. However, the rising and falling edges of the PWM output may occur between model steps. For the same 100 µs switching period a PWM Out component can generate 13,333 different duty cycles. Figure 3 shows an example where a Digital Out signal is latched for a whole model step while the falling edge of a PWM Out signal occurs between model steps.

For real-time HIL applications where accurate PWM sampling is required the PWM Capture component is used. The PWM Capture component samples the digital input at the FPGA sample rate of 7.5 ns. All samples over a model time step are then averaged to determine the signal on time duration as a value between 0 and 1. The PWM Capture component must be used in conjunction with a “hybrid power module” from the Power Module section of the PLECS library browser. The time-averaged value of the PWM input of the previous model step is applied as the effective duty cycle of the hybrid power module. The accuracy of the duty cycle sensing is no longer limited by the model time step. Figure 4 shows an example of how a Digital In component and a PWM Capture block processes the same PWM

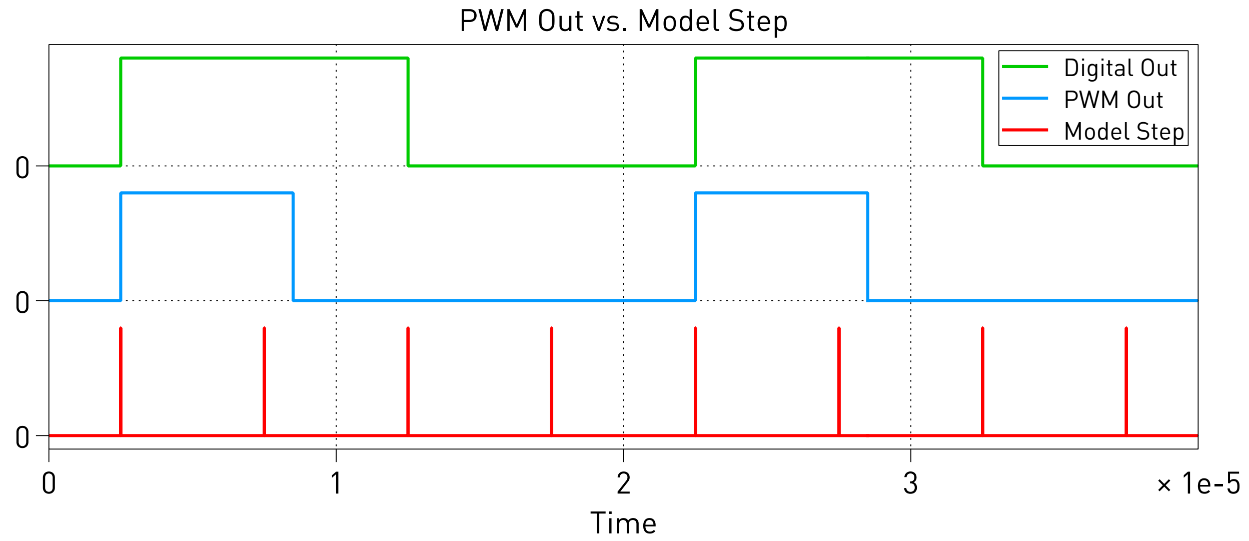

Figure 3: Digital Out and PWM Out signals compared with a model step

input. The PWM Capture result contains additional duty cycle information compared to the digital input.

