Model Interface Package for Simulink

API Reference

For Model Interface Package for Simulink 4.5

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/go/patches for software updates and patches.

Important Notice

Contents

About This Reference

Content

This reference introduces you to the API functions of the Model Interface Package for Simulink.

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
?	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
📖	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

`%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>`

or

`%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>`

Documents folder A standard folder for user-specific documents.

`%USERPROFILE%\Documents\dSPACE\<ProductName>\<VersionNumber>`

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

`%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>`

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

Managing Messages via the dsmpb.MsgContainer Class

Introduction

The Model Interface Package for Simulink provides the dsmpb.MsgContainer class to inform you whether errors or warnings have occurred.

dsmpb.MsgContainer Class

Purpose

To inform you whether errors or warnings have occurred during your work.

Description

The Model Interface Package for Simulink provides the dsmpb.MsgContainer class that implements a container. This container includes notes, warnings, and error messages. When you are working with the Model Interface Package for Simulink, it returns objects of this class to inform you whether errors or warnings have occurred. In custom scripts, you can use these objects for your error management.

To display a detailed description on the dsmpb.MsgContainer class in a Simulink Help window, type the following command in the MATLAB Command Window:

```
doc dsmpb.MsgContainer
```

Model Interface Package for Simulink API Reference

Where to go from here

Information in this section

Overview of API Functions

Alphabetical List of API Functions

API Functions	Purpose
`dsmpb_addsignal(block, signalPath)`	Adds a new signal.
`dsmpb_assignbusobjects(varargin)`	Assigns Simulink.Bus objects that match signal configuration of a Data Port block.
`dsmpb_configurerootslports(model)`	Sets parameter of root level port blocks to explicit values.
`dsmpb_copyblockwithids(source, destination, varargin)`	Copies the block and keep IDs of model port blocks.
`dsmpb_createinversedmodelportblock(dataPortBlocks, targetSystem)`	Creates inverse data port blocks.
`dsmpb_createmodelportblockperiphery(dataPortBlocks)`	Generates a block periphery for data port blocks.
`dsmpb_generatemodelportblock(command, varargin)`	Generates Data Inport or Data Outport blocks.
`dsmpb_generatemodelportblock('DataInportBlockFromBusCreator', busCreator)`	This command creates a Data Outport block whose signal configuration matches the output signal of a BusCreator block. The block will be generated to the system where the BusCreator resides.
`dsmpb_generatemodelportblock('DataInportBlockFromBusObjects', sys, busObjectNames)`	This command creates a Data Inport block whose signal configuration matches Simulink.Bus objects.
`dsmpb_generatemodelportblock('DataOutportBlockFromBusCreator', busCreator)`	This command creates a Data Inport block whose signal configuration matches the output signal of a BusCreator block. The block will be generated to the system where the BusCreator resides.
`dsmpb_generatemodelportblock('DataOutportBlockFromBusObjects', sys, busObjectNames)`	This command creates a Data Outport block whose signal configuration matches Simulink.Bus objects.
`dsmpb_get(blocks, [signalPath], propertyName)`	Gets properties from model port blocks.
`dsmpb_mdlcleanup(model)`	Removes model port lib data from a model.
`dsmpb_migrate(model)`	Migrates a Simulink model or library to the current MIPS version.

API Functions	Purpose
`dsmpb_pref(command, varargin)`	Manages MIPS preferences.
`dsmpb_pref('Get')`	This command gets the values of all preferences.
`dsmpb_pref('Get', preferenceName)`	This command gets the value of a specified preference.
`dsmpb_pref('Save')`	This command saves the preferences to file. The current values are then available in the next MATLAB session.
`dsmpb_pref('Set', preferenceName, preferenceValue)`	This command sets the value of a specified preference.
`dsmpb_rmsignal(block, signalPath)`	Removes a signal or port.
`dsmpb_set(block, [signalPath], propertyName, propertyValue)`	Sets properties in model port blocks.
`dsmpb_updatemodelportblock(command, varargin)`	Updates the signal configuration of data port blocks.
`dsmpb_updatemodelportblock('UpdateFromBusObjects', dataPortBlock, busObjectNames)`	This command updates the signal configuration of a data port block to match specified Simulink.Bus objects.
`dsmpb_updatemodelportblock('UpdateFromInputs', dataOutportBlocks)`	This command updates the signal configuration of Data Outport blocks to match the input signals. This command requires that all Data Outport blocks reside in the same model and that the model can be initialized. Signal configurations associated with an unconnected block port remain unmodified.
`dsmsb_demo(modelName)`	Opens Model Separation demo model.
`dsmsb_separate(varargin)`	Runs model separation.
`dsrt_addfiles(files, varargin)`	Adds files to a code generation result.
`dsrt_build(varargin)`	dSPACE real-time target (DSRT) code generation API

dsmpb_addsignal

dsmpb_addsignal

Purpose

Adds a new signal.

