Getting Started with the B&R MapleSim Connector

Getting Started with the B&R MapleSim Connector

Copyright

This document was produced using Maple and DocBook.

Contents

Introduction

The B&R MapleSim™ Connector provides all of the tools you need to prepare and export your dynamic systems models into B&R Automation Studio ANSI-C source code from a MapleSim model. You can create a model in MapleSim, simplify it in Maple™ by using an extensive range of analytical tools, and then generate the source code that you can incorporate into your toolchain.

Scope of Model Support

MapleSim is a comprehensive modeling tool where it is possible to create models that could go beyond the scope of this B&R Connector. In general, the B&R MapleSim Connector supports systems of any complexity, including systems of DAEs of any index, in any mix of domains.

System Requirements

For installation instructions and a complete list of system requirements, see the **Install.html** file on the product disc.

Distribution of an FMU

Under the terms and conditions of the Maplesoft End User License Agreement, you have the right to use an FMU created with the B&R MapleSim Connector within your organization. **If you wish to distribute this FMU outside your organization, you must first obtain a FMU Commercial Distribution License from Maplesoft.** For more information, see FMU Distribution.

1 Getting Started

1.1 Working with Apps

The B&R MapleSim Connector includes the following apps:

- B&R Automation Studio FMU Generation app — Generate a custom B&R FMU archive that can be used in B&R Automation Studio.
- B&R SERVOsoft Data Generation app — Generate a Microsoft® Excel spreadsheet that contains the operational data (time, velocity, thrust) for a motion driver in the model. The spreadsheet can be used by the SERVOsoft® drive design tool that's in B&R Automation Studio.

Other apps are included in MapleSim itself, such as the 1-D Motion Generation app.

This chapter describes how to use the B&R MapleSim Connector to generate an FMU archive for use in B&R Automation Studio.

The next chapter describes how to use the B&R SERVOsoft Data Generation app.

1.2 B&R FMU Generation Steps

In the *FMU Generation Example: Slider Crank Model (page 6)* section of this chapter, a step by step example shows you how to generate the FMU archive. The MapleSim Connector for B&R FMU Generation app consists of the following steps in generating FMU archive and is described in the sections below:

1. Subsystem selection
2. Inputs/Outputs and parameter management
3. Export Options
4. Info Options
5. Generate and export FMU archive

The BRConnector Package

The **BRConnector** package is a collection of procedures for generating a custom B&R FMU archive from MapleSim models, based on the model's algebraic equations and DynamicSystems objects.

For information about the BRConnector package, in Maple, refer to the BRConnector help page.

1.3 Opening the B&R FMU Generation App

To open the B&R FMU Generation app:

1. From the main toolbar, click **Show Apps Manager** (▦).
2. Double-click the **B&R Automation Studio FMU Generation** entry in the **Apps** palette. The **B&R FMU Generation** app opens.

1.4 Using the B&R FMU Generation App

The **B&R FMU Generation** app can be used to manipulate and export MapleSim subsystems.

With this app, you can define inputs and outputs for the subsystem, set the level of code optimization, generate the source code, and choose the format of the resulting FMU component and library code. You can use any Maple commands to perform task analysis, assign model equations to a variable, group inputs and outputs and define additional input and output ports for variables.

Note: FMU component generation now handles all systems modeled in MapleSim, including hybrid systems with defined signal input (RealInput) and signal output (RealOutput) ports.

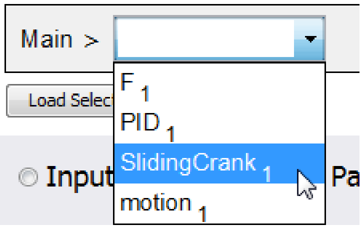
Example models are available in MapleSim. To access these, from the **Help** menu, select **Examples > BR Connector Examples**.

Step 1: Subsystem Selection

This part of the app identifies the subsystem modeling components that you want to export as a block component.

To connect a subsystem to modeling components outside of its boundary, you add subsystem ports to your model. A subsystem port is an extension of a component port in your subsystem. The resulting signals can then be directed as inputs and outputs for the FMU archive file. By creating a subsystem you improve the visual layout of a system in the model workspace and also prepare the model for export. The *Preparing the 2-D Rigid Slider Crank Model for Export Using Subsystems (page 8)* section in Chapter 2 shows you how to group all of the components into a subsystem.

You can select which subsystems from your model you want to export to an FMU archive file. To begin, select a subsystem.

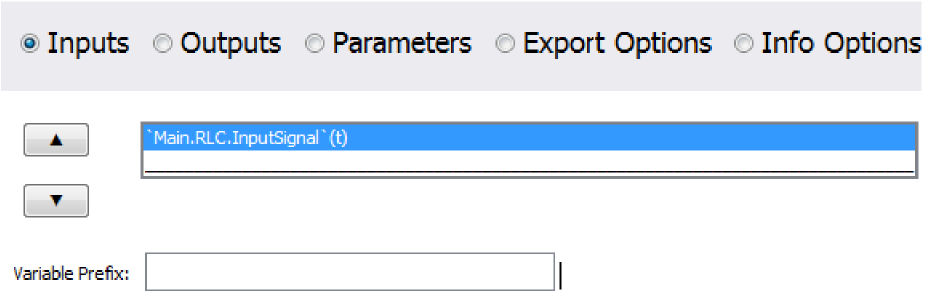


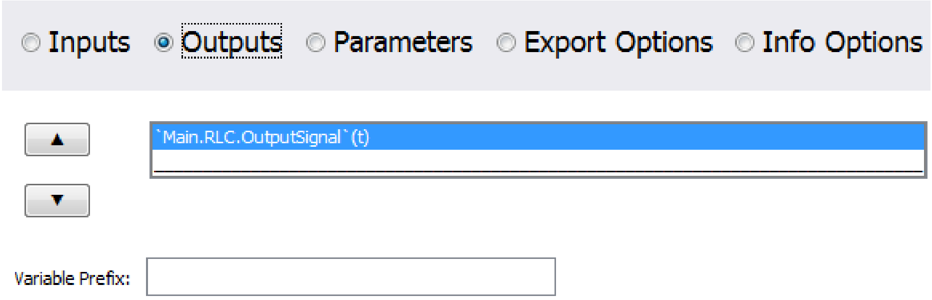
After selecting a subsystem, click **Load Selected Subsystem**. All defined input and output ports are loaded.

Step 2: Inputs, Outputs, and Parameters

The **Inputs**, **Outputs**, and **Parameters** sections let you customize, define, and assign parameter values to specific ports. Subsystem components to which you assign the parameter inherit a parameter value defined at the subsystem level.

