

Model Interface Package for Simulink

Reference

For Model Interface Package for Simulink 4.5

Release 2021-A – May 2021

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

This publication contains proprietary information that is protected by copyright. All rights are reserved. The publication may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the publication must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© 2008 - 2021 by:
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

This publication and the contents hereof are subject to change without notice.

AUTERA, ConfigurationDesk, ControlDesk, MicroAutoBox, MicroLabBox, SCALEXIO, SIMPHERA, SYNECT, SystemDesk, TargetLink and VEOS are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

Contents

About This Reference	5
Model Interface Blockset	7
Data Inport Block.....	9
Data Inport Block Overview.....	9
ConfigurationDesk Page (Data Inport Block).....	10
Signal Configuration Page (Data Inport Block).....	12
Block Configuration Page (Data Inport Block).....	16
Block Connections Page (Data Inport Block).....	17
Data Outport Block.....	18
Data Outport Block Overview.....	18
ConfigurationDesk Page (Data Outport Block).....	19
Signal Configuration Page (Data Outport Block).....	21
Block Configuration Page (Data Outport Block).....	24
Block Connections Page (Data Outport Block).....	25
Hardware-Triggered Runnable Function Block.....	26
Hardware-Triggered Runnable Function Block Overview.....	26
ConfigurationDesk Page (Hardware-Triggered Runnable Function Block).....	27
Runnable Function Page (Hardware-Triggered Runnable Function Block).....	29
Block Configuration Page (Hardware-Triggered Runnable Function Block).....	31
Block Connections Page (Hardware-Triggered Runnable Function Block).....	32
Software-Triggered Runnable Function Block.....	33
Software-Triggered Runnable Function Block Overview.....	33
Runnable Function Page (Software-Triggered Runnable Function Block).....	34
Block Configuration Page (Software-Triggered Runnable Function Block).....	35
Model Separation Setup Block.....	37
Model Separation Setup Block Overview.....	37
Model Separation Setup.....	39
Model Separation Hook Functions.....	43

Menu Commands of Model Port Blocks.....	47
Check Model for DSRT Code Generation.....	48
Create Data Inport Block from Bus Creator Block.....	48
Create Data Outport Block from Bus Creator Block.....	52
Create Inverse Model Port Block.....	54
Create Model Port Block Periphery.....	56
Open Model Interface Blockset.....	57
Open Model Separation Setup.....	58
Paste and Keep IDs.....	58
Select All Outport Blocks in This Model.....	59
Update Selected Outport Blocks from Input Signals.....	60

Menu Commands for the Remote Access of ConfigurationDesk 63

Add Model to ConfigurationDesk Project: <ConfigurationDesk Project Name>.....	63
Analyze Model in ConfigurationDesk (Including Task Information).....	64
Analyze Model in ConfigurationDesk (Model Interface Only).....	66
Create ConfigurationDesk Project from Model.....	67
Delete Selection and Connected Blocks in ConfigurationDesk.....	68
Open a ConfigurationDesk Project.....	69
Save ConfigurationDesk Project.....	69
Show Connected Block in ConfigurationDesk.....	70
Show in ConfigurationDesk.....	71
Start ConfigurationDesk Build.....	71

M Files for Customizing the Simulation Environment 73

startup.m.....	73
dsstartup.m.....	74
dspoststartup.m.....	75
dsfinish.m.....	76
dsmpb_pref.....	76

Model Interface Package for Simulink Glossary 79

Index 85






About This Reference

Content

The Model Interface Package for Simulink consists of specific components, such as block libraries, that let you specify the interface of Simulink behavior models. You can use these Simulink behavior models in ConfigurationDesk or VEOS Player. This reference provides information on the block libraries and menu commands of the Model Interface Package for Simulink. It also contains information on files that let you customize your simulation environment.

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.

Model Interface Blockset

Intended use

You can use the blocks of the [Model Interface Blockset](#)  to implement communication between a Simulink behavior model and ConfigurationDesk or VEOS Player.

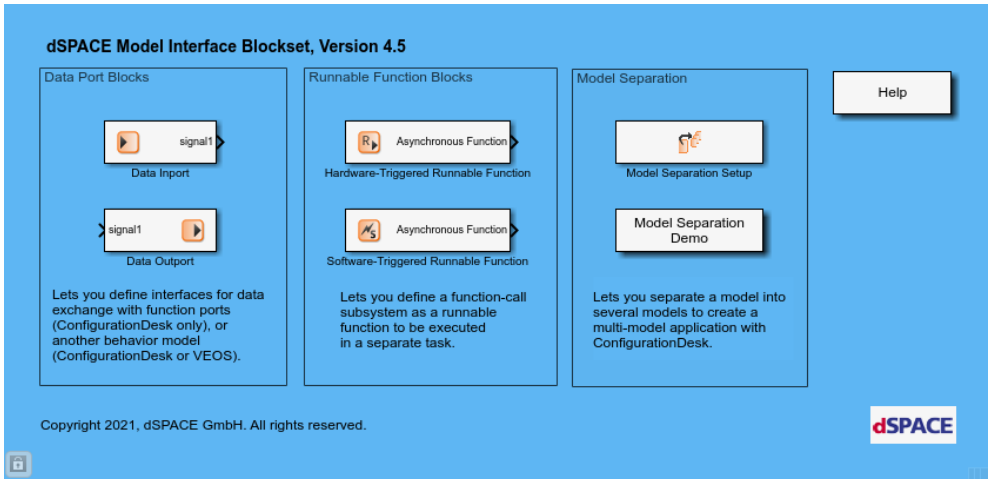
The interface of a Simulink behavior model can be used independently from the target platform:

- In ConfigurationDesk, the model port blocks of the Simulink model correspond to the model port blocks of a ConfigurationDesk application. Refer to [Introduction to the Modeling Concept \(Model Interface Package for Simulink - Modeling Guide\)](#) .
- In VEOS Player, the model port blocks of the Simulink model correspond to the VPU ports. Refer to [Elements Generated for Simulink Implementation Containers \(VEOS Manual\)](#) .

Access

The Model Interface Blockset is installed together with the Model Interface Package for Simulink. To open the library, type `mips` in the MATLAB Command Window, select the **Open Model Interface Blockset** command from the **Model Port Blocks** menu in the Simulink model, or select **dSPACE Model Interface Blockset** from the Simulink Library Browser.

Blocks of the Model Interface Blockset The following illustration shows the blocks of the Model Interface Blockset:



Model Port Blocks menu commands The Model Interface Package for Simulink provides commands that are accessible via the Model Port Blocks menu of Simulink models. Refer to [Menu Commands of Model Port Blocks](#) on page 47.

Where to go from here **Information in this section**

Data Input Block.....	9
Data Output Block.....	18
Hardware-Triggered Runnable Function Block.....	26
Software-Triggered Runnable Function Block.....	33
Model Separation Setup Block.....	37
Menu Commands of Model Port Blocks.....	47

Data Inport Block

Purpose Data Inport blocks receive data from the ports of another behavior model (ConfigurationDesk and VEOS Player) or from a function port (ConfigurationDesk only).

Where to go from here

Information in this section

Data Inport Block Overview.....	9
ConfigurationDesk Page (Data Inport Block).....	10
Signal Configuration Page (Data Inport Block).....	12
Block Configuration Page (Data Inport Block).....	16
Block Connections Page (Data Inport Block).....	17

Data Inport Block Overview

Block illustration



Purpose To prepare the behavior model for receiving data from other elements in [ConfigurationDesk](#) or in [VEOS Player](#).

Description

You can add a [behavior model](#) that contains [Data Inport](#) blocks to a ConfigurationDesk application or import it to VEOS Player. In a ConfigurationDesk application, the Data Inport blocks of a behavior model are represented by model port blocks. In VEOS Player, the Data Inport blocks are represented by VPU ports.

Depending on the tool you use, these blocks receive data from the following elements:

- In ConfigurationDesk: Other model port blocks or function blocks.
- In VEOS Player: VPU ports.

Data Inport blocks can have one or more data inports. Data Inport blocks are configurable (e.g., with respect to the sample time) and so are their data ports (e.g., with respect to the data type and width).

Copying model port blocks You can use Simulink's standard copy & paste operations to create a new model port block from an existing one. If you want to work with model port blocks with the same identification, you must use Simulink's standard Copy command together with the Paste and Keep IDs command. For more information on block identities, refer to [Basics on Model Port Block IDs and Signal IDs \(Model Interface Package for Simulink - Modeling Guide !\[\]\(529949c2c3dadbaa4e538e8c643454bc_img.jpg\)](#)).

Related topics

Basics

[Handling the Model Interface \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(0f848bbd71cef6b345273b16f905912a_img.jpg\)](#))

References

[Paste and Keep IDs..... 58](#)

ConfigurationDesk Page (Data Inport Block)

Purpose

To view information about the [Data Inport !\[\]\(6059a5aa8b4ca7bb793408023d6c6e42_img.jpg\)](#) block in the related ConfigurationDesk project and application. You can view information on the function blocks to which the model port block that represents this **Data Inport** block in ConfigurationDesk is mapped. Additionally, you can display the model port block and the function blocks to which it is mapped in ConfigurationDesk itself.

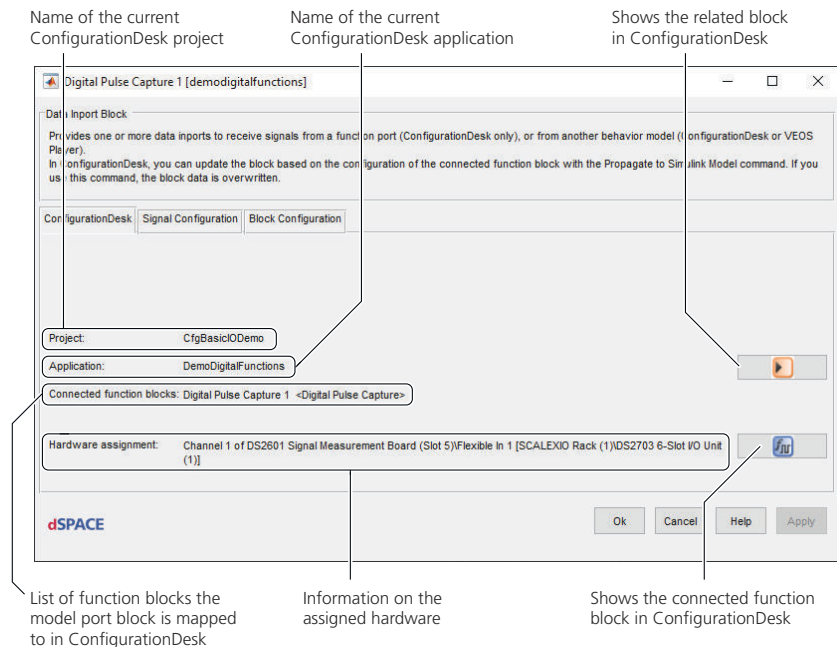
Note

This information is displayed only if the following preconditions are fulfilled:

- You must have added the Simulink model to a ConfigurationDesk project and application.
- You must have performed a model analysis to make the selected Data Inport block available in ConfigurationDesk.
- The related ConfigurationDesk project is open.
- The ConfigurationDesk application to which the Simulink model was added is active.

ConfigurationDesk page overview

Shows information about the selected model port block in the related ConfigurationDesk project and application. Refer to the following illustration:



Project Displays the name of the related ConfigurationDesk project.

Application Displays the name of the related ConfigurationDesk application.

Connected function blocks In ConfigurationDesk, the block is represented by a model port block. Connected function blocks displays the function blocks to which the model port block is mapped.

Hardware assignment Shows information on the hardware that is assigned to the function blocks mapped to the model port block that represents the selected block in ConfigurationDesk.



Shows the model port block that represents the selected block in ConfigurationDesk. The model port block is selected in ConfigurationDesk's Model-Function Mapping Browser.

Tip

In ConfigurationDesk, you can map a function block to the the model port block. Alternatively, you can configure the function block to which the model port block is mapped. You can then update the values on the Port configuration page of this block based on the configuration of the connected function block. This can be done via the Propagate to Simulink Model command in ConfigurationDesk.



Shows the function block mapped to the model port block that represents the selected block in ConfigurationDesk. The function block is highlighted in ConfigurationDesk's Model-Function Mapping Browser.

Tip

In ConfigurationDesk, you can configure the function block the model port block is mapped to. You can then update the values on the Signal configuration page of this block based on the configuration of the connected function block. This can be done via the Propagate to Simulink Model command in ConfigurationDesk.

Signal Configuration Page (Data Inport Block)



Purpose

To view and change the available ports and their configurations.

**Configuration parameters
and ports in
ConfigurationDesk and VEOS
Player**

If you use ConfigurationDesk, the following applies:

The following methods for configuring the values on the Signal configuration page are available:

- You can specify the values directly in the [Data Inport](#)  block of the Simulink model. If you do so, you must perform a model analysis to make the values available in ConfigurationDesk. Refer to [Analyzing Simulink Behavior Models \(ConfigurationDesk Real-Time Implementation Guide\)](#) .

Note

The configuration parameters are displayed as read-only in ConfigurationDesk.

- ConfigurationDesk lets you propagate the configuration of the function block to which the related model port block is mapped to the Data Inport block in Simulink. If you do so, the values on the Signal configuration page are updated with values that are suitable for the configuration of the mapped function block.

If you use VEOS Player, the following applies:

You must specify the values directly in the Data Inport block in the Simulink model. In VEOS Player, the ports of the Data Inport blocks are displayed as VPU ports. Refer to [Elements Generated for Simulink Implementation Containers \(VEOS Manual\)](#). The ports' configuration parameters are not displayed and cannot be configured in VEOS Player.