Description

This functions adds a new signal or port to a Data Inport or Data Outport block.

Syntax

```
msgContainer = dsmpb_addsignal(block, signalPath)
```

Example

```
Adds a new port:
msgContainer = dsmpb_addsignal(block, 'signal2');

Adds a new  'a' signal to signal2:
msgContainer = dsmpb_addsignal(block, {'signal2','a'})
msgContainer = dsmpb_addsignal(block, 'signal2.a')

Adds a new port and creates a tree hierarchy in a 'signal' signal:
msgContainer = dsmpb_addsignal(block, {'data3','level1','level2','signal'})
```

Input parameters

The following input parameters are available:

Parameter	Description
`block`	Model port block.
`signalPath`	Signal path of the new signal: ▪ Cell array with strings including signal names. ▪ String with dot notation. However, this notation might have problems with dots in signal names.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks

If the parent signal is an unstructured signal, the parent signal is converted to a structured signal while the unstructured signal is moved to the level beneath.

dsmpb_assignbusobjects

dsmpb_assignbusobjects

Purpose

Assigns Simulink.Bus objects that match signal configuration of a Data Port block.

Description

This function enables to convert untyped buses defined by a Data Inport or Outport block port configuration to typed buses by assigning Simulink.Bus objects to the block ports. The functions creates the objects and/or looks for matching Simulink.Bus that it can assign.

If the model is associated with a Data Dictionary, the function processes objects in this Data Dictionary, otherwise, the base workspace.

Preconditions for creating Simulink.Bus objects:

- All signal names must be valid MATLAB identifiers.
- Signal widths must be specified by numeric expressions (as opposed to workspace variables).

Newly created Simulink.Bus objects are saved in the base workspace, or the Data Dictionary of the model in which the block resides. The function makes sure not to overwrite existing variables.

For each newly created Simulink.Bus object, the function produces a message.

When Simulink.Bus objects have been created into the base workspace, you must make sure to save them with the model, for example, in a MAT file. The function cannot do this for you.

When called with no output arguments, messages produced during object creation are printed to the MATLAB Command Window. Otherwise, the function returns a dsmpb.MsgContainer object that contains the messages.

Syntax

```
msgContainer = dsmpb_assignbusobjects(varargin)
```

Example

-

Input parameters

-

Property value pairs

Additional function parameters are expected as propertyName/propertyValue pairs, refer to the following table:

Parameter	Description
`'block'`	Data port block whose signal configuration data must be processed.
`'portIdx'`	One-based indices of block ports that must be processed (default = all ports)
`'considerExistingBusObjects'`	If true, existing Simulink.Bus objects that match a port signal configuration are assigned rather than new Simulink.Bus objects created. Therefore, when signal configurations of block ports are identical, the same Simulink.Bus object is assigned to all ports. (default = true)
`'createBusObjects'`	If true, the function creates Simulink.Bus objects if no matching object exists. (default = true)

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages.

Remarks

-

dsmpb_configurerootslports

dsmpb_configurerootslports

Purpose

Sets parameter of root level port blocks to explicit values.

Description

The function sets the following parameters of all port blocks at the root level of a specified Simulink model to explicit values:

- OutDataTypeStr
- PortDimensions
- SignalType
- Unit
- VarSizeSig
- SignalType

Explicit values means that if any of these parameters specifies inheritance, it is set to the actual value, which is retrieved by compiling the model. This requires that the model can be initialized.

When the function is called up without output parameters, all messages are printed to the MATLAB Command Window.

Syntax

```
msgContainer = dsmpb_configurerootslports(model)
```

Example

-

Input parameters

The following input parameters are available:

Parameter	Description
`model`	Simulink model

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks

-

dsmpb_copyblockwithids

dsmpb_copyblockwithids

Purpose

Copies the block and keep IDs of model port blocks.

Description

This function creates a copy of the Simulink block. The block and signal IDs of all model port blocks in the copy are identical to those in the source block.

This function corresponds with the Paste and Keep IDs menu command.

Syntax

```
hBlock = dsmpb_copyblockwithids(source, destination, varargin)
```

Example

```
% Copies a block:
hBlock = dsmpb_copyblockwithids(myBlockHandle, [bdroot '/someBlock']);

% Copies a block ensuring a unique name:
hBlock = dsmpb_copyblockwithids('myModel/myBlock', 'myModel/myBlock', 'MakeNameUnique', 'on');

% Copies a block specifying a block position and ensuring a unique name:
hBlock = dsmpb_copyblockwithids('myModel/myBlock', 'myOtherModel/myBlock', 'MakeNameUnique', 'on', 'Position',[100 100
300 300]);
```

Input parameters

The following input parameters are available:

Parameter	Description
`source`	Source block.
`destination`	The Simulink path of the destination block that must be created.