After the subsystem is loaded you can group individual input and output variable elements into a vector array, and add additional input and output ports for customized parameter values. Input ports can include variable derivatives, and output ports can include subsystem state variables. You can specify prefixes for both input and output port variables. The prefix for the input (output) port variables will be applied to all the variables in the input ports (outputs) table.

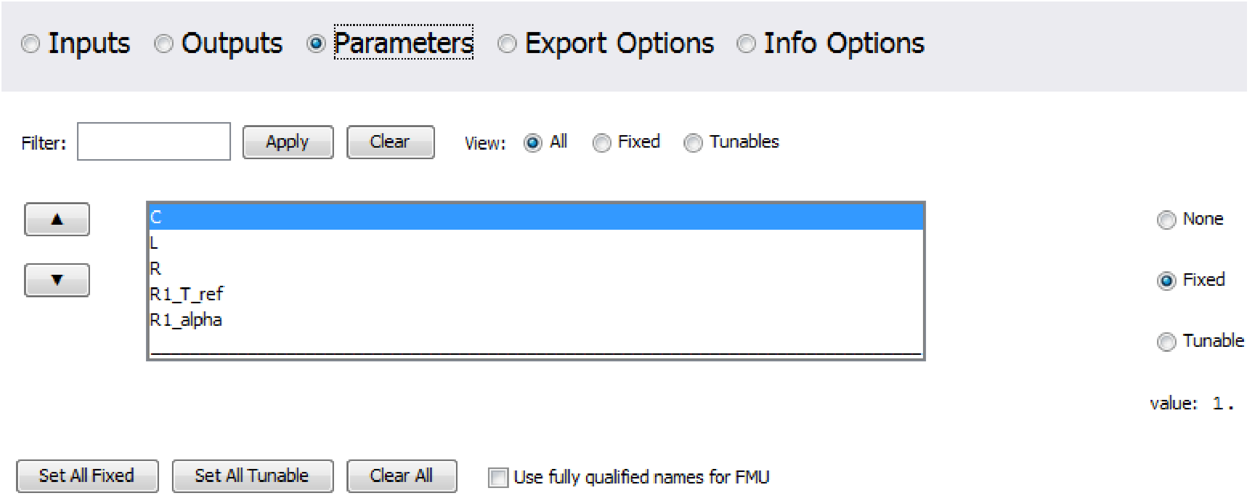

● Inputs ○ Outputs ○ Parameters ○ Export Options ○ Info Options
▲
▼
`Main.RLC.InputSignal`(t)
Variable Prefix:



○ Inputs ● Outputs ○ Parameters ○ Export Options ○ Info Options

`Main.RLC.OutputSignal`(t)

Variable Prefix:



○ Inputs ○ Outputs ● Parameters ○ Export Options ○ Info Options

Filter:

View: ● All ○ Fixed ○ Tunables

C
L
R
R1_T_ref
R1_alpha

○ None
● Fixed
○ Tunable

value: 1.

□ Use fully qualified names for FMU

Note: If the parameters are not marked for export they will be numerically substituted. Parameters marked as tunable will also be marked for export by default. However, tunable parameters will be ignored if FMI Version and Environment (see Step 3: Export Options) are anything other than FMI 2.0 and Co-Simulation. To use fully qualified parameter names for the generated FMU, check the corresponding check box; otherwise, short parameter names will be used by default.

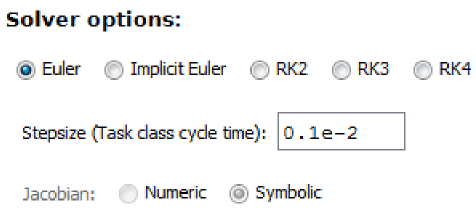
The Port and Parameter Management interface lets you customize, define and assign parameter values to specific ports. Subsystem components to which you assign the parameter inherit a parameter value defined at the subsystem level.

Step 3: Export Options

The export options include solver specifications and visualization options (for models with multibody components).

Solver Options

Specify the solver and step size.



Solver options:

● Euler ○ Implicit Euler ○ RK2 ○ RK3 ○ RK4

Stepsize (Task class cycle time): 0.1e-2

Jacobian: ○ Numeric ● Symbolic

For solver, select one of the following options:

Euler: *forward Euler* method

Implicit Euler: *implicit Euler* method

RK2: *second-order Runge-Kutta* method

RK3: *third-order Runge-Kutta* method

RK4: *fourth-order Runge-Kutta* method

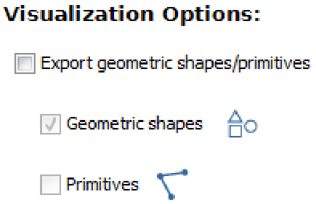
CK45: Variable Step CK45 solver (Semi - stiff)

Rosenbrock : Variable step stiff Rosenbrock solver

For the Implicit Euler (stiff) solver, you can select between a symbolic or numeric approximation to the system Jacobian.

Visualization Options

For models with multibody components, the B&R Generation app creates an FMU that includes a B&R specific scene file, which can be used to create an animation in B&R Automation Studio.



Visualization Options:

□ Export geometric shapes/primitives

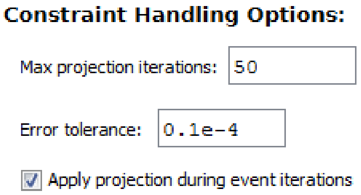
☑ Geometric shapes

□ Primitives

Select **Export geometric shapes/primitives** to control whether geometric shapes or primitives are exported.

Constraint Handling Options

The Constraint Handling Options specifies whether the constraints are satisfied in a DAE system by using constraint projection in the generated C code. Use this option to improve the accuracy of a DAE system that has constraints. If the constraint is not satisfied, the system result may deviate from the actual solution and could lead to an increase in error at an exponential rate.



Constraint Handling Options:

Max projection iterations: 50

Error tolerance: 0.1e-4

☑ Apply projection during event iterations

Set the **Max projection iterations** to specify the maximum number of times that a projection is permitted to iterate to obtain a more accurate solution.

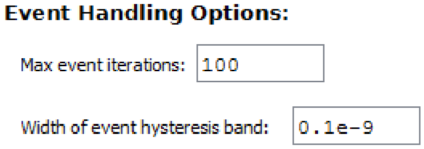
Set the **Error tolerance** to specify the desirable error tolerance to achieve after the projection.

Select **Apply projection during event iterations** to interpolate iterations to obtain a more accurate solution.

Constraint projection is performed using the **constraint projection** routine in the External Model Interface as described on The MathWorks™ website to control the drift in the result of the DAE system.