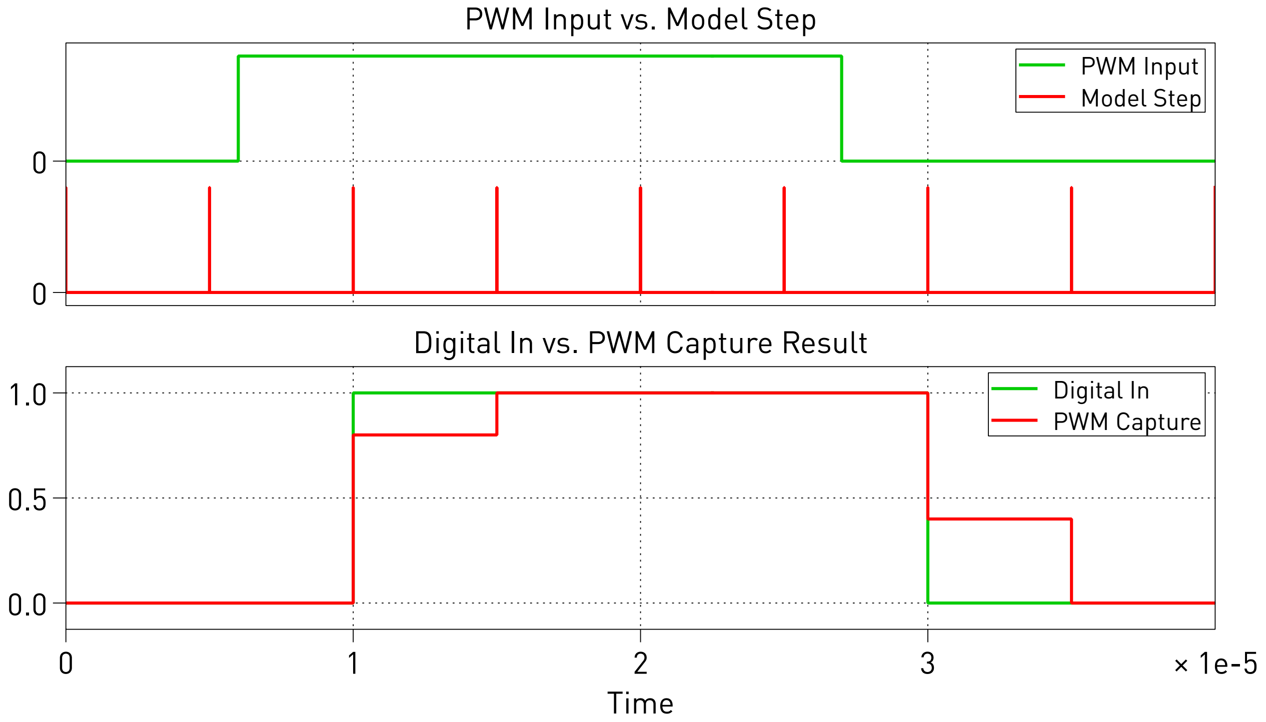

Figure 4: PWM Capture signal showing the output of a time-averaged PWM input signal

6.1 Generating a PWM Signal Using a Digital Out and a PWM output

In this exercise you will use the Digital Out and a PWM Out components to generate two PWM signals. You will observe that the PWM Out component can represent a wider range of duty cycles since the duty cycle is not coupled to the model step.

You will also learn how to use the Parameter Inlining feature to change parameters on the RT Box via PLECS. Parameter Inlining defines which parameters are tunable in the generated code. By default, all parameters are coded as numeric constants in the generated code. Exceptions to the default behavior can be specified so parameters can be changed at execution time without recompilation. Component parameters that affect the physical model equations such as resistances or inductances, cannot be kept tunable because changing such parameters in general requires a recalculation of the complete equation system. You can, however, keep the parameters of current, voltage, and other sources tunable.

Your Task:

1 Connect an Interface Card to the RT Box. On the LaunchPad Interface, ControlCard Interface, and Digital Breakout boards digital outputs 28 through 31 drive the LEDs in the lower right hand corner of the board.

2 Create a new model and place a Digital Out component from the library browser onto your schematic. Assign the Digital Output to channel 30. Add a Triangular Wave Generator to create a 5 kHz falling-edge sawtooth waveform. Change the Triangular Wave Generator minimum and maximum signal values to 0 and 1. Add a Relational Operator to your schematic and compare the output of the triangular wave carrier with a constant value of 0.3. The output of the Relational Operator should be true when the constant duty ratio is greater than or equal to the sawtooth carrier.

Add a PWM Out component to your model. Connect the Constant representing the PWM duty to the PWM Out block. Change the carrier limits to [0,1], the carrier type to sawtooth, and assign the PWM Out to digital output channel 31.

Refer to Figure 5 to see a representative schematic.