Property value pairs

Additional function parameters are expected as propertyName/propertyValue pairs, refer to the following table:

Parameter	Description
`'MakeNameUnique'`	[on or off] Lets Simulink assign a unique name to the new block.
`'Position'`	The block position of the destination. The position argument is a (1, 4) vector [left, top, right, bottom].

Parameter	Description
`'propertyName'`	Any propertyName/value pair supported by Simulink's add_block command.

Output parameters

The following output parameters are available:

Parameter	Description
`hBlock`	Handle of the newly created block.

Remarks

Invalid parameters such as

- properties not supported by Simulink's add_block command

- invalid value for Simulink property

- read-only destination system

- undefined source block

will result in a script abort.

dsmpb_createinversedmodelportblock

dsmpb_createinversedmodelportblock

Purpose

Creates inverse data port blocks.

Description

This function creates inverse data port blocks. This means that for each specified Data Inport block a matching Data Outport block is created and vice versa. Matching means that the block signal configurations are identical.

When called without output parameters, messages are displayed in a message box.

Syntax

```
[msgContainer, hDataPortBlocks] = dsmpb_createinversedmodelportblock(dataPortBlocks, targetSystem)
```

Example

-

Input parameters

The following input parameters are available:

Parameter	Description
`dataPortBlocks`	Data Inport or Data Outport blocks.
`targetSystem`	System (subsystem or model) in which the blocks must be generated. (optional, default = subsystems or models from which the DataPort blocks originate)

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hDataPortBlocks`	Column vector with handles of blocks that were generated.

Remarks	-

dsmpb_createmodelportblockperiphery

dsmpb_createmodelportblockperiphery

Purpose

Generates a block periphery for data port blocks.

Description

This function generates a block periphery for each unconnected port of a data port block according to following rules:

- for each unstructured data port of a Data Outport block, a Simulink Constant block is generated.
- for each structured data port of a Data Outport block, a Simulink Subsystem block is generated that contains a representation of the bus hierarchy (via Constant, Bus Creator and Outport blocks as well as suitably named signals).
- for each unstructured data port of a Data Inport block, a Simulink Terminator block is generated.
- for each structured data port of a Data Inport block, a Simulink Subsystem block is generated that separates the bus hierarchy into its individual signals (via Inport, Bus Selector, and Terminator blocks as well as signals).

The generated blocks are automatically connected.

The value, type, and width of a generated Simulink Constant block are set according to the signal of the Data Outport block.

Syntax

```
msgContainer = dsmpb_createmodelportblockperiphery(dataPortBlocks)
```

Example

-

Input parameters

The following input parameters are available:

Parameter	Description
`dataPortBlocks`	Handles or block paths of data port blocks. [handle, block path, vector of handles, cell array of block paths]

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.

Remarks

-

dsmpb_generatemodelportblock

Where to go from here

Information in this section

dsmpb_generatemodelportblock

Purpose

Generates Data Inport or Data Outport blocks.

Syntax overview

The following syntaxes are available:

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataOutportBlockFromBusObjects', sys,
busObjectNames)
```

This command creates a Data Outport block whose signal configuration matches Simulink.Bus objects.

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataInportBlockFromBusObjects', sys,
busObjectNames)
```

This command creates a Data Inport block whose signal configuration matches Simulink.Bus objects.

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataOutportBlockFromBusCreator',
busCreator)
```

This command creates a Data Inport block whose signal configuration matches the output signal of a BusCreator block.
The block will be generated to the system where the BusCreator resides.

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataInportBlockFromBusCreator',
busCreator)
```

This command creates a Data Outport block whose signal configuration matches the output signal of a BusCreator block.
The block will be generated to the system where the BusCreator resides.

Example -

Remarks -

dsmpb_generatemodelportblock('DataInportBlockFromBusCreator', busCreator)

Purpose

Generates Data Inport or Data Outport blocks.

Syntax

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataInportBlockFromBusCreator',
busCreator)
```

Description

This command creates a Data Outport block whose signal configuration matches the output signal of a BusCreator block.

The block will be generated to the system where the BusCreator resides.

Input parameters

The following input parameters are available:

Parameter	Description
`busCreator`	Simulink identifier of BusCreator block. The model in which the BusCreator resides must be initializable, i.e., it cannot be a library.

Property value pairs -

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hModelPortBlock`	Handle of newly created model port block.

dsmpb_generatemodelportblock('DataInportBlockFromBusObjects', sys, busObjectNames)

Purpose

Generates Data Inport or Data Outport blocks.

Syntax

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataInportBlockFromBusObjects', sys,
busObjectNames)
```

Description

This command creates a Data Inport block whose signal configuration matches Simulink.Bus objects.

Input parameters

The following input parameters are available:

Parameter	Description
`sys`	Simulink model or subsystem.
`busObjectNames`	Names of Simulink.Bus objects in the base workspace or the model's Data Dictionary.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hModelPortBlock`	Handle of newly created model port block.

dsmpb_generatemodelportblock('DataOutportBlockFromBusCreator', busCreator)

Purpose

Generates Data Inport or Data Outport blocks.