Event Handling Options

The Event Handling Options specifies whether the events are satisfied in a DAE system by using event projection in the generated C code. Use this option to improve the accuracy of a DAE system with events. If the constraint is not satisfied, the system result may deviate from the actual solution and could lead to an increase in error at an exponential rate.



Event Handling Options:

Max event iterations: 100

Width of event hysteresis band: 0.1e-9

Set the **Max event iterations** to specify the maximum number of times that a projection is permitted to iterate to obtain a more accurate solution.

Set the **Width of event hysteresis band** to specify the desirable error tolerance to achieve after the projection.

Event projection is performed using the **event projection** routine in the External Model Interface as described on The MathWorks website to control the drift in the result of the DAE system.

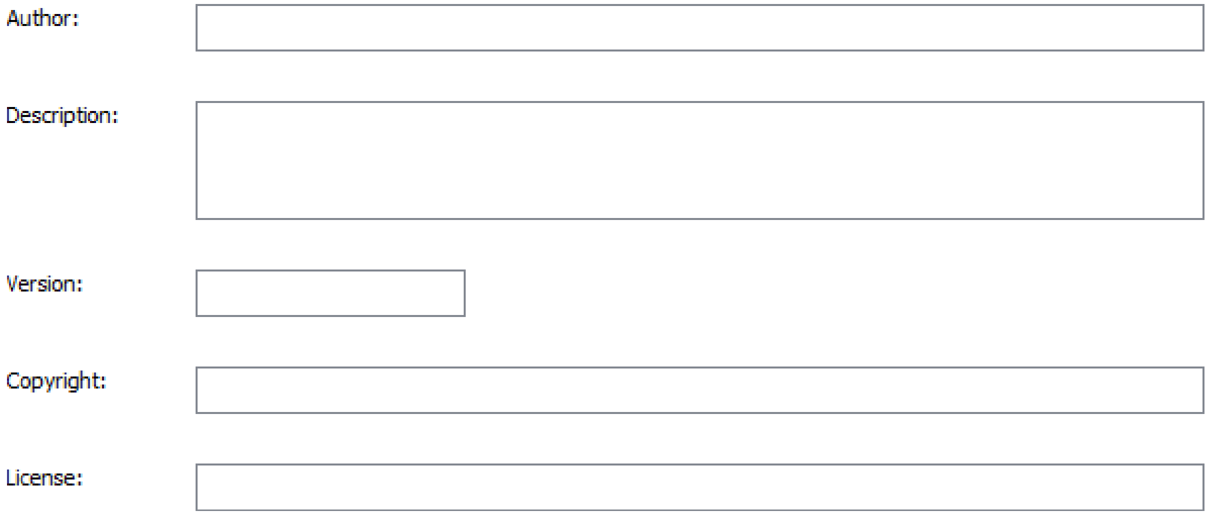
Note: Currently, if the model has events, they are handled using the event handling functions in the generated MsimModel.c file, and not the FMI provided Event Handling routines.

Run-time Error Reporting

Select the check box to report run-time errors in detail.

Step 4: Info Options

In the generated FMU archive file, model information (for example, variable names, units and initial values) are stored in a text file in XML format. Additional information about the model, such as the model author, description, and version, can also be included in this file.



Author:

Description:

Version:

Copyright:

License:

Author: Name and organization of the model author (for example, J. Smith, Maplesoft http://www.maplesoft.com).

Description: Brief description of the model (for example, Model of a lithium-ion battery).

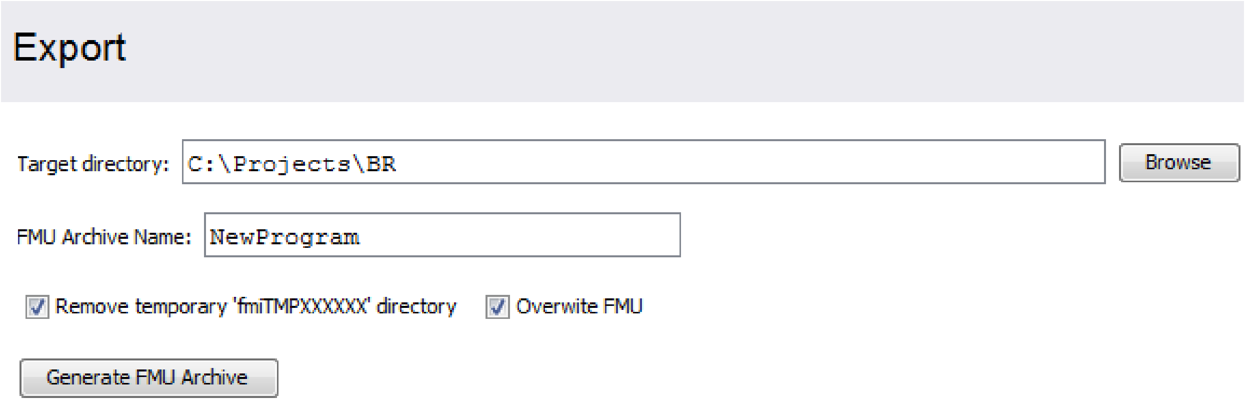
Version: Model version or FMU version.

Copyright: Intellectual property copyright (for example, (C) Maplesoft 2024).

License: Intellectual property licensing (for example, Proprietary, Public Domain, or BSD License).

Step 5: Export

Generating the FMU archive creates temporary files for viewing purposes.



Target directory: C:\Projects\BR

FMU Archive Name: NewProgram

☑ Remove temporary 'fmiTMPXXXXXX' directory ☑ Overwrite FMU

Specify the locations for the **Target Directory**.

Provide a name for the generated FMU archive in **FMU Archive Name**.

Select **Remove temporary `fmiTMPXXXXXX`directory** to remove temporary files after code generation.

Select **Overwrite FMU** to overwrite the file if it already exists.

To generate an FMU archive click **Generate FMU Archive**. The file is saved to the target directory.

1.5 Viewing Examples

Within MapleSim there are many examples for you to view.

To view an example:

1. From the **Help** menu, select the **Examples > BR Connector Examples** submenu.
2. Click the entry for the model that you want to view.

Note: Some models may include additional documents, such as templates that display model equations or define custom components. These are found in the **Attached Files** tab.

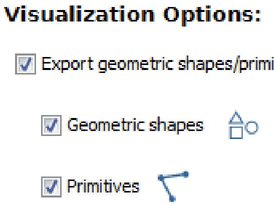
1.6 FMU Generation Example: Slider Crank Model

In this example, you will generate an FMU archive for a Slider Crank model created in MapleSim.