Signals



Lets you add a new signal to the root level or to a selected signal of the Ports tree. If you add a new signal to a signal in the Ports tree, the data type of that signal changes to Bus: <untyped> automatically, if required.



Lets you delete a signal.



Lets you cut a signal and copy it to the Clipboard of the dialog.



Lets you copy a signal to the Clipboard of the dialog.



Lets you paste a signal from the Clipboard to the selected signal. If you selected the Ports (root) node, you create a new block port. If you selected a bus signal, you create a new bus element.



Lets you move the selected signal up one position.



Lets you move the selected signal down one position.

Signals Displays the block's signal configuration.

Signal properties

The following signal properties are available:

Name Lets you enter a name for the signal selected in the signal tree. The name of the root signal (port) is used as a variable name in the generated variable description file after the real-time application is built.

The name must fulfill the following conditions:

- The name must not begin or end with /.
- The names of signals on the same structure level must be unique. They are case-sensitive.

Description Lets you enter a description of the signal. This is only for the purpose of documentation.

Unit Lets you specify a physical unit for the signal. This is only for the purpose of documentation.

Type Lets you specify the Simulink data type of the signal used in the model. For the model port block that represents this model port block in ConfigurationDesk, the following applies: This Simulink data type will be converted to a data type used in ConfigurationDesk when you perform a model analysis in ConfigurationDesk or when you import a Simulink implementation container in ConfigurationDesk.

The following table shows Simulink data types and the data types they are converted to in ConfigurationDesk:

Simulink	ConfigurationDesk
single	Float32
double	Float64
int8	Int8
int16	Int16
int32	Int32
int64	Int64
uint8	UInt8
uint16	UInt16
uint32	UInt32
uint64	UInt64
boolean	Bool

To make a signal an untyped bus, select Bus: <untyped>. This converts a non-bus signal into a bus signal with one bus element. Additionally, you can select a Simulink.Bus object that is defined in the base workspace or the model's Data Dictionary. When you click the Type list, it shows all defined Simulink.Bus objects. When you select a Simulink.Bus object for a signal, its configuration (bus elements) is modified to match the object. Elements of a signal whose data type is defined by a Simulink.Bus object are read-only.



(Only enabled for untyped bus signals) Lets you assign a Simulink.Bus object that matches the signal configuration of the selected signal. If no matching Simulink.Bus object exists in the base workspace or the model's Data Dictionary, the Model Interface Package for Simulink creates this object and assigns it to the selected signal. A prerequisite for this is that all signal names are valid C identifiers. If you assign a Simulink.Bus object with a Simulink.BusElement whose DimensionMode is set to Variable at a data port, the specified signal is also of variable-size.



Lets you update the Simulink.Bus objects in the dialog with the Simulink.Bus objects in the MATLAB Base Workspace and in the model's data dictionary. The signal configuration displayed in the **Signal Configuration** page is also updated.

Output as nonvirtual bus Lets you specify that the signal must be a nonvirtual signal (nonvirtual bus). This applies only to signals whose data type is specified by a Simulink.Bus object.

Width Lets you specify the width (vector size) of the signal passed through the data port. The width value must be of integer type and in the range $1 \dots 2^{31}-1$ (1 means scalar). For variable-size signals, you can specify the maximum number of elements via this edit field.

Tip

- You can also reference a workspace variable in the Simulink model, including associated calculation rules.
- In ConfigurationDesk, the following applies:
The width of the data port does not necessarily have to match the width of the function port you want to map. If the port widths differ, the cut set of vector elements is used. The other elements are neglected, and ConfigurationDesk issues a warning for the signal chain.

Note

Only scalars and vectors are supported. Matrix and multidimensional signals are not supported.

Variable-size signal Lets you specify that the selected signal is a variable-size signal. The number of elements of variable-size signals varies at run-time. You can specify the maximum number of elements via the **Width** edit field.

Initial value Lets you specify a default value that is used in the real-time application (ConfigurationDesk) or offline simulation application (VEOS Player) until the mapped function is executed for the first time or when the behavior model is running in a Simulink simulation mode.

To specify a vector, you can enter its elements as follows:


- If all the elements are the same, you can enter just one value, which will be applied to all elements, e.g., `0`.
- If the elements are different, enter the vector. Its width must equal the **Width** parameter, e.g., `[0, 1]`.
- If the signal data type is an integer type, its initial value must not exceed the type-specific range.
- The initial value must not have infinite or NaN elements.
- If the signal data type is an integer type, non-integer initial values result in a warning message during the build process.

Tip

You can also reference a workspace variable in the Simulink model, including associated calculation rules.

Signal ID Displays the identity (ID) of the signal selected in the signal tree.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmptb_pref('Set','EnableEditIds', true);
```



Block Configuration Page (Data Inport Block)

Purpose

To view and change the block configuration.

Configuration

In ConfigurationDesk, the following applies:


ConfigurationDesk displays the configuration parameters described below as read-only. If a parameter was changed in the [Data Inport](#)  block, the new value is displayed in ConfigurationDesk after the model was analyzed. Refer to [Analyzing Simulink Behavior Models \(ConfigurationDesk Real-Time Implementation Guide\)](#) .

In VEOS Player, the following applies:

The block configuration parameters are not displayed and cannot be configured in VEOS Player.

Block ID Displays the identity (ID) of the model port block.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmptb_pref('Set','EnableEditIds', true);
```


Display signal and block IDs Lets you display the signal and block IDs in the behavior model.

Note

The maximum number of displayed IDs is 14.

Sample time Lets you enter the sample time of the Data Inport block. The sample time implicitly determines the periodic task that the block belongs to.

Sample Time	Description
-1	Lets you use the sample time inherited from the behavior model.
≥ 0	<p>Lets you specify the sample time in seconds. Use any multiple of the Fixed step size (base sample time) specified for the behavior model.</p> <p>In addition, you can specify a sample time offset. The syntax is as follows:</p> <p>[Sample time, Sample time offset], e.g., [0.01, 0.002]</p> <p>The offset must be an integer multiple of the model's fixed step size (base sample time), and must be smaller than the sample time value.</p>

Tip

You can also reference a workspace variable in the Simulink model, including associated calculation rules.

Description Lets you enter a description of the block. This is only for the purpose of documentation.

Block Connections Page (Data Inport Block)

Purpose

To enable and configure data connections between model port blocks and blocks from the RTI FPGA Programming Blockset during Simulink simulations.

Note

This page is only available if you have enabled it for the use with the RTI FPGA Programming Blockset. Refer to [Processor Interface Blocks \(MicroAutoBox III, SCALEXIO\) \(RTI FPGA Programming Blockset - Processor Interface Reference !\[\]\(95b425611cbd2b8716a140cf67c81822_img.jpg\)](#)).

Data Output Block

Purpose Data Output blocks send data to the ports of another behavior model (ConfigurationDesk and VEOS Player) or to a function port (ConfigurationDesk only).

Where to go from here

Information in this section

Data Output Block Overview.....	18
ConfigurationDesk Page (Data Output Block).....	19
Signal Configuration Page (Data Output Block).....	21
Block Configuration Page (Data Output Block).....	24
Block Connections Page (Data Output Block).....	25

Data Output Block Overview

Block illustration



Purpose To prepare the behavior model for sending data to other elements in [ConfigurationDesk](#) or [VEOS Player](#).

Description

You can add a [behavior model](#) that contains [Data Output](#) blocks to a ConfigurationDesk application or import it to VEOS Player. In a ConfigurationDesk application, the Data Output blocks of a behavior model are represented by model port blocks. In VEOS Player, the Data Output blocks are represented by VPU ports.

Depending on the tool you use, these blocks send data to the following elements:

- In ConfigurationDesk: Other model port blocks or function blocks.
- In VEOS Player: VPU ports.

Data Output blocks can have one or more data outputs. Data Output blocks are configurable (e.g., with respect to the sample time) and so are their data ports (e.g., with respect to the data type and width).

Copying model port blocks You can use Simulink's standard copy & paste operations to create a new model port block from an existing one. If you want to work with model port blocks with the same identification, you must use Simulink's standard Copy command together with the Paste and Keep IDs command. For more information on block identities, refer to [Basics on Model Port Block IDs and Signal IDs \(Model Interface Package for Simulink - Modeling Guide !\[\]\(1d3a1175dd4902218e694b9c098adb83_img.jpg\)\)](#).

Related topics

Basics

[Handling the Model Interface \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(cbe80b694ebd74fcfe136a095b608235_img.jpg\)\)](#)

References

[Paste and Keep IDs..... 58](#)

ConfigurationDesk Page (Data Output Block)

Purpose

To view information about the [Data Output !\[\]\(5361750c22c4e047a52f4eac1ec2d4cc_img.jpg\)](#) block in the related ConfigurationDesk project and application. You can view information on the function blocks to which the model port block that represents this Data Output block in ConfigurationDesk is mapped. Additionally, you can display the model port block and the function blocks to which it is mapped in ConfigurationDesk itself.

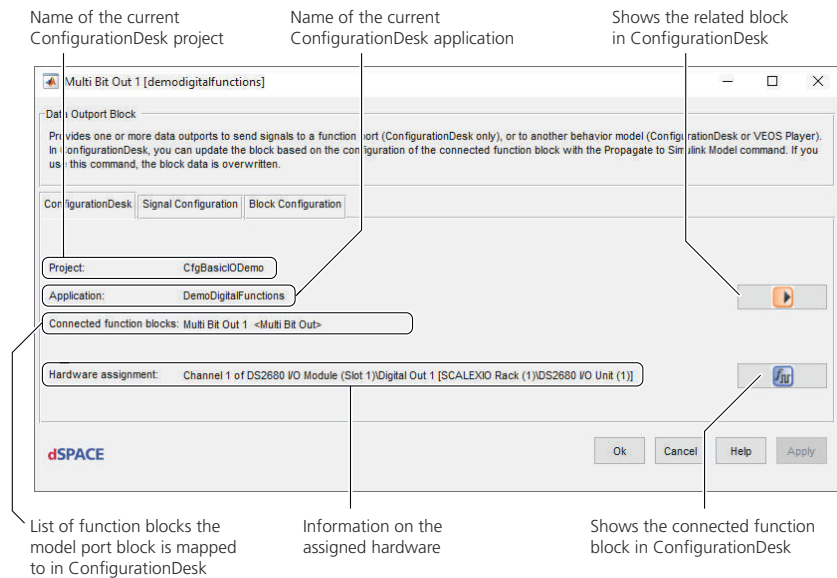
Note

This information is displayed only if the following preconditions are fulfilled:

- You must have added the Simulink model to a ConfigurationDesk project and application.
- You must have performed a model analysis to make the selected Data Output block available in ConfigurationDesk.
- The related ConfigurationDesk project is open.
- The ConfigurationDesk application to which the Simulink model was added is active.

ConfigurationDesk page overview

Shows information about the selected model port block in the related ConfigurationDesk project and application. Refer to the following illustration:



Project Displays the name of the related ConfigurationDesk project.

Application Displays the name of the related ConfigurationDesk application.

Connected function blocks In ConfigurationDesk, the block is represented by a model port block. **Connected function blocks** displays the function blocks to which the model port block is mapped.

Hardware assignment Shows information on the hardware that is assigned to the function blocks mapped to the model port block that represents the selected block in ConfigurationDesk.



Shows the model port block that represents the selected block in ConfigurationDesk. The model port block is selected in ConfigurationDesk's Model-Function Mapping Browser.

Tip

In ConfigurationDesk, you can map a function block to the model port block. Alternatively, you can configure the function block to which the model port block is mapped. You can then update the values on the Port configuration page of this block based on the configuration of the connected function block. You can do this with the Propagate to Simulink Model command in ConfigurationDesk.



Shows the function block mapped to the model port block that represents the selected block in ConfigurationDesk. The function block is highlighted in ConfigurationDesk's Model-Function Mapping Browser.

Tip

In ConfigurationDesk, you can configure the function block the model port block is mapped to. You can then update the values on the Signal configuration page of this block based on the configuration of the connected function block. You can do this with the Propagate to Simulink Model command in ConfigurationDesk.

Signal Configuration Page (Data Outport Block)

Purpose

To view and change the available ports and their configuration.

Configuration parameters and ports in ConfigurationDesk and VEOS Player

If you use ConfigurationDesk, the following applies:

The following methods for configuring the values on the Signal configuration page are available:

- You can specify the values directly in the [Data Outport](#) block in the Simulink model. If you do so, you must perform a model analysis to make the values available in ConfigurationDesk. Refer to [Analyzing Simulink Behavior Models \(ConfigurationDesk Real-Time Implementation Guide\)](#).

Note

- The configuration parameters are displayed in ConfigurationDesk as read-only.
 - The port configuration for structured data ports is read-only.
- ConfigurationDesk lets you propagate the configuration of the function block to which the related model port block is mapped to the Data Outport block in Simulink. If you do so, the values on the Signal configuration page are updated with values that are suitable for the configuration of the mapped function block.

If you use VEOS Player, the following applies:

You must specify the values directly in the Data Outport block in the Simulink model. In VEOS Player, the ports of the Data Outport blocks are displayed as VPU ports. Refer to [Elements Generated for Simulink Implementation Containers \(VEOS Manual\)](#). The ports' configuration parameters are not displayed and cannot be configured in VEOS Player.

Signals



Lets you add a new signal to the root level or to a selected signal of the Ports tree. If you add a new signal to a signal in the Ports tree, the data type of that signal changes to Bus: <untyped> automatically, if required.