Syntax

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataOutportBlockFromBusCreator',
busCreator)
```

Description

This command creates a Data Inport block whose signal configuration matches the output signal of a BusCreator block.

The block will be generated to the system where the BusCreator resides.

Input parameters

The following input parameters are available:

Parameter	Description
`busCreator`	Simulink identifier of BusCreator block. The model in which the BusCreator resides must be initializable, i.e., it cannot be a library.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hModelPortBlock`	Handle of newly created model port block.

dsmpb_generatemodelportblock('DataOutportBlockFromBusObjects', sys, busObjectNames)

Purpose

Generates Data Inport or Data Outport blocks.

Syntax

```
[msgContainer, hModelPortBlock] = dsmpb_generatemodelportblock('DataOutportBlockFromBusObjects', sys,
busObjectNames)
```

Description

This command creates a Data Outport block whose signal configuration matches Simulink.Bus objects.

Input parameters

The following input parameters are available:

Parameter	Description
`sys`	Simulink model or subsystem.
`busObjectNames`	Names of Simulink.Bus objects in the base workspace or the model's Data Dictionary.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hModelPortBlock`	Handle of newly created model port block.

dsmpb_get

dsmpb_get

Purpose

Gets properties from model port blocks.

Description

This function returns the value of a specified model port block property.

Syntax

```
propertyValue = dsmpb_get(blocks, [signalPath], propertyName)
```

Example

```
Gets the block ID:
propertyValue = dsmpb_get(block, 'BlockId')

Gets the signal ID of port 'data1':
propertyValue = dsmpb_get(block, 'data1', 'SignalId')

Gets the signal ID of signal 'signal1' from bus 'bus1':
propertyValue = dsmpb_get(block, {'bus1','signal1'}, 'SignalId')

Gets the description of signal 'signal1' from bus 'bus1' using dot notation to indicate the signal path:
propertyValue = dsmpb_get(block, 'bus1.signal1', 'Description')

Gets the signal names of each element in the bus 'bus1':
propertyValue = dsmpb_get(block, 'bus1', 'Elements')
```

Input parameters

The following input parameters are available:

Parameter	Description
`blocks`	Model port blocks.
`[signalPath]`	Optional argument for signal specific properties. Signal path from which to get property: ▪ Cell array with strings including signal names. ▪ Example: {'Bus1','signalA'} ▪ String with dot notation. However, this notation might have problems with dots in signal names. ▪ Example: 'Bus1.signalA'
`propertyName`	Property of model port blocks. Case-insensitive character vector. Data Inport and Data Outport blocks

Parameter	Description
	- Block properties: ▪ SampleTime - MATLAB expression that defines the sample time. ▪ Description - Block description. ▪ BlockId - Block identifier. ▪ DisplayIds - 0 or 1: Specifies whether IDs should be displayed in the model. ▪ Signals - Block leaf signal paths (Column cell array). ▪ BusSignals - Block bus signal paths (Column cell array). ▪ Elements - Block port signal names (Column cell array). ▪ GotoTags - Block port signal Goto tags (Column cell array). - Signal properties: ▪ Description - Signal description. ▪ Name - Signal name. ▪ Unit - Signal unit. ▪ DataType - Signal data type. ▪ Width - MATLAB expression that defines the signal width. ▪ SignalId - Signal identifier. ▪ Signals - Leaf signal paths in the specified signal (Column cell array). ▪ BusSignals - Bus signal paths in the specified signal (Column cell array). ▪ Position - Position (one based index) of signal in its bus signal parent element list. ▪ Elements - Name of bus elements (Column cell array). - Only for Data Inport blocks ▪ InitialValue - MATLAB expression that defines the initial value. ▪ NonVirtualBus - 0 or 1: Specifies whether a signal is a non-virtual bus. Ignored when signal is not a typed bus, e.g., the data type is not defined by a Simulink.Bus object. ▪ VariableSize - 0 or 1: Specifies whether a signal is variable-size. Ignored when signal is a bus. - Only for data port blocks: ▪ GotoTag - Goto tag of the block port signal. - Runnable Function blocks properties ▪ SampleTime - MATLAB expression that defines the sample time. ▪ Description - Block description. ▪ BlockId - Block identifier. ▪ SignalId - Runnable function identifier. ▪ Name - Runnable function name. ▪ TaskPriority - MATLAB expression that defines the runnable function priority. ▪ Goto tag - Goto tag of the block.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`propertyValue`	Value of the specified property.

Remarks

The data type of untyped bus signals is an empty string ''.

dsmpb_mdlcleanup

dsmpb_mdlcleanup

Purpose

Removes model port lib data from a model.

Description

This function removes all entries in model that belong to model port lib including blocks from the dsmpblib library.

Syntax

```
msgContainer = dsmpb_mdlcleanup(model)
```

Example

```
Without output parameter
dsmpb_mdlcleanup(modelName)