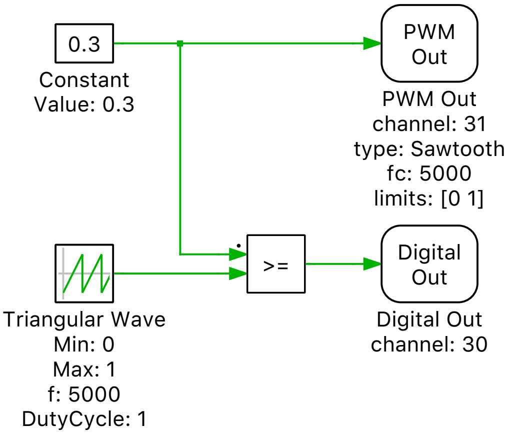

Figure 5: Comparing two different methods of generating a PWM signal

3 Open the **Coder Options** window and navigate to the **Parameter Inlining** tab. Drag and drop the Constant defining the 0.30 duty ratio from your schematic into the exceptions list.

4 Deploy your model to the RT Box with a discretization time of $1/20^{\text{th}}$ the switching period or $10\text{e}{-6}\,\text{s}$. Connect to the RT Box via the External Mode and activate autotriggering.

? What happens to the LEDs as you change the duty cycle?

A The brightness of the LEDs change.

5 Set the duty cycle to 0.01.

With a duty cycle of 0.01 do both LEDs have the same brightness? If they differ, explain why.

The LED associated with the Digital Out component will be brighter than the LED with the PWM Out component. The PWM resolution is directly associated with the minimum brightness of the LED. The PWM Out component has a higher PWM resolution than the Digital Out component. Furthermore, the PWM Out component resolution is not directly associated with the simulation time step.

Set the duty cycle such that the LEDs are no longer illuminated.

What duty cycle is required to turn both LEDs off? How does the chosen discretization time impact the minimum brightness of the LED connected to a Digital Out component?

A negative value is required to turn off the LED. The triangular ramp starts exactly at zero, resulting in the greater than or equal to comparison being true for a zero value. The logical high output is then latched for one simulation time step. The PWM signal resolution, directly associated with the minimum brightness of the LED, will change in relation to the discretization step size.

At this stage your model should be the same as the reference model *introduction_rtbox_digital_1.plecs*.

6.2 Digital Inputs and High Fidelity PWM Capture (optional)

This exercise compares the Digital In and PWM Capture component behavior with identical PWM input stimuli. In this exercise a loopback cable is required to connect the PWM outputs of the RT Box to the digital input pins. Alternatively, an external PWM source or signal generator could be used for this purpose.

The Digital In component records the input signal as either a 0 or 1 representing the digital input status just prior to the model step, while the PWM Capture produces a value between 0 and 1 representing the percentage of time the digital input signal was active over the previous model step, as shown in Figure 4. When the average on-time information is used with “hybrid power modules” the model accuracy is significantly improved compared to fully switched models.

Your Task:

1 Create a new PLECS model and build the circuit shown in Figure 6. The circuit consists of a Digital In component assigned to channel 0 and a PWM Capture component assigned to channel 1. The output of the Digital In and PWM Capture blocks are connected to a Signal Multiplexer component and then a Scope. The PWM Out block uses digital outputs 0 and 1, with all other PWM Out parameters remaining at the default values of a 10 kHz carrier frequency and carrier limits of [-1,1]. A Constant representing the PWM duty is connected to the PWM Out block via a Signal Selector so that both PWM outputs have the same duty and settings. Add the duty cycle Constant to the parameter inlining list. Set the discretization step size in the **Coder Options** window to $10\mathrm{e}{-6}\,\mathrm{s}$.