Lets you delete a signal.



Lets you cut a signal and copy it to the Clipboard of the dialog.



Lets you copy a signal to the Clipboard of the dialog.



Lets you paste a signal from the Clipboard to the selected signal. If you selected the Ports (root) node, you create a new block port. If you selected a bus signal, you create a new bus element.



Lets you move the selected signal up one position.



Lets you move the selected signal down one position.

Signals Displays the block's signal configuration.

Properties

The following signal properties are available:

Name Lets you enter a name for the signal selected in the signal tree. The name of the root signal (port) is used as a variable name in the generated variable description file after the real-time application is built.

The name must fulfill the following conditions:

- The name must not begin or end with /.
- The names of signals on the same structure level must be unique. They are case-sensitive.

Description Lets you enter a description of the signal. This is only for the purpose of documentation.

Unit Lets you specify a physical unit for the signal. This is only for the purpose of documentation.

Type Lets you specify the Simulink data type of the signal used in the model. For the model port block that represents this model port block in ConfigurationDesk, the following applies: This Simulink data type will be converted to a data type used in ConfigurationDesk when you perform a model analysis in ConfigurationDesk or when you import a Simulink implementation container in ConfigurationDesk.

The following table shows Simulink data types and the data types they are converted to in ConfigurationDesk:

Simulink	ConfigurationDesk
single	Float32
double	Float64

Simulink	ConfigurationDesk
int8	Int8
int16	Int16
int32	Int32
int64	Int64
uint8	UInt8
uint16	UInt16
uint32	UInt32
uint64	UInt64
boolean	Bool

To make a signal an untyped bus, select Bus: <untyped>. This converts a non-bus signal into a bus signal with one bus element. Additionally, you can select a Simulink.Bus object that is defined in the base workspace or the model's Data Dictionary. When you click the Type list, it shows all defined Simulink.Bus objects. When you select a Simulink.Bus object for a signal, its configuration (bus elements) is modified to match the object. Elements of a signal whose data type is defined by a Simulink.Bus object are read-only.



(Only enabled for untyped bus signals) Lets you assign a Simulink.Bus object that matches the signal configuration of the selected signal. If no matching Simulink.Bus object exists in the base workspace or the model's Data Dictionary, the Model Interface Package for Simulink creates this object and assigns it to the selected signal. A prerequisite for this is that all signal names are valid C identifiers. If you assign a Simulink.Bus object with a Simulink.BusElement whose DimensionMode is set to Variable at a data port, the specified signal is also of variable-size.



Lets you update the Simulink.Bus objects in the dialog with the Simulink.Bus objects in the MATLAB Base Workspace and in the model's data dictionary. The signal configuration displayed in the **Signal Configuration** page is also updated.

Width Lets you specify the width (vector size) of the signal passed through the data port. The width value must be of integer type and in the range $1 \dots 2^{31}-1$ (1 means scalar).

Tip


- You can also reference a workspace variable in the Simulink model, including associated calculation rules.
- In ConfigurationDesk, the following applies:
The width of the data port does not necessarily have to match the width of the function port you want to map. If the port widths differ, the cut set of vector elements is used. The other elements are neglected, and ConfigurationDesk issues a warning for the signal chain.

Note

Only scalars and vectors are supported. Matrix and multidimensional signals are not supported.

Signal ID Displays the identity (ID) of the signal selected in the signal tree.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```



Block Configuration Page (Data Output Block)

Purpose

To view and change the block configuration.

Configuration

In ConfigurationDesk, the following applies:


ConfigurationDesk displays the configuration parameters described below as read-only. If a parameter was changed in the [Data Output](#)  block, the new value is displayed in ConfigurationDesk after the model was analyzed. Refer to [Analyzing Simulink Behavior Models \(ConfigurationDesk Real-Time Implementation Guide\)](#) .

In VEOS Player, the following applies:

The block configuration parameters are not displayed and cannot be configured in VEOS Player.

Block ID Displays the identity (ID) of the model port block.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```


Display signal and block IDs Lets you display the signal and block IDs in the behavior model.

Note

The maximum number of displayed IDs is 14.

Sample time Lets you enter the sample time of the Data Output block. The sample time implicitly determines the periodic task that the block belongs to.

Sample Time	Description
-1	Lets you use the sample time inherited from the behavior model.
≥ 0	<p>Lets you specify the sample time in seconds. Use any multiple of the Fixed step size (base sample time) specified for the behavior model.</p> <p>In addition, you can specify a sample time offset. The syntax is as follows:</p> <p>[Sample time, Sample time offset], e.g., [0.01, 0.002]</p> <p>The offset must be an integer multiple of the model's fixed step size (base sample time), and must be smaller than the sample time value.</p>

Tip

You can also reference a workspace variable in the Simulink model, including associated calculation rules.

Description Lets you enter a description of the block. This is only for the purpose of documentation.

Block Connections Page (Data Output Block)

Purpose

To enable and configure data connections between model port blocks and blocks from the RTI FPGA Programming Blockset during Simulink simulations.

Note

This page is only available if you have enabled it for the use with the RTI FPGA Programming Blockset. Refer to [Processor Interface Blocks \(MicroAutoBox III, SCALEXIO\) \(RTI FPGA Programming Blockset - Processor Interface Reference !\[\]\(95b425611cbd2b8716a140cf67c81822_img.jpg\)](#)).

Hardware-Triggered Runnable Function Block

Purpose

Hardware-Triggered Runnable Function blocks let you execute parts of a model asynchronously, for example, triggered by an event of a function block (ConfigurationDesk only). For this purpose, a Hardware-Triggered Runnable Function block exports a function-call subsystem as a runnable function. In ConfigurationDesk, you can then create a task from the runnable function and an asynchronous event.

Where to go from here

Information in this section

Hardware-Triggered Runnable Function Block Overview.....	26
ConfigurationDesk Page (Hardware-Triggered Runnable Function Block).....	27
Runnable Function Page (Hardware-Triggered Runnable Function Block).....	29
Block Configuration Page (Hardware-Triggered Runnable Function Block).....	31
Block Connections Page (Hardware-Triggered Runnable Function Block).....	32

Hardware-Triggered Runnable Function Block Overview

Block illustration



Purpose

To export function-call subsystems as [runnable functions](#) for modeling asynchronous tasks in ConfigurationDesk by using a function block event, for example.

Description

The function-call subsystem is exported as a runnable function. The runnable function can be used in ConfigurationDesk, where it is displayed as follows:

- As a Runnable Function block in the Model Browser and in the Model-Function Mapping Browser.
- As a runnable function in the Task Configuration table view.

You can use these elements to model asynchronous tasks, for example, by connecting them with a function block event.

Note

Hardware-Triggered Runnable Function blocks have the following modeling limitations:

- The `function()` trigger port of the function-call subsystem must be scalar.
- You cannot connect multiple Hardware-Triggered Runnable Function blocks to the same function-call subsystem via a Simulink Mux block.
- You cannot connect a Hardware-Triggered Runnable Function block to multiple function-call subsystems except with a Function-Call Split block.

Related topics

Basics

[Modeling Executable Applications and Tasks \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(e474458956c9a37fbf9586ddb60a7fa1_img.jpg\)\)](#)

References

Block Configuration Page (Hardware-Triggered Runnable Function Block)	31
Runnable Function Page (Hardware-Triggered Runnable Function Block)	29

ConfigurationDesk Page (Hardware-Triggered Runnable Function Block)

Purpose

To view information about the [Hardware-Triggered Runnable Function !\[\]\(0d5ec72f61334709c3fc9450209b754f_img.jpg\)](#) block in the related ConfigurationDesk project and application. You can view information on the function blocks to which the model port block that represents this Hardware-Triggered Runnable Function block in ConfigurationDesk is mapped. Additionally, you can display the model port block and the function blocks to which it is mapped in ConfigurationDesk itself.

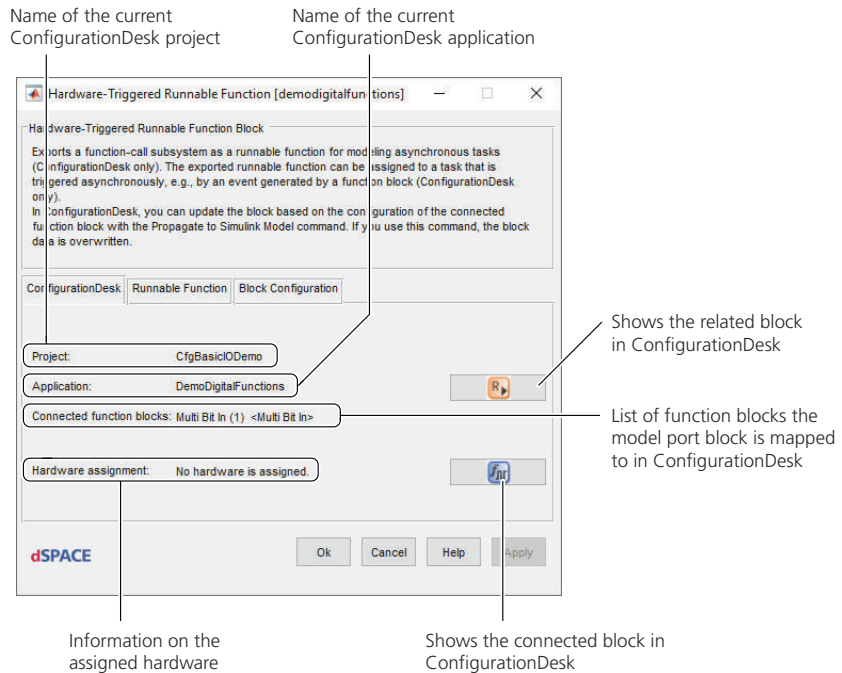
Note

This information is displayed only if the following preconditions are fulfilled:

- You must have added the Simulink model to a ConfigurationDesk project and application.
- You must have performed a model analysis to make the relevant Hardware-Triggered Runnable Function block available in ConfigurationDesk.
- The related ConfigurationDesk project is open.
- The ConfigurationDesk application to which the Simulink model was added is active.

ConfigurationDesk page overview

Shows information about the Hardware-Triggered Runnable Function block in the related ConfigurationDesk project and application. Refer to the following illustration:



Project Displays the name of the related ConfigurationDesk project. This information is available only if the selected Hardware-Triggered Runnable Function block and the current Simulink model are contained in the active ConfigurationDesk application.

Application Displays the name of the related ConfigurationDesk application. This information is available only if the selected Hardware-Triggered Runnable Function block and the current Simulink model are contained in the active ConfigurationDesk application.

Connected function blocks In ConfigurationDesk, the Hardware-Triggered Runnable Function block is represented by a model port block. Connected function blocks displays the function blocks to which the model port block is mapped.

Hardware assignment Shows information on the hardware that is assigned to the function blocks mapped to the model port block that represents the selected Hardware-Triggered Runnable Function block in ConfigurationDesk.



Shows the model port block that represents the selected Hardware-Triggered Runnable Function block in ConfigurationDesk. The model port block is selected in ConfigurationDesk's Model-Function Mapping Browser.



Shows the function block mapped to the model port block that represents the selected Hardware-Triggered Runnable Function block in ConfigurationDesk. The function block is highlighted in ConfigurationDesk's Model-Function Mapping Browser.

Tip

In ConfigurationDesk, you can now map a function block with enabled event generation to the model port block. Alternatively, you can configure the task to which the runnable function is assigned. You can then update the values on the Runnable Function page of this Hardware-Triggered Runnable Function block based on the configuration in ConfigurationDesk. You can do this with the Propagate to Simulink Model command.

Runnable Function Page (Hardware-Triggered Runnable Function Block)

Purpose

To view and change the runnable function configuration.

Note

Depending on the tool you use, the following applies to the configuration parameters of runnable functions:

- ConfigurationDesk: ConfigurationDesk displays the configuration parameters described below as read-only. If one of these parameters is changed in the Simulink model, the new value can be seen in ConfigurationDesk after the model was analyzed. Refer to [Analyzing Simulink Behavior Models \(ConfigurationDesk Real-Time Implementation Guide\)](#).
- VEOS Player: The runnable functions are not executed in VEOS Player (refer to the description below). The configuration parameters are not displayed and cannot be configured in VEOS Player.

Properties

Lets you specify the properties of the runnable function that is exported from the function-call subsystem.

Note

You can update the values on the Hardware-Triggered Runnable function page via the Propagate to Simulink Model command in ConfigurationDesk. For example, if you changed the task priority in ConfigurationDesk, the Propagate to Simulink Model command updates the value of the Required priority on the Runnable function page with the task priority value you specified in ConfigurationDesk.

Name of runnable function Lets you specify a runnable function name.

Note


In ConfigurationDesk, the runnable function name must be unique within the active ConfigurationDesk application, because it serves as an identifier for the modeled asynchronous task. If you map a Runnable Function block to a function block and subsequently change the runnable function name, the assignment to the task is deleted. The exported runnable function is then displayed in the Task Configuration table view as a new runnable function.

The name must fulfill the following conditions:

- The name must be unique within the model.
- The name must not begin or end with /.
- The name is case-sensitive.

Signal ID Displays the signal ID of the Hardware-Triggered Runnable Function block.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```

Required priority Lets you specify a value for the task priority restriction (default value = 30). The default value 30 ensures that the task that executes the runnable function has a higher priority than the periodic tasks in the model.