With output parameter
msgContainer = dsmpb_mdlcleanup(modelHandle);
```

Input parameters

The following input parameters are available:

Parameter	Description
`model`	Model identifier.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks

The model cannot contain any model port block.

dsmpb_migrate

dsmpb_migrate

Purpose	Migrates a Simulink model or library to the current MIPS version.
Description	This function enables migrating Simulink models and libraries to the latest MIPS version. When the model is a library, the function unlocks it to be able to apply changes. Likewise, read-only subsystems are made read-write when they contain model port blocks. The function does not save the migrated model to file.

Syntax

```
msgContainer = dsmpb_migrate(model)
```

Example -

Input parameters

The following input parameters are available:

Parameter	Description
`model`	Simulink model or library that you want to migrate or path to Simulink model file.

Property value pairs -

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks -

dsmpb_pref

Where to go from here

Information in this section

dsmpb_pref

Purpose

Manages MIPS preferences.

Syntax overview

The following syntaxes are available:

```
preferences = dsmpb_pref('Get')
```

This command gets the values of all preferences.

```
[preferenceValue, msgContainer] = dsmpb_pref('Get', preferenceName)
```

This command gets the value of a specified preference.

```
msgContainer = dsmpb_pref('Set', preferenceName, preferenceValue)
```

This command sets the value of a specified preference.

```
msgContainer = dsmpb_pref('Save')
```

This command saves the preferences to file. The current values are then available in the next MATLAB session.

Example

-

Remarks

-

dsmpb_pref('Get')

Purpose

Manages MIPS preferences.

Syntax

```
preferences = dsmpb_pref('Get')
```

Description

This command gets the values of all preferences.

Input parameters

-

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`preferences`	Structure with all preferences.

dsmpb_pref('Get', preferenceName)

Purpose

Manages MIPS preferences.

Syntax

```
[preferenceValue, msgContainer] = dsmpb_pref('Get', preferenceName)
```

Description

This command gets the value of a specified preference.

Input parameters

The following input parameters are available:

Parameter	Description
`preferenceName`	Name of preference that must be set or retrieved. MIPS supports the following preferences:

Parameter	Description
	▪ ModelSaveMode - Whether and how modified models must be saved when you close them in ConfigurationDesk. Possible values: 'SaveAndCloseWithoutConfirmation', 'SaveWithoutConfirmation', 'SaveWithConfirmation' and 'DoNotSave'. Default: 'SaveWithConfirmation'. ▪ AskBeforeDelete - Whether to ask you to confirm the deletion of a block from ConfigurationDesk. Possible values: true or false. Default: true. ▪ EnableEditIDs - Whether to enable assigning new block and signal IDs in model port block GUIs. Possible values: true or false. Default: false. ▪ EnableModelPortBlockConnections - Whether to enable connecting two data port blocks. Possible values: true or false. Default: false. ▪ ModelPortBlockConnectionUseCase - Model port block connection use case. If set to FPGA, block dialogs enable to edit goto Tags according to FPGA patterns. Default: empty string. ▪ CreateBusObjectsDuringPropagation - When ConfigurationDesk passes data to a model or creates a new model (Propagate or Create New Simulink Model Interface commands), MIPS creates Simulink.Bus objects for structured data ports. This enables to have more than 3000 leaf signals in a bus (current limit for signals per block) and enhances code generation performance. However, the user is responsible to save the generated Simulink.Bus objects (for example, to a MAT file). Default: false.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`preferenceValue`	Value of specified preference.
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

dsmpb_pref('Save')

Purpose

Manages MIPS preferences.

Syntax

```
msgContainer = dsmpb_pref('Save')
```

Description

This command saves the preferences to file. The current values are then available in the next MATLAB session.

Input parameters

-

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

dsmpb_pref('Set', preferenceName, preferenceValue)

Purpose

Manages MIPS preferences.

Syntax

```
msgContainer = dsmpb_pref('Set', preferenceName, preferenceValue)
```

Description

This command sets the value of a specified preference.

Input parameters

The following input parameters are available:

Parameter	Description
`preferenceName`	Name of preference that must be set or retrieved. MIPS supports the following preferences: ▪ ModelSaveMode - Whether and how modified models must be saved when you close them in ConfigurationDesk. Possible values: 'SaveAndCloseWithoutConfirmation',

Parameter	Description
	'SaveWithoutConfirmation', 'SaveWithConfirmation' and 'DoNotSave'. Default: 'SaveWithConfirmation'. ▪ AskBeforeDelete - Whether to ask you to confirm the deletion of a block from ConfigurationDesk. Possible values: true or false. Default: true. ▪ EnableEditIDs - Whether to enable assigning new block and signal IDs in model port block GUIs. Possible values: true or false. Default: false. ▪ EnableModelPortBlockConnections - Whether to enable connecting two data port blocks. Possible values: true or false. Default: false. ▪ ModelPortBlockConnectionUseCase - Model port block connection use case. If set to FPGA, block dialogs enable to edit goto Tags according to FPGA patterns. Default: empty string. ▪ CreateBusObjectsDuringPropagation - When ConfigurationDesk passes data to a model or creates a new model (Propagate or Create New Simulink Model Interface commands), MIPS creates Simulink.Bus objects for structured data ports. This enables to have more than 3000 leaf signals in a bus (current limit for signals per block) and enhances code generation performance. However, the user is responsible to save the generated Simulink.Bus objects (for example, to a MAT file). Default: false.
`preferenceValue`	Value of preference.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

dsmpb_rmsignal

dsmpb_rmsignal

Purpose

Removes a signal or port.

Description

This function removes a signal or port from a Data Inport or Data Outport block.

Syntax