To generate the FMU archive:

1. From the **Help** menu, select the **Examples > BR Connector Examples** submenu, and select the **Slider Crank** example.
2. From the main toolbar, click **Show Apps Manager** (▦).
3. Double-click the **B&R Automation Studio FMU Generation** entry in the **Apps** palette. The **B&R FMU Generation** app opens.
4. In the **Subsystem Selection** section, select the **SlidingCrank1** subsystem from the drop-down menu in the toolbar above the model diagram. This menu displays all of the subsystems and components in your MapleSim model.

5. Click **Load Selected Subsystem**. All of the app fields are populated with information specific to the subsystem displayed in the model diagram. You can now specify which subsystem parameters will be kept as configurable parameters in the generated block.
6. Under the **Export Options** section, select all the **Visualization** options:



Visualization Options:

☑ Export geometric shapes/primitives

☑ Geometric shapes

☑ Primitives

7. In the **Export** section of the app, specify the target directory and name.
8. Click **Generate FMU Archive**. The FMU file is created and saved in the target directory.

Note: Generating a block may require a few minutes.

2 Further Examples

This chapter describes how to prepare a model for export by grouping the necessary components in a subsystem, shows how to use the B&R MapleSim Connector to export a spreadsheet of operational data for use by the B&R SERVOsoft drive train tool, and demonstrates the 1-D motion generation app.

2.1 Preparing the 2-D Rigid Slider Crank Model for Export Using Subsystems

Preparing a Model

This example starts with a slider crank model and prepares it for export by grouping the necessary components in a subsystem. At the end of the process, the example will look like the example model used in the *FMU Generation Example: Slider Crank Model (page 6)* section of the Chapter 1. This step is necessary before exporting your model to an FMU.

1. Convert the slider-crank mechanism model to a subsystem.

2. Define subsystem inputs and outputs.

3. Define and assign subsystem parameters.

To open the slider-crank mechanism example:

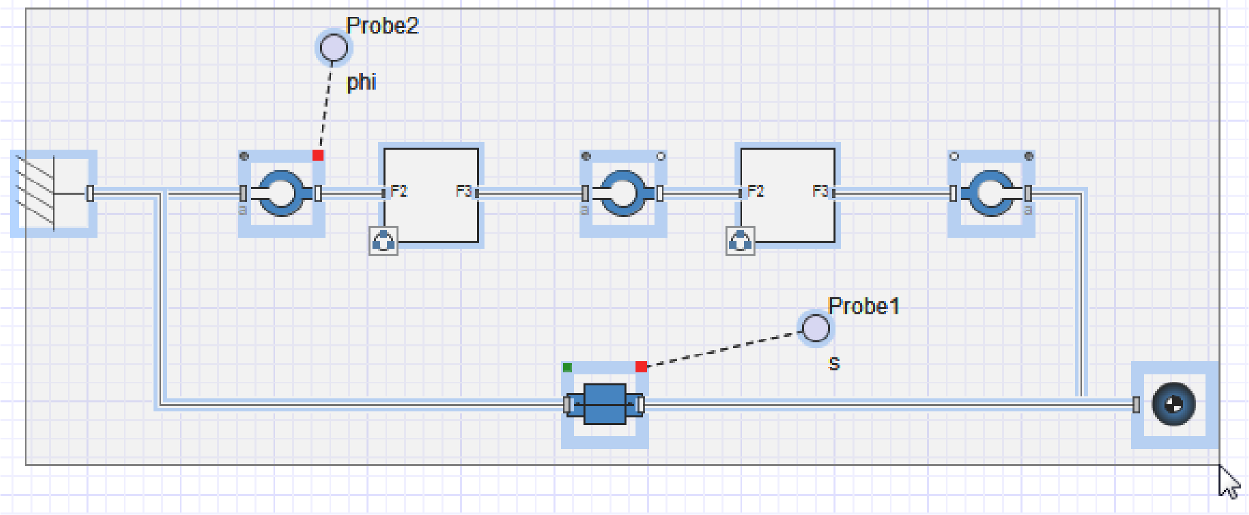
1. In MapleSim, under the **Help** menu, browse to the **Examples** > **User's Guide Examples > Chapter 6** menu.
2. Open the **Planar Slider-Crank Mechanism** example. The example appears in the **Model Workspace**.

Converting the Model to a Subsystem

By converting your entire model or part of your model into a subsystem, you identify which parts of the model that you want to generate C code for. In this example, you will group all of the components into a subsystem.

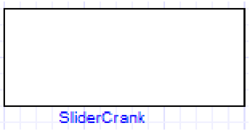
To create a subsystem:

1. Using the selection tool (↖) located in the **Model Workspace Toolbar**, draw a box around all of the components in the model.



2. From the **Edit** menu, select **Create Subsystem**.
3. In the **Create Subsystem** dialog box, enter **SliderCrank** as the subsystem name.

4. Click **OK**. A **SliderCrank** subsystem block appears in the **Model Workspace**.



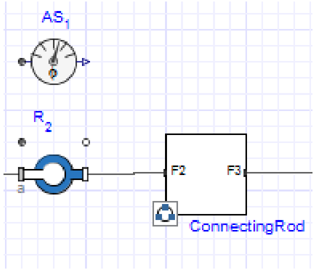
Defining Subsystem Inputs and Outputs

MapleSim uses a topological representation to connect interrelated components without having to consider how signals flow between them, whereas traditional signal-flow modeling tools require explicitly defined system inputs and outputs. Since the B&R MapleSim Connector only supports data signals, properties on acausal ports, such as mechanical flanges and electrical pins, must be converted to signals using the appropriate components. The resulting signals are directed as inputs and outputs for the subsystem in MapleSim and for the generated C code.

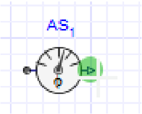
In this example, you will convert the displacements of the slider and the joint between the crank and connecting rod to output signals. The input signal needs to be converted to a torque that is applied to the revolute joint that represents the crank shaft. A spherical geometry component is also added for visualization purposes.

To create a subsystem output port:

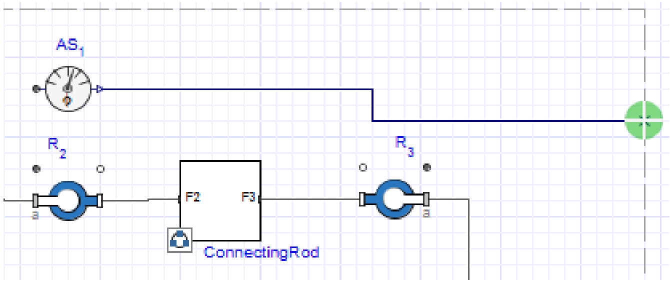
1. Double-click the subsystem block to view its contents. In this view, the broken line indicates the subsystem boundary. You can edit the connection lines and components within the boundary, and add subsystem ports to connect the subsystem to other components.
2. Delete the probes that are attached to the model.
3. Under the **Libraries** tab on the left side of the MapleSim window, expand the **1-D Mechanical** palette and then expand the **Rotational > Sensors** subpalette.
4. Drag the **Angle Sensor** component to the **Model Workspace** and place it above the **Revolute2** component.