2 *Loopback Required*: Use a loopback cable to connect the RT Box digital outputs to the digital inputs. Upload your model to the RT Box and connect via the External Mode. Compare the Digital In and PWM Capture component output signals. Activate autotriggering and set

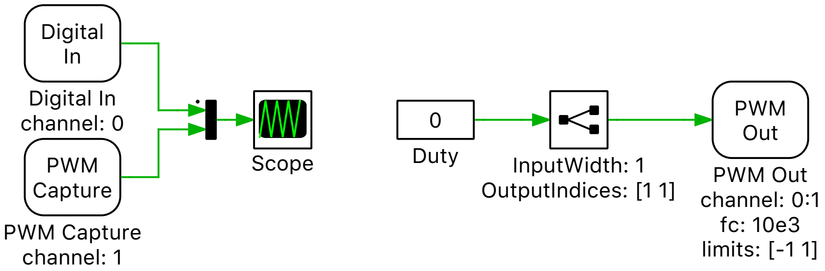

Figure 6: Comparing the Digital In and PWM Capture components

the trigger channel to Digital In (channel 0) at a trigger level of 0.5. Set the number of samples to 300. Then adjust the duty cycle from 0 to 0.1 and observe how the scope signals change.

? How do the Digital In and PWM Capture results compare? Is the width of the Digital In signal constant or variable at different operating points? How would this impact the real-time performance of a simple model, for example a buck converter?

A The Digital In signal is always either 0 or 1. The PWM Capture block is a value between 0 and 1. For a duty cycle of 0, the Digital In behavior appears to remain constant while the PWM Capture signal is changing. This is due to the asynchronous behavior of the PWM generation and PWM sampling.

When the duty cycle is changed to 0.1 you will observe the Digital In duty cycle rapidly fluctuating due to this asynchronous behavior. However, we know the duty cycle is constant based on the PWM signal the RT Box is generating. You will notice that the average on duration for the PWM Capture signal will increase. For an ideal buck converter the output voltage is proportional to the duty cycle. Using a Digital In component to sense PWM signals will result in a varying duty cycle and thusly a varying output voltage due to this sensing error.

🏁 At this stage your model should be the same as the reference model `introduction_rtbox_digital_2.plecs`.

7 Model Structure and an Introduction to Step Size Selection

Structuring a PLECS model into separate subsystems is a key part of the real-time workflow, whether the end use of the RT Box is for HIL or RCP. The portion of the model that will be deployed on an RT Box should be contained within one subsystem. External to that subsystem is a representation of the physical hardware connected to the RT Box. The interface between these two systems defines all required I/O including analog, digital, and PWM signals. This structure allows you to create offline models that represent the controller connected to the RT Box, for HIL, or an externally connected power stage, for RCP.

With this model structure you can directly benchmark and compare the offline and real-time simulation results. Another major benefit of this approach the ability to evaluate the impacts of model discretization using the CodeGen mode, where continuous and discrete model representations can be benchmarked for accuracy.

This exercise shows an example of a simple model structure and how the CodeGen mode can be used to evaluate the impact of discretization step size on the behavior of a simple first order system.

Your Task:

1 Create a new model and then drag and drop a Subsystem component onto your canvas from the library browser window. Double-click on the Subsystem to view its contents and remove the existing Signal Inport and Signal Outport components. Add an Analog In and an Analog Out block into the subsystem. Use analog input 0 and analog output 0 of the RT Box. Add a Transfer Function component implementing a first order system with the form $G(s) = \frac{1}{0.02s+1}$ as shown in Figure 7. Connect the transfer function input and output signals to a Scope.

Notice that the subsystem in the top-level schematic now has new ports where you may connect control signals. The new ports are created by Target Inport and Target Outport components that can be seen by looking under the library objects' masks. The names of the Analog In and Analog Out components correspond to the port name of the parent subsystem. These ports represent interface between the real-time simulation and the hardware connected to the RT Box. Right click the Analog In component and select **Subsystem + Look under mask** to observe the offline implementation of the Analog In model. You can see the scaling and offset parameters of the RT Box analog inputs are reflected in the offline implementation.

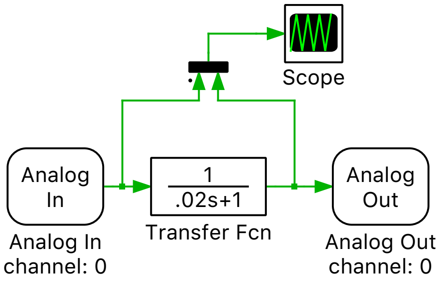

Figure 7: A simple first order system

2 Add an additional scope on the top-level schematic connected to the Analog Out port. Add a Pulse Generator with a 50 Hz frequency. Connect the Pulse Generator output to the Analog In port. You can leave all other parameters at their defaults. Your top-level schematic should look like Figure 8.