```
msgContainer = dsmpb_rmsignal(block, signalPath)
```

Example

```
Removes a port:
msgContainer = dsmpb_rmsignal(gcb, 'data2');

Removes a signal 'a' from bus 'data2':
msgContainer = dsmpb_rmsignal(gcb, {'data2','a'})
msgContainer = dsmpb_rmsignal(gcb, 'data2.a')
```

Input parameters

The following input parameters are available:

Parameter	Description
`block`	Model port block.
`signalPath`	Signal to delete: ▪ Cell array with strings including signal names. ▪ String with dot notation. However, this notation might have problems with dots in signal names.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks	You cannot delete the last element of a bus. You must delete the bus instead.

dsmpb_set

dsmpb_set

Purpose

Sets properties in model port blocks.

Description

This function assigns the value of a specified model port block property.

Syntax

```
msgContainer = dsmpb_set(block, [signalPath], propertyName, propertyValue)
```

Example

```
Sets the block ID:
msgContainer = dsmpb_set(block, 'BlockId', 'SampleBlockID')

Sets the ID of port 'data1':
msgContainer = dsmpb_set(block, 'data1', 'SignalId', 'SampleSignalID')

Sets the ID of signal 'signal1' from the bus 'bus1':
msgContainer = dsmpb_set(block, {'bus1','signal1'}, 'signalId', 'MySignalID')

Sets the description of signal 'signal1' from the bus 'bus1' using dot notation to indicate the signal path:
msgContainer = dsmpb_set(block, 'bus1.signal1', 'Description', 'Description of signal1')
```

Input parameters

The following input parameters are available:

Parameter	Description
`block`	Model port block.
`[signalPath]`	Optional argument for signal specific properties. Signal path to get property from: ▪ Cell array with strings including signal names. ▪ Example: {'Bus1','signalA'} ▪ String with dot notation. This notation however might have problems with dots in signal names. ▪ Example 'Bus1.signalA'.
`propertyName`	Property of model port blocks. Case-insensitive character vector. Data Inport and Data Outport blocks Block properties: ▪ SampleTime - MATLAB expression that defines the sample time. ▪ Description - Block description.

Parameter	Description
	▪ BlockID - Block identifier. ▪ DisplayIDs - 0 or 1: Specifies whether IDs to display the model. Signal properties: ▪ Description - Signal description. ▪ Name - Signal name. ▪ Unit - Signal unit. ▪ DataType - Signal data type. ▪ Width - MATLAB expression that defines the signal width. ▪ SignalID - Signal identifier. ▪ Position - Position (one based index) of signal in its bus signal parent element list. Only for Data Inport blocks: ▪ InitialValue - MATLAB expression that defines the initial value. ▪ NonVirtualBus - 0 or 1: Specifies whether a signal is a non-virtual bus. Ignored when signal is not a typed bus, e.g., the data type is not defined by a Simulink.Bus object. ▪ VariableSize - 0 or 1: Specifies whether a signal is variable-size. Ignored when signal is a bus. - Only for data port blocks: ▪ GotoTag - Goto tag of the block port signal. Runnable Function blocks properties: ▪ SampleTime - MATLAB expression that defines the sample time. ▪ Description - Block description. ▪ BlockID - Block identifier. ▪ SignalID - Runnable function identifier. ▪ Name - Runnable function name. ▪ TaskPriority - MATLAB expression that defines the runnable function priority. ▪ Goto tag - Goto tag of the block.
`propertyValue`	Value of the specified property.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. If this function is called without output parameters, messages are displayed in the MATLAB Command Window.

Remarks

To create a bus from a leaf signal (Bus: <untyped>), set the signal DataType to an empty string.

dsmpb_updatemodelportblock

Where to go from here

Information in this section

dsmpb_updatemodelportblock

Purpose

Updates the signal configuration of data port blocks.

Syntax overview

The following syntaxes are available:

```
msgContainer = dsmpb_updatemodelportblock('UpdateFromBusObjects', dataPortBlock, busObjectNames)
```

This command updates the signal configuration of a data port block to match specified Simulink.Bus objects.