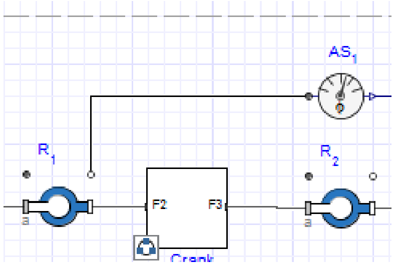
5. To connect the **Angle Sensor** component to the model, click the triangular y connector. The frame is highlighted in green when you hover your pointer over it.



6. Drag your pointer to the subsystem boundary.

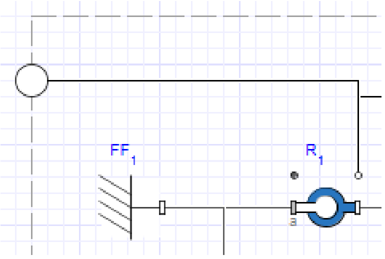


7. Click the boundary once. A real output port is added to your subsystem.
8. Connect the other end of the **Angle Sensor** to the white flange of revolute joint **Revolute1**.

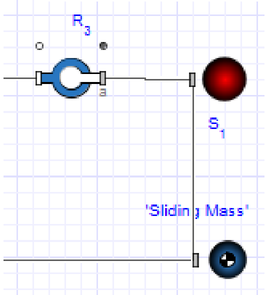


Since the B&R connector supports causal and acausal ports, an acausal input is added outside of the subsystem to control the torque on the crank shaft.

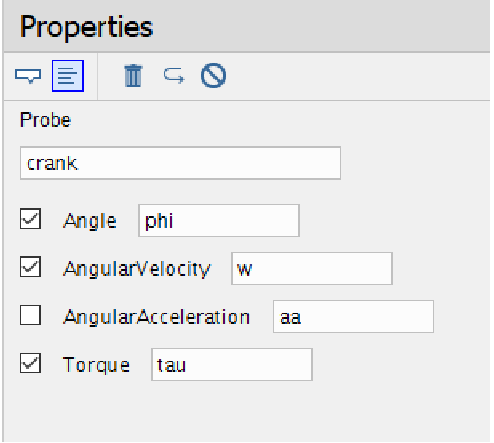
9. Click on the white flange of the leftmost **Revolute Joint R1** and drag the pointer to the left boundary. This will act as the acausal input connection to the subsystem.



10. From the **Multibody > Visualization** palette, select the **Spherical Geometry** component and add it above the Rigid Body component. Use **Edit > Flip Horizontal** to flip the component and then connect it to both **Revolute3** and the **Rigid Body** component.



11. Add a probe: Click **Attach Probe**(⊕) above the Model Workspace toolbar and then click the connection line from **Revolute1** to **Angle Sensor**. Click the probe once to position it.
12. In the **Properties** tab, name the probe **crank**, and select **Angle**, **Angular velocity**, and **Torque** to measure these quantities.



Properties

Probe

crank

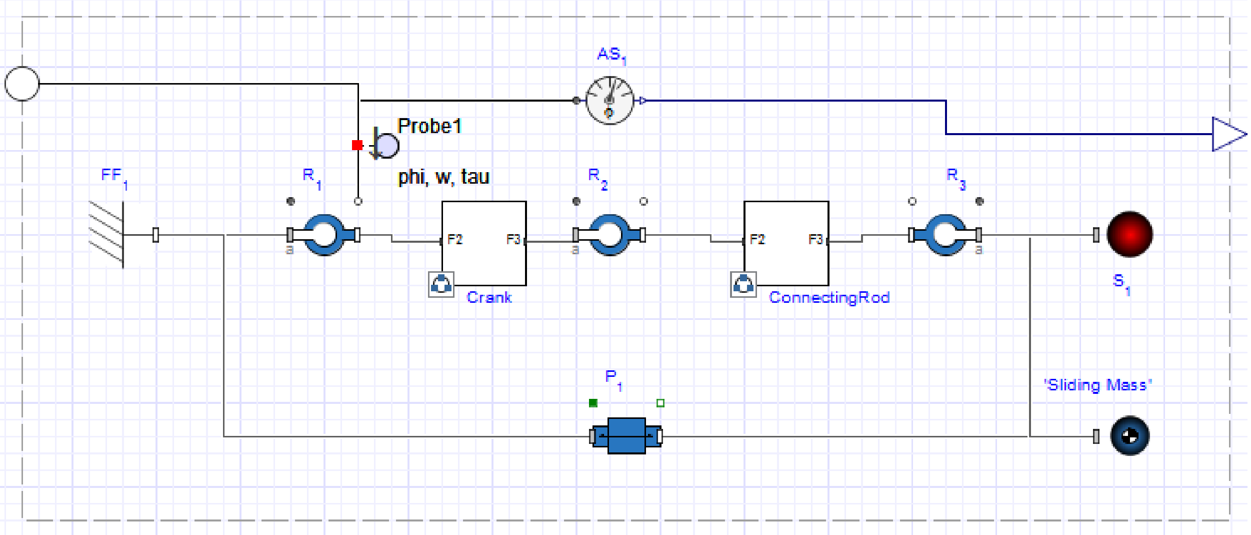
☑ Angle phi

☑ AngularVelocity w

☐ AngularAcceleration aa

☑ Torque tau

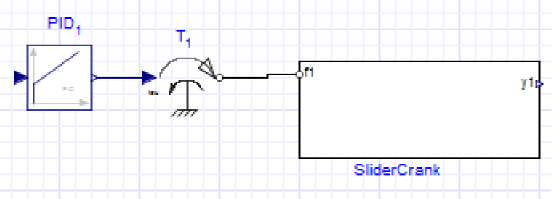
The complete subsystem appears below.



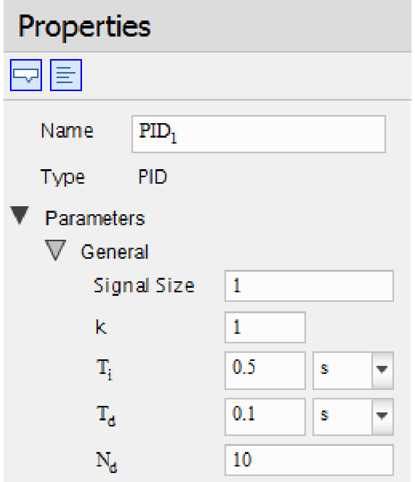
13. Click **Main** (⌂) in the **Model Workspace** toolbar to browse to the top level of the model.