For runnable functions in ConfigurationDesk, the following applies:

When configuring the tasks to which the runnable functions are assigned in ConfigurationDesk, you must make sure that the relationship between the task priorities matches the relationship between the predefined priorities, even if the absolute priority values differ from the predefined values. For details, refer to [Basics on Modeling Tasks in ConfigurationDesk \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(5abce1a84a655b073239ab33e1199487_img.jpg\)](#)). A task with a high priority preempts tasks with a

low priority. The priority value must be of integer type and in the range 0 ... 127. The highest priority is defined as '0'. The priority is defined relative to the priorities of other tasks, asynchronous as well as periodic.

Tip

You can also reference a workspace variable including associated calculation rules.

Block Configuration Page (Hardware-Triggered Runnable Function Block)


Purpose

To view and change the block configuration.

Configuration

Block ID Displays the block identity of the Hardware-Triggered Runnable Function block.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```

Display signal and block IDs Lets you select to display the signal ID and the block ID of the Hardware-Triggered Runnable Function block in the Simulink model.

Sample time Lets you enter the sample time of the [Hardware-Triggered Runnable Function](#) block. The block is executed with this sample time during a Simulink simulation, i.e., the block triggers the connected function-call subsystem with the sample time.

Sample Time	Description
-1	Lets you use the sample time inherited from the behavior model.
≥ 0	Lets you specify the sample time in seconds. Use any multiple of the Fixed step size (base sample time) specified for the behavior model. In addition, you can specify a sample time offset. The syntax is as follows: [Sample time, Sample time offset], e.g., [0.01,0.002]

Sample Time	Description
	The offset must be an integer multiple of the model's fixed step size (base sample time), and must not be greater than the sample time value.

Tip

You can also reference a workspace variable in the Simulink model, including associated calculation rules.


Description Lets you enter a description of the block. This is only for the purpose of documentation.

Block Connections Page (Hardware-Triggered Runnable Function Block)

Purpose

To enable and configure data connections between model port blocks and blocks from the RTI FPGA Programming Blockset during Simulink simulations.

Note

This page is only available if you have enabled it for the use with the RTI FPGA Programming Blockset. Refer to [Processor Interface Blocks \(MicroAutoBox III, SCALEXIO\) \(RTI FPGA Programming Blockset - Processor Interface Reference\)](#) .

Software-Triggered Runnable Function Block

Purpose The Software-Triggered Runnable Function block provides methods for exporting runnable functions in predefined tasks (ConfigurationDesk and VEOS Player).

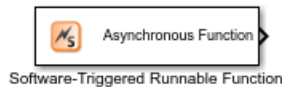
Where to go from here

Information in this section

Software-Triggered Runnable Function Block Overview.....	33
Runnable Function Page (Software-Triggered Runnable Function Block).....	34
Block Configuration Page (Software-Triggered Runnable Function Block).....	35

Software-Triggered Runnable Function Block Overview

Block illustration



Purpose To export function-call subsystems as [runnable functions](#) in predefined tasks for use in [ConfigurationDesk](#) or [VEOS Player](#).

Description The Software-Triggered Runnable Function block exports a function-call subsystem as a runnable function in a predefined task with an assigned software event. In this case, a predefined task including the runnable function with the assigned software event is available in VEOS Player or in ConfigurationDesk after an analysis of the model's task information or. In ConfigurationDesk, the software-triggered runnable function does not have to be connected to a function block. Therefore, no Runnable Function block is displayed.

Related topics

Basics

[Modeling Executable Applications and Tasks \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(99f58673407353e96a019fbca558fd72_img.jpg\)\)](#)

References

[Block Configuration Page \(Software-Triggered Runnable Function Block\)..... 35](#)
[Runnable Function Page \(Software-Triggered Runnable Function Block\)..... 34](#)

Runnable Function Page (Software-Triggered Runnable Function Block)

Purpose

To view and change the runnable function configuration.

Properties

Lets you specify the properties of the runnable function that is exported from the function-call subsystem.

Note

You can update the values on the Runnable function page via the Propagate to Simulink Model command in ConfigurationDesk. For example, if you changed the task priority in ConfigurationDesk, the Propagate to Simulink Model command updates the value of the Required priority on the Runnable function page with the task priority value you specified in ConfigurationDesk.

Name of runnable function Lets you specify a runnable function name.

Note


In ConfigurationDesk, the runnable function name must be unique within the active ConfigurationDesk application, because it serves as an identifier for the modeled asynchronous task.

The name must fulfill the following conditions:

- The name must be unique within the model.
- The name must not begin or end with /.
- The name is case-sensitive.

Signal ID Displays the signal ID of the runnable function.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```

Required priority Lets you specify a value for the task priority restriction (default value = 30). The default value 30 ensures that the task that executes the runnable function has a higher priority than the periodic tasks in the model.

For runnable functions in ConfigurationDesk, the following applies:

When configuring the tasks to which the runnable functions are assigned in ConfigurationDesk, you must make sure that the relationship between the task priorities matches the relationship between the predefined priorities, even if the absolute priority values differ from the predefined values. For details, refer to [Basics on Modeling Tasks in ConfigurationDesk \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(e474458956c9a37fbf9586ddb60a7fa1_img.jpg\)](#)). A task with a high priority preempts tasks with a low priority. The priority value must be of integer type and in the range 0 ... 127. The highest priority is defined as '0'. The priority is defined relative to the priorities of other tasks, asynchronous as well as periodic.

Tip

You can also reference a workspace variable including associated calculation rules.


Block Configuration Page (Software-Triggered Runnable Function Block)

Purpose To view and change the block configuration.

Configuration

Block ID Displays the block identity of the Software-Triggered Runnable Function block.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```

Display signal and block IDs Lets you select to display the signal ID of the runnable function and the block ID of the Software-Triggered Runnable Function block in the Simulink model.

Sample time Lets you enter the sample time of the Software-Triggered Runnable Function block. The block is executed with this sample time during a Simulink simulation, i.e., the block triggers the connected function-call subsystem with the sample time.

Sample Time	Description
-1	Lets you use the sample time inherited from the behavior model.
≥ 0	<p>Lets you specify the sample time in seconds. Use any multiple of the Fixed step size (base sample time) specified for the behavior model.</p> <p>In addition, you can specify a sample time offset. The syntax is as follows:</p> <p><code>[Sample time, Sample time offset]</code>, e.g., <code>[0.01,0.002]</code></p> <p>The offset must be an integer multiple of the model's fixed step size (base sample time), and must not be greater than the sample time value.</p>

Tip

You can also reference a workspace variable in the Simulink model, including associated calculation rules.

Description Lets you enter a description of the block. This is only for the purpose of documentation.

Model Separation Setup Block

Purpose

With the **Model Separation Setup** block, you can separate individual models from an overall model in MATLAB/Simulink.

Note

You can use the overall model for simulations in Simulink even after separating models from it.

Where to go from here	Information in this section
	Model Separation Setup Block Overview..... 37
	Model Separation Setup..... 39
	Model Separation Hook Functions..... 43

Model Separation Setup Block Overview

Block illustration



Purpose

To separate top-level subsystems as individual models from an overall model in MATLAB/Simulink, to create one MDL or SLX file (depending on the Simulink preferences) for each separated model, and to generate a model communication description file (MCD file).

Description

You can open the **Model Separation Setup** tool by adding a **Model Separation Setup** block to the overall model, and double-clicking the block. Alternatively, you can open the **Model Separation Setup** tool via the **Model Port Blocks - Open Model Separation Setup** menu command in the overall model.

Creating models You can define the models to be separated by specifying names and folders for them, and by adding one or more top-level subsystems to each model. After defining the models, you can select the models to be separated. The **Model Separation Setup** tool provides a command to create the

models containing the assigned subsystems as SLX or MDL files, depending on the Simulink preferences.

Creating MCD files The Model Separation Setup tool creates a model communication description file (MCD file) that you can import into ConfigurationDesk. The MCD file references the separated Simulink models. You can then build a multicore real-time application or a multi-processing-unit application and download it to the dSPACE real-time hardware.

Note

- The Model Interface Package for Simulink lets you generate SIC files for models. Refer to [Generating Simulink Implementation Containers \(Model Interface Package for Simulink - Modeling Guide\)](#). If you separate models and subsequently generate SIC files for them, do not work with the MCD file in ConfigurationDesk. The MCD file is created during model separation and thus refers to the separated models, not to the SIC files.
- You cannot use the MCD file with VEOS Player.
- If you separate only one model from an overall model, model separation does not generate an MCD file.

Tip

Instead of using the Model Separation Setup, you can perform model separation with the `dsmsb_separate()` API command. Refer to [dsmsb_separate \(Model Interface Package for Simulink API Reference\)](#).

Preconditions for model separation

The subsystems must fulfill the following preconditions for model separation:

- The subsystems must not have function-call, action, reset, or connector ports.
- To avoid model initialization (for example, because it takes too long), you can clear the **Initialize Model** checkbox. Then, all inports and outports of the subsystems must have an explicitly specified scalar or vectorial port width and a supported data type, or an existing Simulink.Bus object as the data type.

Additionally, the following block parameter settings must be made:

- Sampling mode = Sample-based
- Signal type = real

The following data types are supported:

- single
- double
- int8
- int16
- int32
- int64
- uint8
- uint16
- uint32

- uint64
- Boolean
- Simulink.Bus objects

Alternatively, if the checkbox is not cleared and model initialization can be performed, model separation can determine suitable settings for the inports and outports. In this case, it is also possible to use Simulink bus signals which are not specified via Simulink.Bus objects.

- The subsystems' Read/Write permissions must have one of the following settings
 - ReadWrite
 - ReadOnly

Separating models with Goto/From block connections

If you perform model separation for top-level subsystems that include Goto or From blocks, the Model Separation Setup tool creates new blocks at the top level of the separated models. The following blocks are created during model separation:

- For each Goto block in a subsystem that must be separated, a Data Output block and a suitable From block are created at the top level of the separated model. The From block is connected to the Data Output block.
- For each From block in a subsystem, a Data Inport block and a suitable Goto block are created at the top level of the separated model. The Goto block is connected to the Data Inport block.

For more information, refer to [Separation of Models Containing Goto and From Block Connections \(Model Interface Package for Simulink - Modeling Guide !\[\]\(642aa997563f9a325b310230bb5078b7_img.jpg\)](#)).

Hook functions for custom functionality

You can use hook functions to add custom functionality to the model separation process. Refer to [Model Separation Hook Functions](#) on page 43.

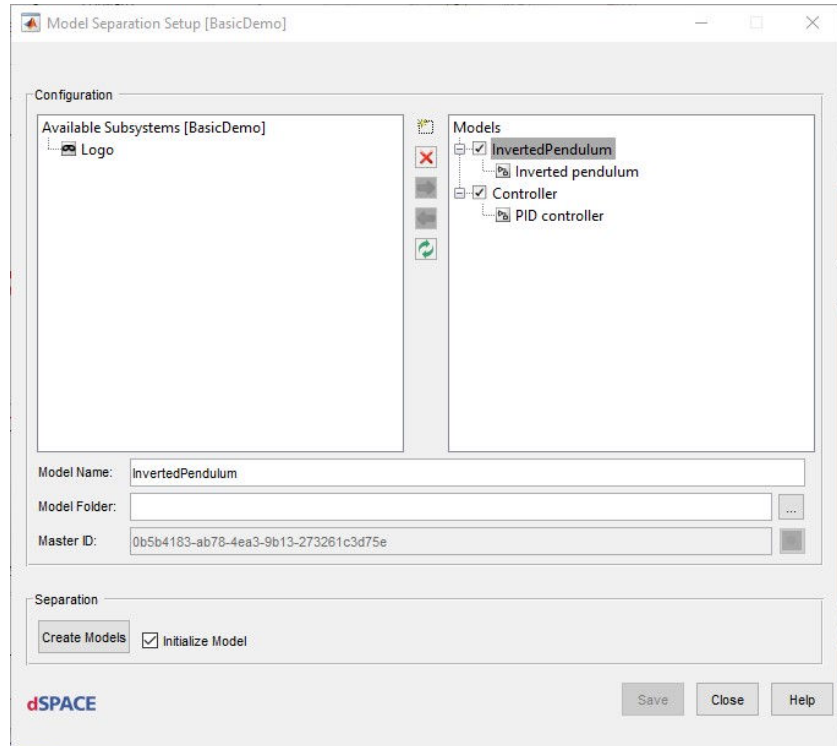
Model Separation Setup

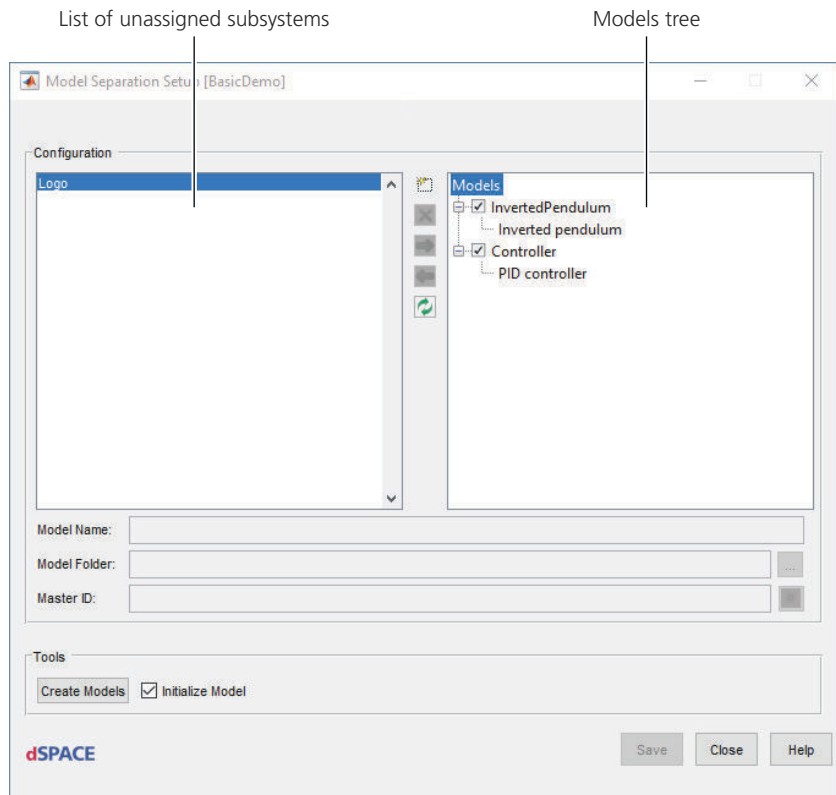
Purpose

To define and create the separated models, and to generate the model communication description file (MCD file).