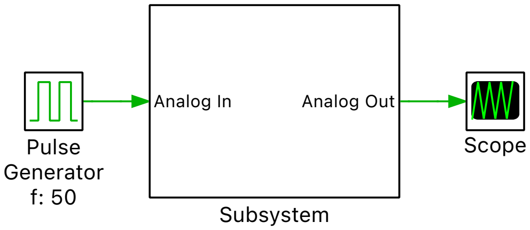

Figure 8: Subsystem showing ports created by the Analog In and Analog Out components

3 Run the simulation and save all Scope traces. Label the traces as “Offline”. Notice that the Scope on the top-level schematic reflects the scaling and offset values of the Analog Out block.

4 Next, enable the subsystem for code generation. Right-click on the subsystem and navigate to **Subsystem + Execution settings...** and click the **Enable code generation** checkbox. The border of the Subsystem is now bold. From the **Coder Options** window navigate to the Subsystem in the left-hand menu. In the **General** tab enter discretization time constant of $1\mathrm{e}{-3}\,\mathrm{s}$. In the **Target** tab select the Generic target from the drop-down menu. Click the **Build** button. In the directory where you saved your model, a new folder will be created that contains the C code representing the discretized subsystem.

5 Right-click the Subsystem and select the **Subsystem + Simulation mode + CodeGen** option. Once you select the CodeGen mode the subsystem border will change to a dashed bold line as shown in Figure 9. When in CodeGen mode, PLECS uses the discretized C code representing that subsystem. Scopes within the discretized subsystem will not update, but Scopes external to the subsystem will.

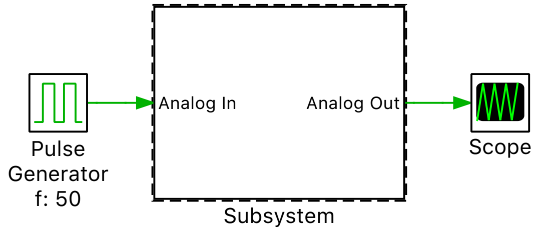

Figure 9: Subsystem in CodeGen mode with a dashed bold border

6 Run the simulation and save all new Scope traces as “CodeGen_1e-3”. Notice the Scope within the Subsystem did not update. Open the Scope on the top-level schematic and zoom into the waveforms until the impact of system discretization is apparent. Figure 10 shows the output from the simulation.