14. From the **1-D Mechanical** > **Rotational** > **Torque Drivers** menu, add a **Torque** component to the **Model Workspace** and place it next to the subsystem. Connect its output port to the input port of the SliderCrank subsystem.

15. From the **Signal Blocks > Controllers** palette, drag a **PID** component into the model workspace and connect its output port to the input signal port of the **Torque** component. The result is shown below.



16. Click on the **PID** and in the **Properties** tab set **k** to 400, $\mathbf{T_i}$ to 0.5 s, and $\mathbf{T_d}$ to 0.1 s.


Properties

Name PID$_1$

Type PID

Parameters

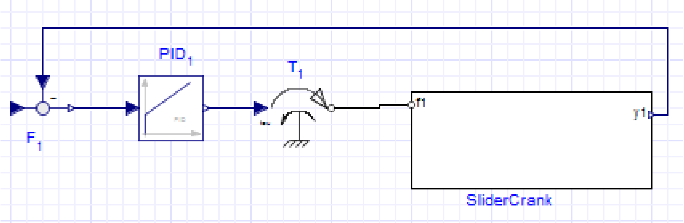
General

Signal Size	1
k	1
T_i	0.5 s
T_d	0.1 s
N_d	10

17. From the **Signal Blocks > Math > Operators** palette, drag a **Feedback** component into the model workspace and connect its output port to the input port of the **PID** component.

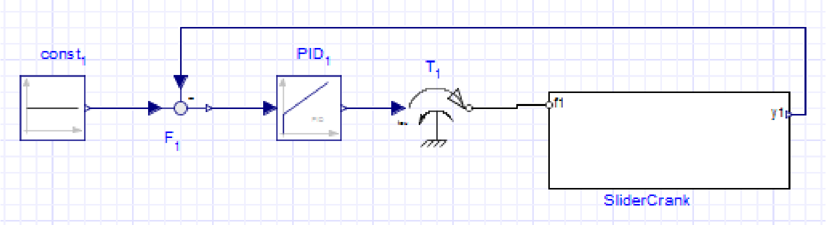
18. Click the **Feedback** component, and from the **Edit** menu, select **Flip Vertical**.

19. Connect the **Feedback** component to the **PID** component and the **SlidingCrank** subsystem to the Feedback component.



20. From the **Signal blocks>Common** menu, add a **Constant** signal component to the **Model Workspace** and connect it to the input of the **Feedback** component.

21. The model is prepared and ready to use with the apps in the B&R MapleSim Connector.

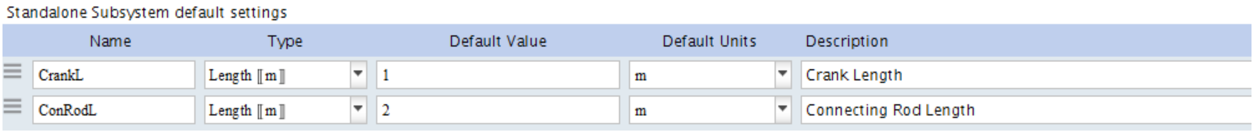


2.2 Defining and Assigning Subsystem Parameters

You can define custom parameters that can be used in expressions in your model to edit values more easily. To do so, you define a parameter with a numeric value in the parameter editor. You can then assign that parameter as a variable to the parameters of other components; those individual components will then inherit the numeric value of the parameter defined in the parameter editor. By using this approach, you only need to change the value in the parameter editor to change the parameter values for multiple components.

To edit parameters

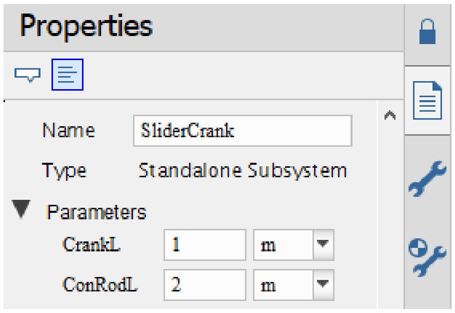
1. Double-click the **SliderCrank** component on the Model Workspace to see the detailed view of the **SliderCrank** subsystem, and then click **Parameters** (▦) in the **Model Workspace** toolbar. The parameter editor appears.
2. In the first **Name** field, type **CrankL** and press **Enter**.
3. Change the **Type** to Length from the drop-down menu.
4. Specify a default value of **1m** and enter **Crank length** as the description.
5. In the second row of the table, define a parameter called **ConRodL** and press **Enter**.
6. Change the **Type** to Length from the drop-down menu.
7. Specify a default value of **2m** and enter **Connecting Rod Length** as the description.



Standalone Subsystem default settings

Name	Type	Default Value	Default Units	Description
CrankL	Length ⟦m⟧	1	m	Crank Length
ConRodL	Length ⟦m⟧	2	m	Connecting Rod Length

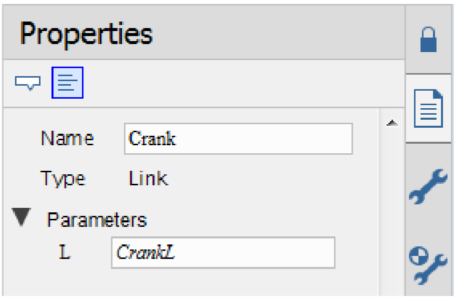
8. Click **Diagram View** (▦) to switch to the diagram view, and then click **Main** (⌂).
9. Select the **SliderCrank** subsystem. The parameters are defined in the **Properties** tab (🗎).



Properties

Name	SliderCrank	
Type	Standalone Subsystem	
▼ Parameters		
CrankL	1	m
ConRodL	2	m

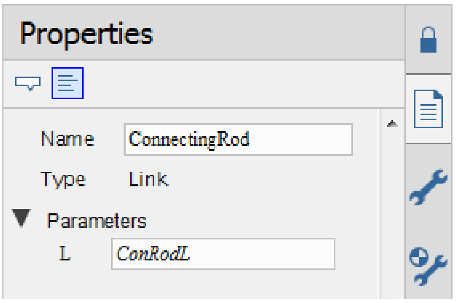
10. Double-click the **SliderCrank** subsystem, and then select the **Crank** subsystem.
11. In the **Properties** tab (🗎), change the length value (**L**) to **CrankL**. The **Crank** subsystem now inherits the numeric value of **CrankL** that you defined.