```
[msgContainer, hBlocks] = dsmpb_updatemodelportblock('UpdateFromInputs', dataOutportBlocks)
```

This command updates the signal configuration of Data Outport blocks to match the input signals.
This command requires that all Data Outport blocks reside in the same model and that the model can be initialized.
Signal configurations associated with an unconnected block port remain unmodified.

Example

-

Remarks

-

dsmpb_updatemodelportblock('UpdateFromBusObjects', dataPortBlock, busObjectNames)

Purpose

Updates the signal configuration of data port blocks.

Syntax

```
msgContainer = dsmpb_updatemodelportblock('UpdateFromBusObjects', dataPortBlock, busObjectNames)
```

Description

This command updates the signal configuration of a data port block to match specified Simulink.Bus objects.

Input parameters

The following input parameters are available:

Parameter	Description
`dataPortBlock`	Simulink identifier of a data port block.
`busObjectNames`	Names of Simulink.Bus objects in the base workspace or the model's Data Dictionary.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.

dsmpb_updatemodelportblock('UpdateFromInputs', dataOutportBlocks)

Purpose

Updates the signal configuration of data port blocks.

Syntax

```
[msgContainer, hBlocks] = dsmpb_updatemodelportblock('UpdateFromInputs', dataOutportBlocks)
```

Description

This command updates the signal configuration of Data Outport blocks to match the input signals.

This command requires that all Data Outport blocks reside in the same model and that the model can be initialized.

Signal configurations associated with an unconnected block port remain unmodified.

Input parameters

The following input parameters are available:

Parameter	Description
`dataOutportBlocks`	Simulink identifiers of Data Outport blocks.

Property value pairs

-

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages. When this function is called without output parameters, messages are displayed in a message box.
`hBlocks`	Handles of updated Data Outport blocks.

dsmsb_demo

dsmsb_demo

Purpose

Opens Model Separation demo model.

Description

-

Syntax

```
dsmsb_demo(modelName)
```

Example

-

Input parameters

The following input parameters are available:

Parameter	Description
`modelName`	Name of the Model Separation demo model.

Property value pairs

-

Output parameters

-

Remarks

-

dsmsb_separate

dsmsb_separate

Purpose	Runs model separation.
Description	This function copies specified subsystems that reside at the root level of a source model to one or more new Simulink models that are created for this purpose. In the target models, the input and output ports are connected to Data Inport and Data Outport blocks whose parameters are set to match data type, dimension, and - for buses - structure of the input/output signals. Additionally, if more than one target model was generated, Model Separation generates a Model Communication Description (MCD) file that describes the signals and connections between the target models.

Syntax

```
msgContainer = dsmsb_separate(varargin)
```

Example

```
% Example 1: Separate model, use model separation data from model
msgContainer = dsmsb_separate('model', 'myModel');
msgContainer.Submit;

% Example 2: Separate model, specify model separation data explicitly
mdlData(1).Name = 'model1';
mdlData(1).AssignedSubsystems = {'Subsys1', 'Subsys2'};
mdlData(1).ModelFolder = '.\FolderForModel1';

mdlData(2).Name = 'model2';
mdlData(2).AssignedSubsystems = 'Subsys3';
mdlData(2).ModelFolder = '.\FolderForModel2';

msgContainer = dsmsb_separate('model', 'myModel', 'mdlData', mdlData);
if ~msgContainer.AnyError
    % success
else
    % failure
end
```

Input parameters	-

Property value pairs

Additional function parameters are expected as propertyName/propertyValue pairs, refer to the following table:

Parameter	Description
`'System'`	The Simulink model. The function aborts if this parameter does not specify a Simulink model. If passed in as the only parameter, Model Separation data is read from the model.
`'MdlData'`	Model Separation data struct array with fields: ▪ .Name Name of model to be created ▪ .ModelFolder Folder to which you save the model; if empty, the model is saved to the current working directory (optional, default = current working directory) ▪ .AssignedSubsystems Cell array with names of top level subsystems to be copied to the model ▪ .Id ID (nonemtpy string) from which to generate block and signal IDs (optional, default = generated GUID)
`'CompileModel'`	Compile source model to retrieve signal attributes. If this property is set to false, signal attributes are read from the subsystems' inport and outport blocks. This requires that dimensions and data types are set explicitly, e.g., not inherited.
`'CreateInterfaceForGotoFroms'`	Insert Goto/From blocks at the root level of the target models to match Goto/From blocks in the subsystems that are separated into different models. The GoTo/From blocks are connected to data port blocks with matching signal configuration. Default value: true. If set to false, you might not be able to initialize target models.

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	A container (dsmpb.MsgContainer class) that holds notes, warnings, and error messages. To submit the messages to the MATLAB Command Window, use the object's Submit method. To check whether there were any errors, retrieve the object's AnyError property. If dsmsb_separate is called without output argument, a modal message box notifies you about success or failure.

Remarks

-

dsrt_addfiles

dsrt_addfiles

Purpose

Adds files to a code generation result.

Description

This function adds specified files to a DSRT code generation result. When a Simulink Implementation Container (SIC) is generated, the specified files are copied to it. Otherwise, they are referenced in a generated makefile by means of a specified path. Use this function to add libraries, object files, source files, header, and other files to a DSRT code generation result.

This function can be called only when code generation is in progress, i.e., in a rtwmakecfg function, or a DSRT custom build hook. Calling the function outside a code generation context results in a warning, and the files are not added to the code generation result.

If a specified file does not exist, an error message is displayed, and code generation fails.

Syntax