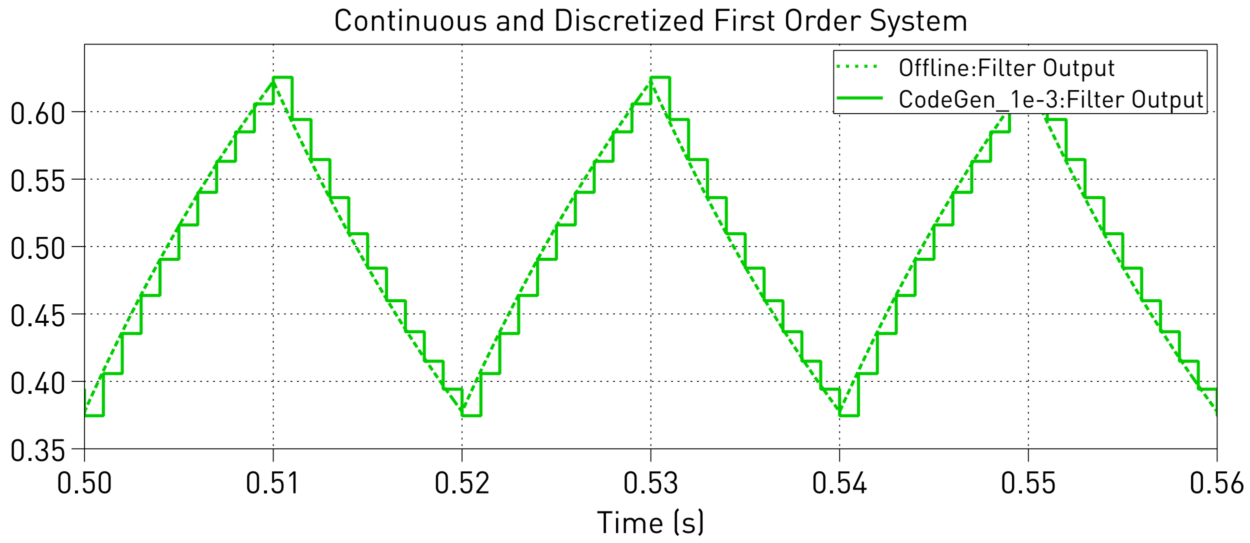

Figure 10: Comparison between “Offline"’ and “CodeGen” results

7 Compare the CodeGen results with the following discretization step sizes: $1\mathrm{e}{-3}\,\mathrm{s}$, $2\mathrm{e}{-3}\,\mathrm{s}$, $10\mathrm{e}{-3}\,\mathrm{s}$, and $25\mathrm{e}{-3}\,\mathrm{s}$.

? What discretization step size is appropriate for this system? At what discretization step size would we begin to anticipate significant deviation from the expected results?

A Discretization step sizes of $1\mathrm{e}{-3}\,\mathrm{s}$ and $2\mathrm{e}{-3}\,\mathrm{s}$ result in a good approximation of the continuous signal. A step size of $10\mathrm{e}{-3}\,\mathrm{s}$ shows the fundamental period of the input signal, but deviates significantly from the reference solution. For any step size above the Nyquist ratio, such as the $25\mathrm{e}{-3}\,\mathrm{s}$ result, one can expect unacceptable performance. In practical HIL and RCP simulations, the required accuracy of the reconstructed signal, as well as the acceptable latencies introduced by discretization and signal sampling, will vary from application to application.

8 Change back to a real-time model by changing the simulation mode back to normal and updating the target to the RT Box. Deploy the model onto the RT Box with a $1e{-}3\,\mathrm{s}$ discretization step size and connect via the External Mode. At this stage you will observe the Scope within the Subsystem updating, but not the Scope on the top-level schematic. Touch the analog input 0 pins with finger and observe the waveforms in the PLECS Scope update.

? Based on the results from Step 7, what can be said about transfer function output?

A The noise generated by touching the analog input pin is generally dominated by the AC mains frequency. The previous results showed that the $1e{-}3\,\mathrm{s}$ discretization step size was appropriate for this first order system. The resulting filtered output is a good representation of the continuous transfer function response.

While this exercise illustrated a simple application of the CodeGen mode, the same approach can be extended to more sophisticated models.

At this stage model should be the same as the reference model *introduction_rtbox_model_structure.plecs*.

8 Conclusion

This tutorial demonstrated step-by-step how to run basic PLECS models on the PLECS RT Box. Starting from an understanding of the RT Box architecture, you learned how several important RT Box library components operate and their key applications. Important topics of analog I/O range scaling and discretization were explored. Different digital I/O implementations were introduced, highlighting how the digital signal performance can be decoupled from the model step-size by utilizing the 7.5 ns time resolution of the FPGA. With the External Mode and Parameter Inlining tools you now can observe waveforms and change parameters on the RT Box in real-time, as well as understand how to gather more information about the RT Box's operational state from the Web Interface.

You have learned how to use the CodeGen mode to observe the impacts of model discretization and check if model step size is appropriate before running your model on the RT Box. By structuring a model around a subsystem that represents the real-time portion of the model, you can directly compare offline, real-time, and CodeGen simulation results from within a single PLECS model.

With an understanding of these topics, you should now be able to develop your own real-time simulation models for both HIL and RCP applications.

References

[1] *RT Box User Manual*, Plexim GmbH, Jan. 2019, Online: https://www.plexim.com/sites/default/files/rtboxmanual.pdf

[2] *PLECS User Manual*, Plexim GmbH, Aug. 2018, Online: https://www.plexim.com/sites/default/files/plecsmanual.pdf

Revision History:

Tutorial Version 1.0	First release
Tutorial Version 1.1	Updated

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

RT Box Tutorial