Properties

Name	Crank
Type	Link
▼ Parameters	
L	*CrankL*

12. Select the **ConnectingRod** subsystem and change its length value to **ConRodL**.



Properties

Name ConnectingRod

Type Link

▼ Parameters

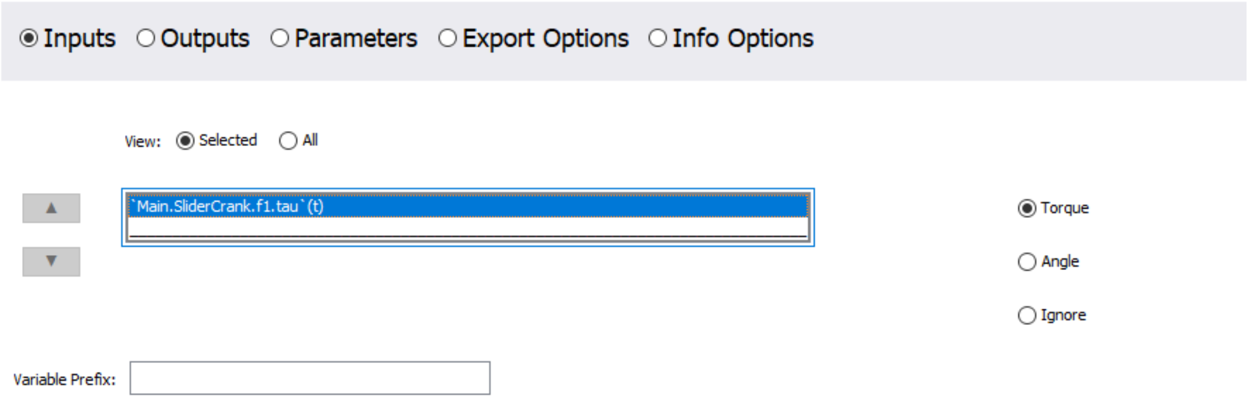
L ConRodL

13. Click **Main** ( ) in the **Model Workspace** toolbar to navigate to the top level of the model. You will include these parameter values in the model that you export. You are now ready to convert your model in the B&R Connector App.

2.3 Exporting Your Model Using the B&R Automation Studio FMU Generation App

After preparing the model, you can use the B&R Automation Studio FMU Generation app to set export options and convert the model to an FMU file. In this example, the Slider Crank Subsystem is exported as an FMU block for which the controller can be designed in the Automation Studio.

1. From the main toolbar, click **Show Apps Manager** ( ).
2. Double-click on the **B&R Automation Studio FMU Generation** entry under B&R connector in the **Apps** palette. The **B&R FMU Component Code Generation** app opens.
3. In the **Subsystem Selection** section, select the **SliderCrank** subsystem from the drop-down list, and then click **Load Selected Subsystem**. All of the app fields are populated with information specific to the subsystem.
4. Since this example uses an acausal 1D - mechanical rotational flange, on selecting the **input** option, you have option to either choose a torque or an angle input that can be fed into the subsystem.



● Inputs ○ Outputs ○ Parameters ○ Export Options ○ Info Options

View: ● Selected ○ All

`Main.SliderCrank.f1.tau`(t)

● Torque

○ Angle

○ Ignore

Variable Prefix:

5. In the **Export** section, specify the location for the **Target Directory** and provide a name for the generated **FMU Archive.**
6. Click **Generate FMU Archive** to generate the .FMU zip file.

2.4 Generating Operational Data Model

In this example, you will use the B&R SERVOsoft Data Generation app to create a spreadsheet of the velocity/torque characteristics needed to drive the slider crank model.

1. Open the Slider Crank example created in the above section,
2. Load the B&R SERVOsoft Data Generation app.
3. Define app settings.
4. Simulate and display the graph.
5. Export data to a spreadsheet.

To open the Slider Crank example and load the app:

1. Open the Slider Crank example created in the previous session.
2. From the main toolbar, click **Show Apps Manager** (▦).
3. Double-click the **B&R SERVOsoft Data Generation** entry from the **B&R connector** in the **Apps** palette. The **B&R SERVOsoft Data Generation** app opens.
4. In the app, click **Load System**. All of the app fields are populated with information specific to your model. You can now specify the app settings.

To define the app settings:

1. In the **Actuator and Export** section, select **** Probes **** from the Device drop-down menu, click the **Rotational** radio button, and set the **Probe**, **Velocity**, and **Thrust** to be crank.phi, crank.w, and crank.tau, respectively. Note the default units, which are appropriate for this example.
2. In the **Export Options** section, ensure the time is from 0 to 2.

To simulate and display the graph:

1. Click **Preview Export(Simulate)**. The simulation runs and the graph is displayed.
2. Click the **Thrust vs Velocity** radio button to view this graph.

To export the data to a spreadsheet:

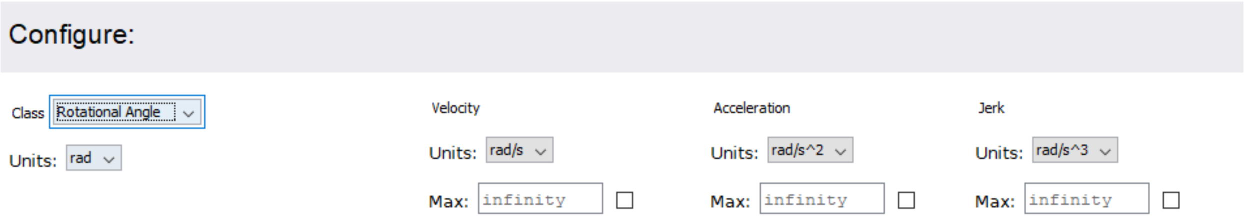
1. Specify the locations for the **Target Directory**.
2. Provide a name for the generated XLS file in **Filename**.
3. Click **Export**. The XLS files are created and saved in the target directory.

Note: Generating a block may require a few minutes.

2.5 Using the 1-D Motion Generation App

The example shows how to create a motion profile using the 1-D Motion Generation app in MapleSim.

1. From the main toolbar, click **Show Apps Manager** (▦).
2. Double-click the **1-D Motion Generation** entry in the Apps palette.
3. Change the class to **Rotational Angle** from the Configure section. Note the default units, which are appropriate for this example.