```
dsrt_addfiles(files, varargin)
```

Example

```
% Adds the myFile.c source file to the SCALEXIO RT Linux target platform.
dsrt_addfiles('myFile.c' 'Platform', 'SCALEXIO_LNX', 'FileType', 'SrcFile');

% Adds the platform-independent myfile.txt and myDoc.doc files to the build result.
dsrt_addfiles({'myFile.txt', 'myDoc.doc'});
```

Input parameters

The following input parameters are available:

Parameter	Description
`files`	Paths of files to add. To specify more than one file, use a cell array. Files that are specified more than once are considered only once.

Property value pairs

Additional function parameters are expected as propertyName/propertyValue pairs, refer to the following table:

Parameter	Description
`'Platform'`	Platform for which you want to use the files: AUTERA_LINUX64_GCC AUTERA AutoBox DS1403 MicroAutoBox III (with ConfigurationDesk) SCALEXIO SCALEXIO QNX system (with ConfigurationDesk) SCALEXIO_LNX SCALEXIO RT Linux system (with ConfigurationDesk) VEOS_WIN32_MSVC VEOS on MS Windows 32-bit with MSVC compiler VEOS_WIN32_GCC VEOS on MS Windows 32-bit with GCC compiler VEOS_WIN64_MSVC VEOS on MS Windows 64-bit with MSVC compiler VEOS_WIN64_GCC VEOS on MS Windows 64-bit with GCC compiler VEOS_LINUX64_GCC VEOS on Linux 64-bit with GCC compiler If no platform is specified, the Model Interface Package for Simulink assumes that the file is platform-independent. However, for binary files, you must specify a platform.
`'Filetype'`	Type of the file you want to add: Binary Libraries, object files, precompiled files SrcFile C or C++ source code files HeaderFile C or C++ source code header files DynamicLinkLibrary DLL and shared object files Files specified with SrcFile will be compiled and linked. Binary files will be linked. The path to header files will be added to the include path during builds. Files without a file type will be copied to the SIC file but not processed during builds.
`'Destination'`	Destination folder in the SIC. Relative to the SIC root or, for files outside the code generation directory, the SIC's otherFiles folder. Supported only if you specify neither a file type nor a platform. Files for which a destination was specified are not included in the generated makefile or in the model description file.
`'Model'`	Name of the Simulink model to which you want to add files. If not specified, the files are added to the model for which code is being generated. Use this option when referenced models are involved to make sure that the files are associated with the intended model.

Output parameters

-

Remarks

-

dsrt_build

dsrt_build

Purpose

dSPACE real-time target (DSRT) code generation API

Description

This function triggers DSRT code generation on a specified Simulink system.

When triggered without output arguments, messages are printed to the MATLAB Command Window. Otherwise, they are returned to the caller in a dsmpb.MsgContainer object.

Syntax

```
msgContainer = dsrt_build(varargin)
```

Example

```
% Generate code and SIC
msgContainer = dsrt_built('system', 'myModel');
if msgContainer.AnyError
   % code generation could not be completed
else
   % code generation completed
end
```

Input parameters

-

Property value pairs

Additional function parameters are expected as propertyName/propertyValue pairs, refer to the following table:

Parameter	Description
`'System'`	Simulink system for which code must be generated
`'GenerateSIC'`	Specifies whether a Simulink Implementation Container (SIC) must be generated (default = true)
`'ProtectModel'`	Generates a protected model. If the GenerateSIC property is set to 'true', ProtectModel cannot be set to 'true'. (default = false)

Parameter	Description
`'ObfuscateCode'`	Specifies whether to generate obfuscated C code. (default = false)
`'CodeOutputPath'`	Directory to which you want to generate the code. If it does not exist, it will be created. (default = current working directory)
`'Force'`	Specifies whether to always generate code, even when the Simulink system was not modified after the last code generation run. (default = false)
`'Target'`	Code generation target: Either 'dsrt' or 'dsrt64'. (default = dsrt)
`'CheckProtectedModelMIPSVersion'`	Check whether protected referenced models were built with MIPS versions prior to 4.1 (Release 2019-A). When you skip this check, code generation might fail with enigmatic Simulink messages. (default = true)

Output parameters

The following output parameters are available:

Parameter	Description
`msgContainer`	dsmpb.MsgContainer object with messages

Remarks

-