Configure:

Class Rotational Angle

Units: rad

Velocity

Units: rad/s

Max: infinity □

Acceleration

Units: rad/s^2

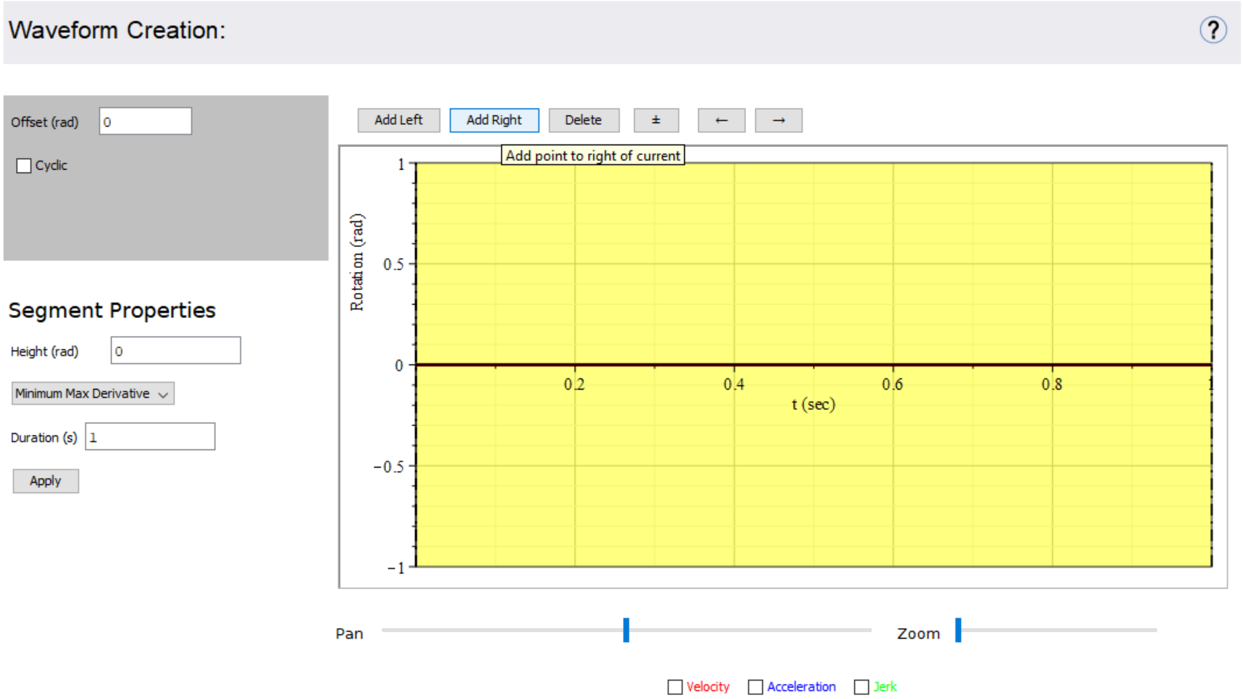
Max: infinity □

Jerk

Units: rad/s^3

Max: infinity □

4. select **Add Right** from the **Waveform Creation.**



Waveform Creation:

Offset (rad) 0

□ Cyclic

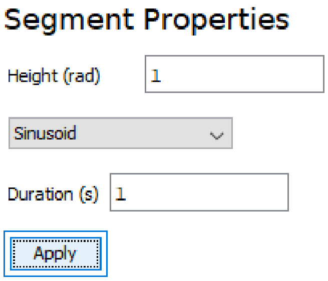
Segment Properties

Height (rad) 0

Minimum Max Derivative

Duration (s) 1

5. Change the **Height** from **Segment Properties** to **1** and choose **Sinusoid** from the drop down menu and click **Apply.**



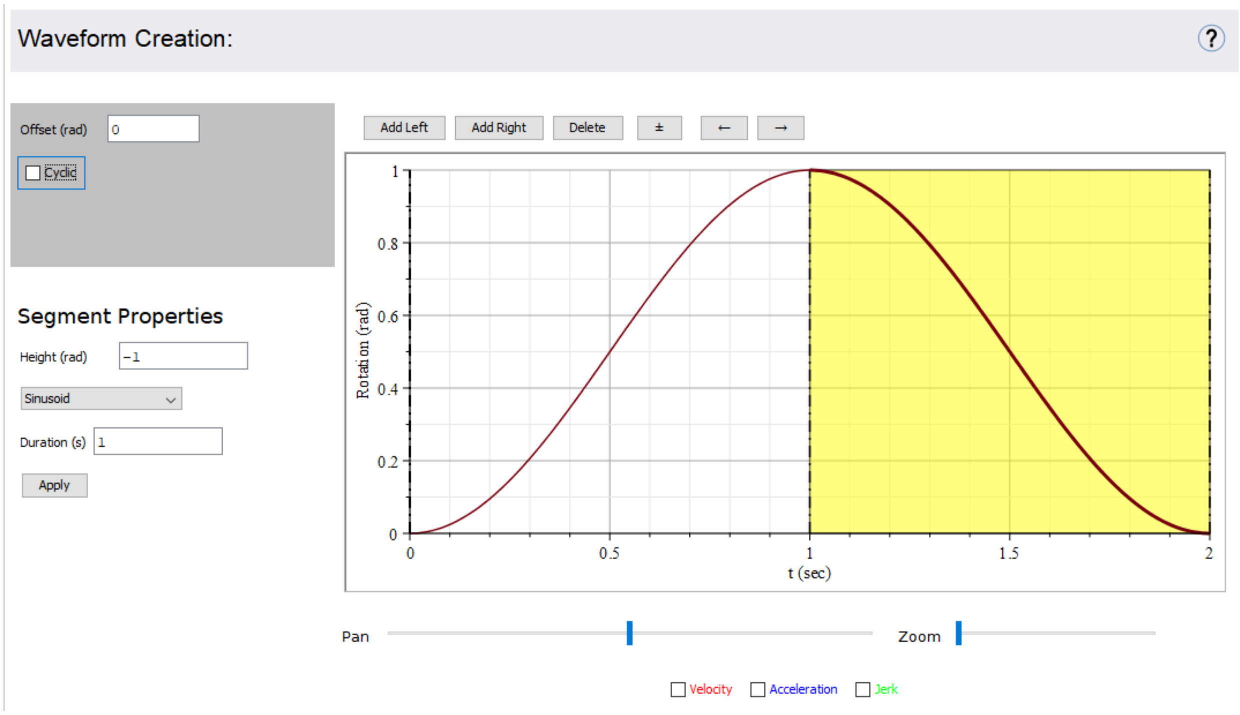
Segment Properties

Height (rad) 1

Sinusoid

Duration (s) 1

6. Click on **Add Right** again and change the **Height** to **-1** from the **Segment Properties.**



Waveform Creation:

Offset (rad) 0

□ Cyclic

Segment Properties

Height (rad) -1

Sinusoid

Duration (s) 1

Apply

7. Change the **Model Name** to **Motion** under **Model Generation.**
8. And click **Generate.**
9. The new motion profile created will be seen in the **Local Components** tab (🔍) in MapleSim.

Index