Configuration

Lets you specify a model tree and define the model entries. Refer to the following illustration:





Lets you add a model entry to the **Models** tree.

Note

There is no limit to the number of model entries.



Lets you delete a model entry from the **Models** tree.



Lets you assign a subsystem to a model entry from the **Unassigned** subsystems list. You can assign multiple top-level subsystems to one model entry.



Lets you remove an assigned subsystem from a model entry.

Models Displays all the model entries and their assigned subsystems in a tree view. You can select the models to be created by selecting their checkboxes.



Analyzes the overall model and refreshes the **Unassigned Subsystems** list.


Unassigned subsystems Displays a list of all top-level subsystems that fulfill the preconditions for model separation.

Model name Lets you specify a name for the model to be created. The name must follow the conventions for Simulink model names.

Model folder Lets you specify a path to the folder to which to save each model.

Master ID Displays a UUID from which to derive block and signal IDs, which are generated during model separation.

Tip

You can assign a new ID to the signal via the  button. To enable this button, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set','EnableEditIds', true);
```

Create Models

Creates and opens the models you selected in the **Models** tree. The models are saved with the specified names in the folders you specified in the **Model folder** edit field. The models contain the assigned top-level subsystems. A **Data Inport** block or **Data Outport** block from the **Model Interface Blockset** is created for each model interface. The model port blocks are automatically connected to the subsystems' inports and outports according to the following rules.

In the separated model, an outport of a top-level subsystem is connected to a **Data Outport** block if one of the following preconditions is fulfilled in the overall model:

- The output is connected to the inport of a block that is not assigned to the same separated model.
- The output is connected to a signal line. At least one branch of the signal line is not connected to another port.
- The output is not connected to a signal line.

In the separated model, an inport of a top-level subsystem is connected to a **Data Inport** block if one of the following preconditions is fulfilled in the overall model:

- The inport is connected to the output of a block that is not assigned to the same separated model.
- The inport is connected to a signal line that is not connected to an output as a signal source.
- The inport is not connected to a signal line.

An SLX or MDL file with the specified model name (**name.slx** or **name.mdl**) is created for each model. If you separate more than one model from the overall model, a model communication description file (MCD file) is generated.

Model-specific parameter values are assigned to each separated model. Additionally, variables in the overall model's model workspace are copied to the model workspaces of the separated models, and the overall model's Data Dictionary (if it exists) is transferred to the generated models..

Note

If a model interface is a Simulink bus, such a model communication runs with a single sample time, even if the Simulink bus in the overall model is a virtual bus with multiple sample times.

Initialize Model

Lets you specify if the overall model must be initialized during model separation.

If the **Initialize Model** checkbox is selected, model separation initializes the overall model to retrieve data types and signal widths. The subsystems' inport and outport blocks do not have to explicitly specify data types and signal widths (port dimensions).

If the **Initialize Model** checkbox is cleared, model separation retrieves data types and signal widths (port dimensions) from the subsystems' inport and outport blocks, which must therefore specify data types and signal widths (port dimensions). Bus signals are supported only if the inport's or outport's data type is set to a Simulink.Bus object.

Save

Lets you save the current configuration to the overall model.

Model Separation Hook Functions

Pre-model separation and post-model separation hooks

You can use hook functions to add custom functionality to the model separation process. The Model Separation Setup tool ensures that hook functions that are on the MATLAB path and that comply with the following naming conventions are called at an appropriate time:

`*_dsmsbhook.m` with the following syntax:

```
[errCode, errMsg] = *_dsmsbhook(method, varargin)
```

The hook function must provide the following methods:

- PreSeparationCheck
- PreSeparation
- PostSeparation

PreSeparationCheck

The **PreSeparationCheck** hook is called once before model separation. It has the following syntax:

```
[errCode, errMsg] = *_dsmsbhook('PreSeparationCheck', modelHdl, modelCfg)
```

The **PreSeparationCheck** hook has the following parameters:

errCode Numeric error code:

0: No error occurred.

1: An error occurred.

Note

If `errCode != 0`, model separation does not start.

errMsg Error message describing the error that occurred.

Empty string: No error.

modelHdl Handle of the overall model

modelCfg The `modelCfg` parameter is a 1 x n struct with the data fields described in the following table. n is the number of models to be separated from the overall model.

Data Field Name	Data Type	Description
Name	String	Name of the model to be created
Modelfolder	String	Name of the folder the created model is saved in
AssignedSubsystems	Cell array of strings	List of the names of the assigned subsystems
Create	Boolean	0: The model is not created. 1: The model is created.

PreSeparation

The `PreSeparation` hook is called for each model immediately after it is created, and before the subsystems and model port blocks are added. It has the following syntax:

```
[errCode, errMsg] = *_dsmsbhook('PreSeparation', modelHdl, modelIdx, modelCfg)
```

The `PreSeparation` hook has the following parameters:

errCode Numeric error code:

0: No error occurred.

1: An error occurred.

errMsg Error message describing the error that occurred.

Empty string: No error.

modelHdl The `modelHdl` parameter specifies the handle of the overall model.

modelIdx The `modelIdx` parameter specifies the index of the model to be created. It can be used to access the `modelCfg` data fields. Refer to the following table.

modelCfg The `modelCfg` parameter is a struct with the data fields described in the following table.

Data Field Name	Data Type	Description
Name	String	Name of the model to be created
ModelFolder	String	Name of the folder the created model is saved in
AssignedSubsystems	Cell array of strings	List of the names of the assigned subsystems
Create	Boolean	0: The model is not created. 1: The model is created.

PostSeparation

The `PostSeparation` hook is called after model separation and before the model is saved. It has the following syntax:

```
[errCode, errMsg] = *_dsmsbhook('PostSeparation', modelHdl, modelIdx, modelCfg)
```

The `PostSeparation` hook has the following parameters:

errCode Numeric error code:

0: No error occurred.

1: An error occurred.

errMsg Error message describing the error that occurred.

Empty string: No error.

modelHdl The `modelHdl` parameter specifies the handle of the overall model.

modelIdx The `modelIdx` parameter specifies the index of the model to be created. It can be used to access the `modelCfg` data fields. Refer to the following table.

modelCfg The `modelCfg` parameter is a struct with the data fields described in the following table.

Data Field Name	Data Type	Description
Name	String	Name of the model to be created.

Data Field Name	Data Type	Description
Modelfolder	String	Name of the folder the created model is saved in.
AssignedSubsystems	Cell array of strings	List of the names of the assigned subsystems.
Create	Boolean	0: The model is not created. 1: The model is created.

Menu Commands of Model Port Blocks

Introduction

The Model Interface Blockset provides commands that are accessible via the menu bar of Simulink models and Simulink libraries.

Where to go from here

Information in this section

Check Model for DSRT Code Generation.....	48
To check if the Simulink behavior model fulfills the requirements for code generation.	
Create Data Inport Block from Bus Creator Block.....	48
To create a Data Inport block with a structured data port from a Simulink Bus Creator block.	
Create Data Outport Block from Bus Creator Block.....	52
To create a Data Outport block with a structured data port from a Simulink Bus Creator block.	
Create Inverse Model Port Block.....	54
To create inverse model port blocks that you can easily use to set up model communication in ConfigurationDesk.	
Create Model Port Block Periphery.....	56
To create a block periphery for each unconnected data port of the selected model port blocks.	
Open Model Interface Blockset.....	57
To open the Model Interface Blockset.	
Open Model Separation Setup.....	58
To open the Model Separation Setup dialog.	
Paste and Keep IDs.....	58
To paste model port blocks to a Simulink system, including the identifiers of the blocks and their signals.	
Select All Output Blocks in This Model.....	59
To select all Data Outport blocks in the current model.	
Update Selected Output Blocks from Input Signals.....	60
To change all the data outputs of a Data Outport block to match the hierarchical structure of the input signals connected to these data outputs.	

Check Model for DSRT Code Generation

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None

Purpose

To check if the Simulink behavior model meets the requirements for code generation.

Description

If you execute this command, error messages are displayed in the following cases:

- You have placed model port blocks in Stateflow charts, MATLAB Function blocks, iterated subsystems, or referenced models.
- Block or signal IDs of active model port blocks are not set or are not unique. A model port block in the behavior model is active, if it is not commented out, or if it resides in the active subsystem of a Variant Subsystem block. For more information on Variant Subsystem blocks, refer to the Simulink documentation
- The model contains blocks or parameters that are not supported for DSRT code generation.

If one or more of the above-mentioned cases apply, a message informs you, that the behavior model does not fulfill the preconditions for DSRT code generation. Error messages displayed in the MATLAB Command Window give you information about the errors. They also provide links to the affected blocks in the behavior model and to the related topics in the Model Interface Package for Simulink Message Reference for more information.

Create Data Inport Block from Bus Creator Block

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None

Note

This command is also available as the `dsmpb_generatemodelportblock()` API command. Refer to [dsmpb_generatemodelportblock \(Model Interface Package for Simulink API Reference\)](#).

Purpose

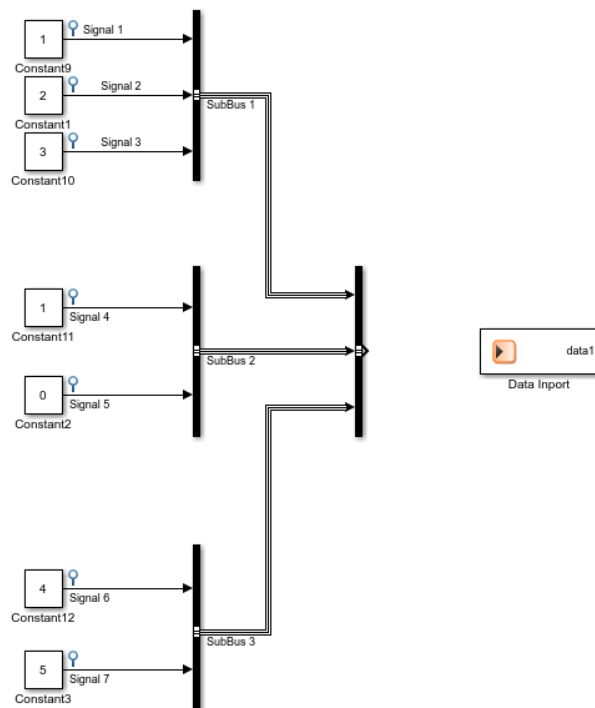
To create a [Data Import](#) block with a structured data port from a Simulink Bus Creator block.

Description

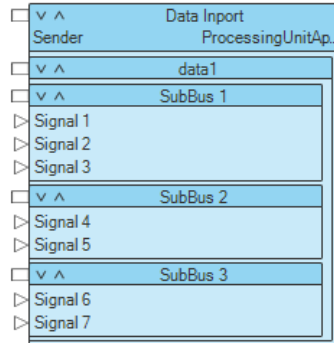
The Create Data Inport Block from Bus Creator Block command generates a structured Data Inport block for a selected Bus Creator block. The Data Inport block with the structured data port reflects the structure of the Bus Creator block's bus signal. In ConfigurationDesk, it is represented by a model port block with a structured data port whose ports can be mapped to function ports or other model port blocks. In VEOS Player, it is represented by VPU ports.

Model port block with structured data port in ConfigurationDesk

The following illustration shows a Simulink model with a Data Inport block that has structured data ports.

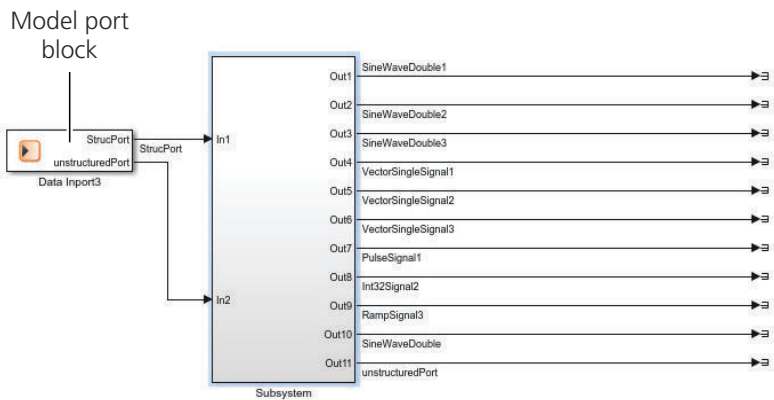


If you add the Simulink model to a ConfigurationDesk application, ConfigurationDesk creates a model port block with structured data ports for the Data Inport block.

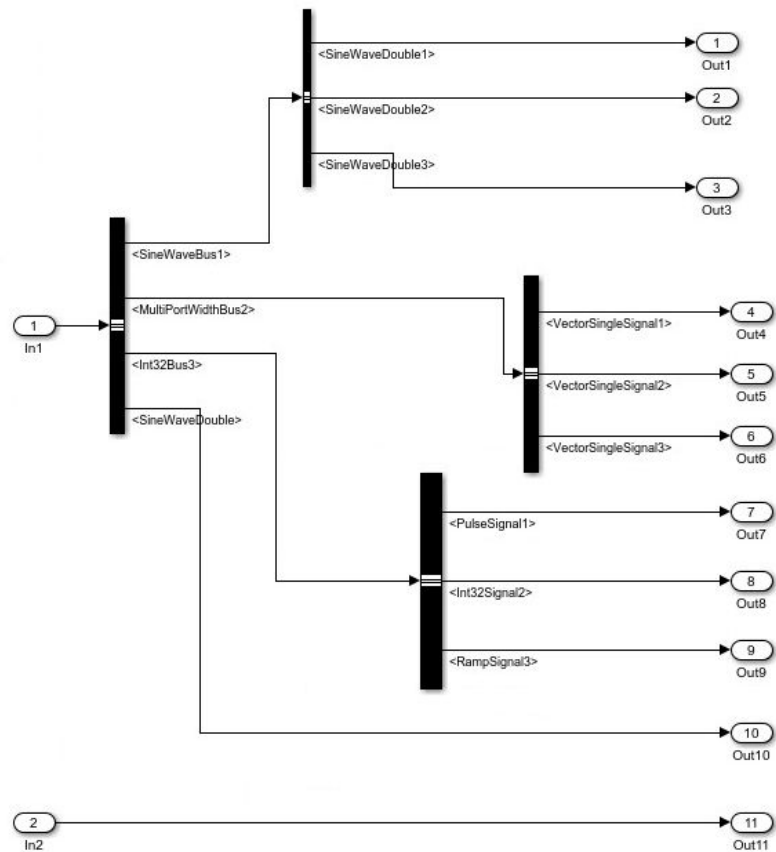


VPU ports created for buses in VEOS Player

Structured and unstructured ports of model port blocks The following illustration shows an excerpt from a Simulink model with a model port block connected to a subsystem. The model port block has one structured port and one unstructured port.

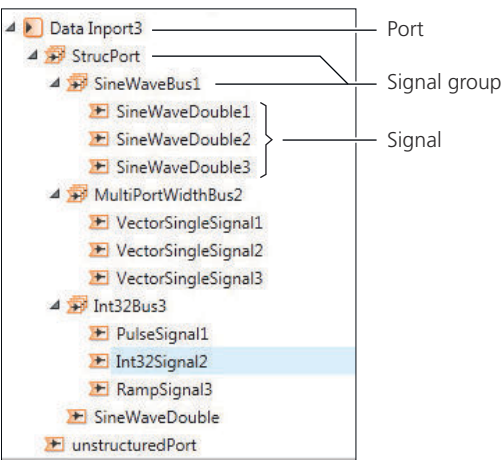


Model port structure The following illustration shows the internal structure of the two ports:



Display in VEOS Player When you import a Simulink implementation container to VEOS Player, VEOS Player creates VPU ports for the model port blocks.

The following illustration of VEOS Player shows the hierarchical structure of the VPU ports of the example model port block shown above.



VEOS Player displays the dimension of a signal via the Data width property of the related VPU port in the Configuration Grid.

Note

Structured data ports for bus signals can also be created via Simulink.Bus objects defined in the base workspace or in the model's data dictionary.

Related topics

Basics

[Exchanging Simulink Bus Signals Between Simulink and ConfigurationDesk or VEOS Player \(Model Interface Package for Simulink - Modeling Guide\)](#)

Create Data Outport Block from Bus Creator Block

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None

Note

This command is also available as the `dsmpb_generatemodelportblock()` API command. Refer to [dsmpb_generatemodelportblock \(Model Interface Package for Simulink API Reference\)](#).

Purpose

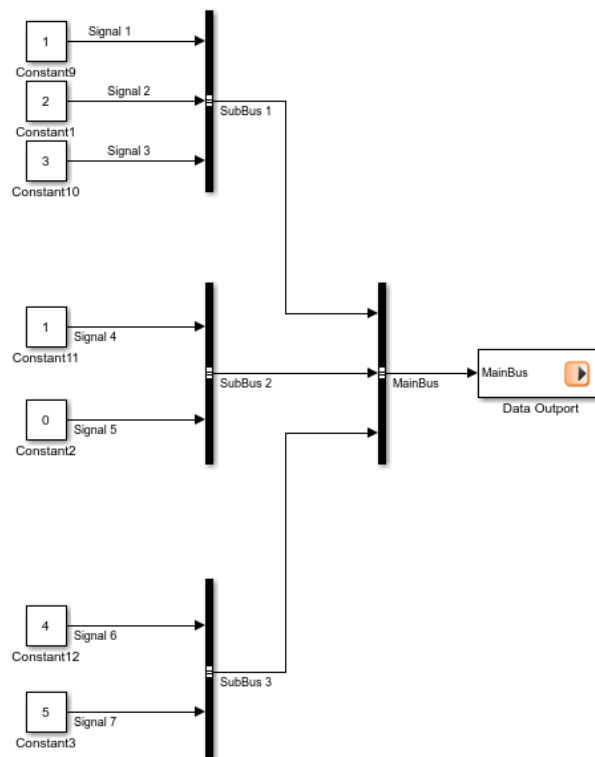
To create a [Data Output](#) block with a structured data port from a Simulink Bus Creator block.

Description

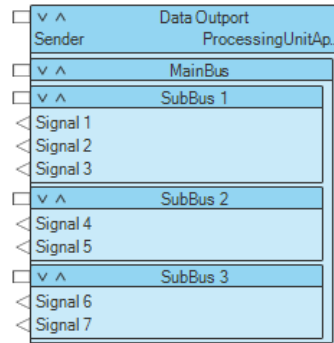
The Create Data Output Block from Bus Creator Block command generates a structured Data Output block for a selected Bus Creator block. The Data Output block with the structured data port reflects the structure of the Bus Creator block's bus signal. In ConfigurationDesk, it is represented by a model port block with a structured data port whose ports can be mapped to function blocks or other model port blocks. In VEOS Player, it is represented by VPU ports.

Model port block with structured data port in ConfigurationDesk

The following illustration shows a Simulink model with a Data Output block that has structured data ports.



If you add the Simulink model to a ConfigurationDesk application, ConfigurationDesk creates a model port block with structured data ports for the Data Output block.



Note

Structured data ports for bus signals can also be created via Simulink.Bus objects defined in the base workspace.

VPU ports created for buses in VEOS Player

When you import a Simulink implementation to VEOS Player, VEOS Player creates VPU ports for the model port blocks and displays the hierarchical structure of the VPU ports. VEOS Player displays the dimension of a signal via the **Data width** property of the related VPU port in the **Configuration Grid**.

Related topics

Basics

[Exchanging Simulink Bus Signals Between Simulink and ConfigurationDesk or VEOS Player \(Model Interface Package for Simulink - Modeling Guide 📖\)](#)

Create Inverse Model Port Block

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	<ul style="list-style-type: none"> A Data Inport block — Model Port Blocks A Data Outport block — Model Port Blocks
Shortcut key	None
Toolbar icon	None

Purpose

To create [inverse model port blocks](#) that you can easily use to set up model communication in ConfigurationDesk.

Result

The Create Inverse Model Port Block command creates a new model port block for each selected model port block. The new model port blocks have the following characteristics:

- They have the same structure and port configuration, for example, port names, as the original model port blocks.
- They have the inverse data direction of the original model port block.
- They have the same names as the original model port blocks, but with an additional postfix. The postfix indicates the data direction of the inverse model port block. The following table shows some examples:

Name of Original Model Port Block	Name of Preconfigured Inverse Model Port Block
MyDataInportBlock	MyDataInportBlock_Out ¹⁾
MyDataOutportBlock	MyDataOutportBlock_In ¹⁾

¹⁾ If the Simulink model already contains a model port block with the name of the new model port block, the new model port block gets an incrementing number as an additional postfix.

- Each new model port block gets a unique block ID and unique signal IDs.

Description

You can use the inverse model port blocks to set up model communication in ConfigurationDesk. You can add the inverse model port blocks to the desired behavior models via cut & paste. After you add the Simulink models to the ConfigurationDesk application, you can easily set up model communication between them.

Note

The Create Inverse Model Port Block command is also available as an API command. Refer to [dsmpb_createinversedmodelportblock](#) (Model Interface Package for Simulink API Reference).

Related topics**Basics**

[Simplified Preparation of Model Interfaces for Model Communication](#) (Model Interface Package for Simulink - Modeling Guide).

Create Model Port Block Periphery

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	<ul style="list-style-type: none"> ▪ A Data Inport block — Model Port Blocks ▪ A Data Outport block — Model Port Blocks
Shortcut key	None
Toolbar icon	None

Note

You can also use the `dsmpb_createmodelportblockperiphery()` API command to create the block periphery. Refer to [dsmpb_createmodelportblockperiphery \(Model Interface Package for Simulink API Reference\)](#).

Purpose

To create a block periphery for each unconnected data port of the selected model port blocks.

Description

The Create Model Port Block Periphery command creates a peripheral Simulink block for each unconnected data port of selected [model port blocks](#). This is especially useful for model port blocks with unconnected structured data ports. Each newly created block is automatically connected to the related data port. The following table gives an overview of the created model port block periphery, depending on the model port type:

Model Port Type	Created Model Port Block Periphery
Unstructured data outport	Simulink Constant block with the following properties: <ul style="list-style-type: none"> ▪ Value = <code>zeros(1, <Width>)</code>. <Width> is the width of the associated signal as specified in the data port block's signal configuration. ▪ Type as specified by the data port block's signal configuration.
Structured data outport	Simulink Subsystem block that contains a representation of the hierarchical bus structure. This includes Constant blocks, Bus Creator blocks and Outport blocks as well as suitably named signals.
Unstructured data inport	Simulink Terminator block.

Model Port Type	Created Model Port Block Periphery
Structured data inport	Simulink Subsystem block that contains a representation of the hierarchical bus structure. This includes Terminator blocks, Bus Selector blocks and Outport blocks as well as suitably named signals.

Naming scheme

The names of the new peripheral blocks are derived from the data ports for which they are created. In addition, new Constant blocks and Terminator blocks get an appropriate **Const_** or **Term_** prefix. In the case of a naming conflict, the new block gets a number as a suffix which is increased if necessary.

Related topics**Basics**

[Creating a Suitable Simulink Block Periphery for Unconnected Data Ports \(Model Interface Package for Simulink - Modeling Guide !\[\]\(e3f8612927870f2e0f9f5989e6dd3064_img.jpg\)](#))

Open Model Interface Blockset

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None
MATLAB Command Window	mips
Simulink Library Browser	dSPACE Model Interface Blockset

Purpose

To open the Model Interface Blockset.

Result

The blockset is opened.

Open Model Separation Setup

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None
MATLAB Command Window	None

Purpose

To open the **Model Separation Setup**.

Result

The **Model Separation Setup** is opened.

Paste and Keep IDs

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	Simulink model
Shortcut key	Ctrl+Alt+V
Toolbar icon	None

Purpose

To paste model port blocks to a Simulink model, including the internal identification (ID) of the blocks and their signals.

Description

The **Paste and Keep IDs** command lets you create model port blocks with identical IDs. Model port blocks with identical IDs are required in the following use cases:

- Creating behavior model variants with identical model interfaces.
- Creating variants of behavior model parts using Variant Subsystem blocks, for example.

- Copying model port blocks from a temporary interface model created by ConfigurationDesk to a behavior model.

Note

This command is available only if the Copy command was executed command beforehand.

Related topics

Basics

[Adding the Generated Model Interface to Your Behavior Model via an Interface Model \(Model Interface Package for Simulink - Modeling Guide !\[\]\(cbe2492b119e39e02a1dab2af4a4b296_img.jpg\)\)](#)
[Using Model Port Blocks to Handle Simulink Behavior Model Variants \(ConfigurationDesk Real-Time Implementation Guide !\[\]\(2f36c159ea3670f7a62f64a4f1cf5c05_img.jpg\)\)](#)

Select All Outport Blocks in This Model

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	None
Shortcut key	None
Toolbar icon	None

Purpose

To select all [Data Outport !\[\]\(0d5ec72f61334709c3fc9450209b754f_img.jpg\)](#) blocks in the current model.

Description

The Select All Outport Blocks in This Model command lets you select all Data Outport blocks in the current model. You can then execute the Update Selected Outport Blocks from Input Signals command to update all Data Outport blocks in the current model simultaneously.

Related topics

References

[Update Selected Outport Blocks from Input Signals..... 60](#)

Update Selected Output Blocks from Input Signals

Access

You can access this command via:

Menu bar	Model Port Blocks
Context menu of	A Data Output block — Model Port Blocks
Shortcut key	None
Toolbar icon	None

Note

This command is also available as the `dsmpb_updatemodelportblock()` M function. Refer to [dsmpb_updatemodelportblock \(Model Interface Package for Simulink API Reference\)](#).

Purpose

To update the signal configuration of all ports of a [Data Output](#) block to match the input signals connected to these ports.

Description

With the Update Selected Output Blocks from Input Signals command, you can update the signal configuration of one or more selected Data Output blocks. This is done to match the hierarchical structure and properties of the input signals connected to the block's ports. During the update, the rules described below apply to data port identifiers, port properties, and the change of the data port type. Each signal keeps its ID when its name and path within the signal hierarchy remain unchanged.

Signal properties For signal properties, the following applies:

- If signals of the connected Simulink bus signal have no equivalents in the signal configuration, a suitable signal is created.
- If signals contained in the port's signal configuration have no equivalents in the connected Simulink bus signal, they are removed from the signal configuration.
- The signals get their names, data types and widths from the related Simulink signal. If the Simulink signal is not named, the signal gets the default name data, with a postfix number to make it unique.

For more information on model port blocks and data port identifiers, refer to [Basics on Model Port Block IDs and Signal IDs \(Model Interface Package for Simulink - Modeling Guide\)](#).

Related topics

Basics

[Exchanging Simulink Bus Signals Between Simulink and ConfigurationDesk or VEOS Player \(Model Interface Package for Simulink - Modeling Guide !\[\]\(bd1a142de767a21e5362c595f844a4ff_img.jpg\)\)](#)

Menu Commands for the Remote Access of ConfigurationDesk

Introduction The Model Interface Package for Simulink provides commands for the remote access of ConfigurationDesk.

Where to go from here

Information in this section

Add Model to ConfigurationDesk Project: <ConfigurationDesk Project Name>.....	63
Analyze Model in ConfigurationDesk (Including Task Information).	64
Analyze Model in ConfigurationDesk (Model Interface Only).....	66
Create ConfigurationDesk Project from Model.....	67
Delete Selection and Connected Blocks in ConfigurationDesk.....	68
Open a ConfigurationDesk Project.....	69
Save ConfigurationDesk Project.....	69
Show Connected Block in ConfigurationDesk.....	70
Show in ConfigurationDesk.....	71
Start ConfigurationDesk Build.....	71

Add Model to ConfigurationDesk Project: <ConfigurationDesk Project Name>

Access This command is available only if a ConfigurationDesk project with an active ConfigurationDesk application is open. There must not be another [behavior](#)

[model](#) with the same name as the Simulink behavior model in the model topology of the active ConfigurationDesk application.

Menu bar	ConfigurationDesk
Context menu of	None
Shortcut key	None
Toolbar icon	None

Purpose To add the current Simulink behavior model to the active ConfigurationDesk application.

Description If you execute this command, a dialog opens showing you that the following actions are performed, if required:

- The Simulink behavior model is configured for the dSPACE Run-Time Target (dsrt.tlc).
- The Simulink behavior model is saved. You are prompted to specify a file name and a location for the model.

If you click **OK**, the command adds the current Simulink model to the active ConfigurationDesk application. The model interface is analyzed, and model port blocks representing **Data Inport** blocks, **Data Outport** blocks and **Hardware-Triggered Runnable Function** blocks in ConfigurationDesk are displayed in the **Model Browser** and in the **Model-Function Mapping Browser**. Preconfigured tasks of the current Simulink model are displayed in ConfigurationDesk's **Task Configuration** table view.

Related topics

References

[Analyze Model in ConfigurationDesk \(Including Task Information\)..... 64](#)

Analyze Model in ConfigurationDesk (Including Task Information)

Access This command is available only if the following preconditions are fulfilled:

- A ConfigurationDesk project is open.
- The current Simulink model is part of the active ConfigurationDesk application.

Menu bar	ConfigurationDesk
Context menu of	None
Shortcut key	Ctrl+Alt+D
Toolbar icon	None

Purpose

To perform an analysis of the model topology and the task information of the current Simulink behavior model with ConfigurationDesk.

Description

If you select this command, the model interface and the task information provided by the Simulink behavior model are analyzed in ConfigurationDesk. For each Data Input block, Data Output block, and Hardware-Triggered Runnable Function block in the Simulink model, a model port block representing the relevant block is created in ConfigurationDesk's Model Browser and in the Model-Function Mapping Browser. Subsystems and Simulink Inports and Outports are also displayed in the Model-Function Mapping Browser. Model port blocks that are already represented by blocks in ConfigurationDesk are updated with current values, if required. Refer to [Basics on Model Port Block IDs and Signal IDs \(Model Interface Package for Simulink - Modeling Guide\)](#). Additionally, all predefined task information provided by the Simulink model is analyzed and displayed in the Task Configuration table view and in the Properties Browser.

You can use these elements to model the executable application (= real-time application) in ConfigurationDesk.

Tip

Analyzing the task information requires the initialization of the complete Simulink behavior model. Depending on model complexity, this can take a long time. If you changed only the model port blocks or the structure of the Simulink behavior model, it is recommended to select the **Analyze Model (Model Interface Only)** command to make the changes known to ConfigurationDesk.

Note

In the following case, the results of the model analysis and the code generation differ:

- The signal of an In Bus Element block is specified by a Simulink.Bus object.
- The In Bus Element block is connected to a Bus Selector block that selects a subset of the bus signals.
- No other signal is used.

In the above case, a model port block with the complete bus hierarchy is displayed in ConfigurationDesk after a model analysis. However, after code generation, a model port block with only those bus signals that are selected by the Bus Selector block is displayed in ConfigurationDesk.

Related topics**References**

[Analyze Model in ConfigurationDesk \(Model Interface Only\)](#)..... 66

Analyze Model in ConfigurationDesk (Model Interface Only)

Purpose

To analyze the Simulink model's interface.

Note

This command is available only if the following preconditions are fulfilled:

- A ConfigurationDesk project is open.
- The current Simulink model was added to the active ConfigurationDesk application.

Description

If you select this command, the model interface of the Simulink behavior model is analyzed in ConfigurationDesk. For each [Data Inport](#) block, [Data Outport](#) block, and [Hardware-Triggered Runnable Function](#) block in the Simulink model, a model port block representing the relevant block is created in ConfigurationDesk's Model Browser and in the Model-Function Mapping Browser. Model port blocks that are already represented by blocks in ConfigurationDesk are updated with current values, if required. For more information on ID handling, refer to [Basics on Model Port Block IDs and Signal IDs \(Model Interface Package for Simulink - Modeling Guide\)](#).

Note

- If you select this command, the task information of the Simulink model is not analyzed by ConfigurationDesk. If required, you must use the [Analyze Model in ConfigurationDesk \(Including Task Information\)](#) command.
- Performing an analysis of a Simulink model that contains root-level In Bus Element and Out Bus Element blocks requires an initialization of the model. Therefore, an analysis of the model interface via the [Analyze Simulink Model \(Model Interface Only\)](#) command is not possible in this case.

Related topics

References

[Analyze Model in ConfigurationDesk \(Including Task Information\)](#)..... 64

Create ConfigurationDesk Project from Model

Access

This command is available only if no project is open in ConfigurationDesk.

Menu bar	ConfigurationDesk
Context menu of	None
Shortcut key	None
Toolbar icon	None

Purpose

To create a project in [ConfigurationDesk](#) with a ConfigurationDesk application that contains the current Simulink behavior model.

Description

If required, the Simulink [behavior model](#) is configured for the use in ConfigurationDesk. In this case, a dialog opens to make you aware of the automatic configuration.

If you select this command, a dialog opens that lets you confirm that the Simulink behavior model is prepared for use in ConfigurationDesk. If you click OK, the following actions are performed:

- The Simulink behavior model is configured for the dSPACE Run-Time Target (dsrt.tlc).
- The Simulink behavior model is saved. You are prompted to specify a file name and a location for the model.

Then, ConfigurationDesk is brought to the front and the File – New page is opened. You can specify the following parameters:

Parameter	Description	Default
Root directory	Lets you select a project root directory from the list. The Browse button opens the Project page of the ConfigurationDesk Properties dialog, which lets you specify additional root directories.	Current root directory.
Project name	Lets you modify the project name.	Proj_<modelName>
Application name	Lets you modify the application name.	App_<modelName>
Models	Lets you select additional models to be added to the ConfigurationDesk application.	The current Simulink model.
Hardware	Lets you add a hardware topology to the ConfigurationDesk application.	The hardware that you have saved as the default hardware.

If you click **Create**, a new ConfigurationDesk project with the specified parameters is created. The new project opens, and the new ConfigurationDesk application is active. The selected models are added to the ConfigurationDesk application. The model interface is analyzed and model port blocks representing Data Inport blocks, Data Outport blocks, and Hardware-Triggered Runnable Function blocks in ConfigurationDesk are displayed in the Model Browser and in the Model-Function Mapping Browser. Task information of the models is analyzed. Preconfigured tasks are created and displayed in the Task Configuration table view.

Delete Selection and Connected Blocks in ConfigurationDesk

Access

This command is available only if you selected one or more Data Inport blocks, Data Outport blocks, or Hardware-Triggered Runnable Function blocks in the Simulink behavior model.

Menu bar	None
Context menu of	<ul style="list-style-type: none">▪ Data Inport block — ConfigurationDesk▪ Data Outport block — ConfigurationDesk▪ Hardware-Triggered Runnable Function block — ConfigurationDesk
Shortcut key	None
Toolbar icon	None

Purpose

To delete the following elements in one step:

- In Simulink: The selected Data Inport, Data Outport, or Hardware-Triggered Runnable Function blocks in the Simulink model.
- In ConfigurationDesk:
 - The model port blocks that correspond to the selected Data Inport, Data Outport, or Hardware-Triggered Runnable Function blocks in ConfigurationDesk.
 - The function blocks to which the corresponding model port blocks are mapped in ConfigurationDesk.

Description

If you select this command, the **Delete Model Port Block and Function** dialog opens, prompting you to confirm that you really want to perform this operation. If you click **Delete**, the elements listed above are deleted.

Note

- You cannot undo this command in Simulink.
- The Model Interface Package for Simulink lets you activate or deactivate the prompt via the `dsmpb_pref('Set', 'AskBeforeDelete', true);` API command.

Open a ConfigurationDesk Project

Access

This command is available only if no project is open in ConfigurationDesk.

Menu bar	ConfigurationDesk
Context menu of	None
Shortcut key	None
Toolbar icon	None

Purpose

To open a project in [ConfigurationDesk](#) .

Description

If you select this command, ConfigurationDesk is brought to the front. The Select a Project dialog opens, prompting you to select the ConfigurationDesk project you want to open. You can also select a ConfigurationDesk application of the project that you want to be active after opening the project. For more information on opening a ConfigurationDesk project, refer to [Open Project/Open Project and Application \(ConfigurationDesk User Interface Reference !\[\]\(6bb0e4f14c4133b37d2887cb37e67ddd_img.jpg\)\)](#).

Save ConfigurationDesk Project

Access

This command is available only if there are unsaved changes in the open ConfigurationDesk project or its active ConfigurationDesk application.

Menu bar	ConfigurationDesk
Context menu of	None

Shortcut key	Ctrl+Alt+S
Toolbar icon	None

Purpose To save the open [ConfigurationDesk](#) project and its active ConfigurationDesk application.

Description A dialog is opened to show that the open ConfigurationDesk project and the active ConfigurationDesk application are saved. If there are any unsaved changes in the Simulink behavior model, it is saved without confirmation.

If other behavior models that are part of the ConfigurationDesk application were changed, you are prompted to save the models. To specify not to be prompted every time if you want to save the models, you must type the following command in the MATLAB Command Window:

```
dsmpb_pref('Set', 'ModelSaveMode', 'SaveWithoutConfirmation');
```

Show Connected Block in ConfigurationDesk

Access This command is available only if the following preconditions are fulfilled:

- You selected a Data Input block, Data Output block, or Hardware-Triggered Runnable Function block in the Simulink model.
- The Data Input block, Data Output block, or Hardware-Triggered Runnable Function block is a part of the model topology of the ConfigurationDesk project.
- The Data Input block, Data Output block, or Hardware-Triggered Runnable Function block is mapped to exactly one function block in ConfigurationDesk.

Menu bar	ConfigurationDesk
Context menu of	<ul style="list-style-type: none"> ▪ Data Input block — ConfigurationDesk ▪ Data Output block — ConfigurationDesk ▪ Hardware-Triggered Runnable Function block — ConfigurationDesk
Shortcut key	Ctrl+Alt+H
Toolbar icon	None

Purpose To show the function block, the selected Data Input block, Data Output block, or Hardware-Triggered Runnable Function block is mapped to in [ConfigurationDesk](#).

Description

If you select this command, the function block to which the model port block is mapped in ConfigurationDesk is selected in the **Model-Function Mapping Browser**. You can now directly view and change the configuration of the function block. If necessary, you can propagate any changes in the function block configuration back to the Simulink behavior model.

Show in ConfigurationDesk

Access

This command is available only if the following preconditions are fulfilled:

- A ConfigurationDesk project is open.
- The current Simulink model was added to the active ConfigurationDesk application.

Menu bar	ConfigurationDesk
Context menu of	<ul style="list-style-type: none"> ▪ A Data Inport block — ConfigurationDesk ▪ A Data Outport block — ConfigurationDesk ▪ A Hardware-Triggered Runnable Function block — ConfigurationDesk ▪ A subsystem ▪ The background of the model
Shortcut key	Ctrl+Alt+G
Toolbar icon	None

Purpose

To show the Simulink behavior model in the ConfigurationDesk project.

Description

If a Data Inport block, Data Outport block, or Hardware-Triggered Runnable Function block, or a subsystem is selected in the Simulink behavior model and you select this command, the related element is selected in the **Model-Function Mapping Browser**.

Start ConfigurationDesk Build

Access

This command is available only if the following preconditions are fulfilled:

- A ConfigurationDesk project is open.
- The current Simulink model was added to the active ConfigurationDesk application.

Menu bar	ConfigurationDesk
Context menu of	None
Shortcut key	Ctrl+Alt+B
Toolbar icon	None

Purpose

To start the build process in [ConfigurationDesk](#)¹. If you select this command, ConfigurationDesk is brought to the front. Messages in the Build Log Viewer indicate the progress of the build process.

Description

The build process for the real-time application is started in ConfigurationDesk. During the build process, messages in ConfigurationDesk's Build Log Viewer indicate the progress of the build process in chronological order. After the build process has finished, its results can be found at <ConfigurationDesk Application>\Build Results.

M Files for Customizing the Simulation Environment

Introduction dSPACE provides M files for customizing the MATLAB startup and shutdown procedure.

Where to go from here	Information in this section
	startup.m 73 To carry out user-specific commands before the dSPACE software is initialized.
	dsstartup.m 74 To carry out user-specific commands after the dSPACE software is initialized.
	dspoststartup.m 75 To carry out user-specific commands after the initialization phase of the dSPACE software.
	dsfinish.m 76 To carry out user-specific commands before MATLAB is shut down.
	dsmpb_pref 76 To read, write, and save the preferences of the Model Interface Package for Simulink.

startup.m

Purpose To carry out user-specific commands during the dSPACE software is initialized.

Result Each time you start MATLAB, this M file is executed after the dSPACE software is initialized.

Description

Place this M file in the MATLAB search path, for example, in MATLAB's working folder. In general, all valid MATLAB commands are allowed in it. Use the standard M file syntax.

The **startup.m** file is a standard MATLAB feature. For further information type **doc startup** at the MATLAB prompt.

Related topics**HowTos**

[How to Customize the MATLAB Start-Up \(Model Interface Package for Simulink - Modeling Guide !\[\]\(a870788d6ed9b8fd294b7654a8c8526b_img.jpg\)\)](#)

References

[dsstartup.m..... 74](#)

dsstartup.m

Purpose

To carry out user-specific commands after the dSPACE software is initialized.

Result

Each time you start MATLAB, this M-file is executed after the dSPACE software is initialized.

Description

Create a new M file and place it in the MATLAB search path, for example, in MATLAB's working folder. In general, all valid MATLAB commands are allowed in it, including the dSPACE-specific commands. Use the standard M file syntax.

Example

Suppose you want MATLAB to automatically change to your working folder **d:\work** and open the Simulink model **my_model.slx**. Create the **dsstartup.m** file and write the following commands to it:

```
cd d:\work
my_model
```

Related topics

HowTos

[How to Customize the MATLAB Start-Up \(Model Interface Package for Simulink - Modeling Guide !\[\]\(4729e517bc6a7cd81c8025b9646574fb_img.jpg\)\)](#)

References

dsfinish.m	76
dspoststartup.m	75
startup.m	73

dspoststartup.m

Purpose

To carry out user-specific commands after the initialization phase of the dSPACE software.

Result

Each time you start MATLAB, this M-file is executed after the initialization phase of the dSPACE software.

Description

If you create the M file representing this optional script, you can add generally all valid MATLAB commands and dSPACE-specific commands in it. The file must be placed in the MATLAB search path. It is executed directly in the MATLAB base workspace, which is necessary for executing some specific MATLAB functions.

The script is also executed if the initialization of the dSPACE software failed. It is executed after **dsstartup.m**.

Related topics

HowTos

[How to Customize the MATLAB Start-Up \(Model Interface Package for Simulink - Modeling Guide !\[\]\(2bae76de5ebbd5c4d7d47162f1673734_img.jpg\)\)](#)

References

dsstartup.m	74
-----------------------------------	----

dsfinish.m

Purpose	To carry out user-specific commands before MATLAB is shut down.
Result	Each time you exit MATLAB, this M file is executed.
Description	Create a new M file and place it in the MATLAB search path, for example, in MATLAB's working folder. All valid MATLAB commands are allowed in it, including the dSPACE-specific commands. Use the standard M file syntax.

Note

- You can specify several **dsfinish.m** files. All **dsfinish.m** files located in the MATLAB search path are executed before MATLAB is shutdown.
- For compatibility reasons available user-specific **finish.m** files are executed. This behavior can, however, change with future MATLAB versions. It is therefore recommended to use **dsfinish.m** files.

Related topics

HowTos

[How to Customize the MATLAB Start-Up \(Model Interface Package for Simulink - Modeling Guide !\[\]\(ec9132f1d27c8919987d92907322654d_img.jpg\)\)](#)

References

dsstartup.m	74
startup.m	73

dsmplib_pref

Purpose	To write, read, and save the preferences of the Model Interface Package for Simulink.
----------------	---

Description	<p>The dsmplib_pref() API command lets you perform the following actions:</p> <ul style="list-style-type: none"> ▪ Write the preferences of the Model Interface Package for Simulink. ▪ Read the preferences of the Model Interface Package for Simulink. ▪ Save the preferences of the Model Interface Package for Simulink to a file.
--------------------	---

Writing preferences The following preferences can be set:

- **ModelSaveMode:**
Lets you specify whether and how to save modified models when they are closed in ConfigurationDesk.
- **AskBeforeDelete:**
Lets you specify whether to ask you to confirm the deletion of a block from ConfigurationDesk.
- **EnableEditIDs:**
Lets you enable or disable the assignment of new block and signal IDs in model port block dialogs.
- **CreateBusObjectsDuringPropagation:**
Lets you specify that Simulink.Bus objects are created for structured data ports when ConfigurationDesk propagates data to a model or when a new Simulink model is created via the **Generate New Simulink Model Interface** command. This allows more than 3000 leaf signals. Code generation performance is also improved.

Reading preferences You can read the value of specific settings or of all settings.

Saving preferences to a file You can save the preferences to a file. The current values are then available in the next MATLAB session.

For more information on the `dsmpb_pref()` API command and its parameters, refer to [dsmpb_pref \(Model Interface Package for Simulink API Reference\)](#).

Model Interface Package for Simulink Glossary

Introduction	The glossary briefly explains the most important expressions and naming conventions used in the Model Interface Package for Simulink documentation.
--------------	---

Where to go from here	Information in this section
-----------------------	-----------------------------

B.....	79
C.....	80
D.....	80
H.....	81
I.....	81
M.....	81
R.....	83
S.....	83
V.....	84

B

Behavior model A model that contains the control algorithm for a controller (function prototyping system) or the algorithm of the controlled system (hardware-in-the-loop system). It does not contain I/O functionality and access to the hardware.

You can add a Simulink behavior model directly to a ConfigurationDesk application. Alternatively, you can generate a [Simulink implementation](#)

[container](#) ¹ file (SIC file) from the Simulink model to add this file to a ConfigurationDesk application, or to import it into VEOS Player.

C

ConfigurationDesk A software tool for configuring and implementing real-time applications. There are two independent versions:

- ConfigurationDesk
- ConfigurationDesk for RapidPro

Models created with the [Model Interface Package for Simulink](#) ¹ can be used with ConfigurationDesk. ConfigurationDesk lets you add Simulink behavior models or SIC files to ConfigurationDesk projects and map them to I/O functions. You can implement real-time applications for the ConfigurationDesk projects and execute them on dSPACE real-time hardware.

ConfigurationDesk model interface The part of the [model interface](#) ¹ that is available in ConfigurationDesk. This specific term is used to explicitly distinguish between the model interface in ConfigurationDesk and the model interface in Simulink.

D

Data Inport block A model port block of the [Model Interface Blockset](#) ¹ that provides one or more data inports to receive data from the ports of another model implementation (ConfigurationDesk and VEOS Player) or from function ports (ConfigurationDesk only). **Data Inport** blocks specify data interfaces that are available in ConfigurationDesk or in VEOS Player for further use. For this purpose, a graphical representation is created for them in the respective tool during the import.

Data Output block A model port block of the [Model Interface Blockset](#) ¹ that provides one or more data outputs to send data to the ports of another model implementation (ConfigurationDesk and VEOS Player) or to function ports (ConfigurationDesk only). **Data Output** blocks specify data interfaces that are available in ConfigurationDesk or in VEOS Player for further use. For this purpose, a graphical representation is created for them in the respective tool during the import.

H

Hardware-Triggered Runnable Function block A [model port block](#) that lets you execute parts of a model asynchronously and in a separate task. For this purpose, a Hardware-Triggered Runnable Function block exports a function-call subsystem in the Simulink behavior model as a [runnable function](#). In ConfigurationDesk, you can then create a task from the runnable function and an I/O event.

I

Interface model A temporary Simulink model created with ConfigurationDesk, that contains model port blocks from the [Model Interface Blockset](#). ConfigurationDesk initiates the creation of an interface model in Simulink. You can copy the blocks with their identities from the interface model and paste them into an existing Simulink behavior model. You can use the newly created interface model to create a new Simulink behavior model.

Inverse model port block A model port block that was created as a suitable counterpart for a selected model port block. Its purpose is setting up model communication. An inverse model port block has the same configuration, for example, the same port structure and port names, but the reverse data direction of the model port block from which it was created.

Inverse model port blocks cannot be created from Hardware-Triggered Runnable Function blocks or Software-Triggered Runnable Function blocks.

M

Model interface The part of the Simulink behavior model that is created using the [model port blocks](#) from the Model Interface Blockset. The model interface is used to connect the Simulink model with ConfigurationDesk or VEOS Player.

- If you connect the Simulink behavior model with ConfigurationDesk:
If you add a Simulink model containing model port blocks to a ConfigurationDesk application, the model port blocks are a part of the model topology in ConfigurationDesk and can be used as an interface to the behavior model. Model port blocks can also be created in ConfigurationDesk. They can then be exported to a Simulink model and thus create or update the model interface of the Simulink model.

- If you connect the Simulink behavior model with VEOS Player:
The model port blocks of the Simulink model interface are provided via a [Simulink implementation container](#) file (SIC file). The model port blocks correspond to VPU port groups in VEOS Player.

Model Interface Blockset A library of the Model Interface Package for Simulink that contains blocks to implement communication between a Simulink model and ConfigurationDesk or VEOS Player.

The Model Interface Blockset provides the following blocks:

- [Data Inport block](#)
- [Data Outport block](#)
- [Hardware-Triggered Runnable Function block](#)
- [Software-Triggered Runnable Function block](#)
- [Model Separation Setup block](#)

Model Interface Package for Simulink A dSPACE software product that lets you specify the interface of a Simulink behavior model that you can directly use in ConfigurationDesk. For the Simulink behavior model, you can also create a [Simulink implementation container](#) file (SIC file) that you can use in ConfigurationDesk and VEOS Player.

Model port An element of a [model port block](#). There are the following types of model ports:

- Data inports:
Data inports receive data from other model ports (ConfigurationDesk or VEOS Player), or from function ports (ConfigurationDesk only).
- Data outports:
Data outports send data to other model ports (ConfigurationDesk or VEOS Player), or to function ports (ConfigurationDesk only).
- Runnable function ports:
Runnable function ports are provided by [Hardware-Triggered Runnable Function blocks](#) or [Software-Triggered Runnable Function blocks](#). A runnable function port can be connected to a function-call subsystem that must be executed asynchronously in a separate task.

Model port block A block of the Model Interface Blockset. A model port block is part of the [Simulink model interface](#). Model port blocks specify data interfaces or export [runnable functions](#). There are the following types of model port blocks:

- [Data Inport block](#) and [Data Outport block](#)
In ConfigurationDesk, Data Inport blocks and Data Outport blocks correspond to model port blocks in the ConfigurationDesk model interface.
- [Hardware-Triggered Runnable Function block](#)
Hardware-Triggered Runnable Function blocks are available as runnable function blocks for modeling asynchronous tasks (ConfigurationDesk only).
- [Software-Triggered Runnable Function block](#)
Software-Triggered Runnable Function blocks are available as predefined tasks with an assigned software event (ConfigurationDesk and VEOS Player).

Model Separation Setup block A block of the Model Interface Blockset that opens the **Model Separation Setup**. The **Model Separation Setup** is used to separate individual models from an overall model in MATLAB/Simulink. Additionally, model separation starts the generation of a model communication description file (MCD file) that contains information on the separated models and their connections. You can use this MCD file in ConfigurationDesk.

R

Runnable function A function that must be called within a task to compute results. The runnable functions provided by a Simulink behavior model result either from different sample rates specified in the behavior model, or from the [Hardware-Triggered Runnable Function block](#), which exports a function-call subsystem as a runnable function. In ConfigurationDesk, runnable functions are available after an analysis of the Simulink behavior model or after the import of an SIC file. In VEOS Player, the tasks in which the runnable functions are executed are available as measurement rasters in an A2L file.


S

Simulink implementation container A container that contains the model code of a Simulink behavior model. The Model Interface Package for Simulink lets you generate a Simulink implementation container file for a behavior model. The Simulink implementation container file can be used in ConfigurationDesk or in VEOS Player. A MATLAB installation is not required for this purpose. You can also use SIC files in ConfigurationDesk or VEOS Player that were created with an earlier dSPACE Release. This simplifies the reuse of the behavior model code. The file name extension of a Simulink implementation container is SIC.

Simulink model interface The part of the [model interface](#) that is available in the Simulink behavior model. This specific term is used to explicitly distinguish between the model interface in ConfigurationDesk and the model interface in Simulink.

Software-Triggered Runnable Function block A [model port block](#) that lets you execute parts of a model asynchronously and in a separate task. The Software-Triggered Runnable Function block exports the runnable function from the function-call subsystem as a predefined task with an assigned software event. The predefined task can be used in ConfigurationDesk or VEOS Player.

V

VEOS Player A software tool for configuring and implementing offline simulation applications on a VEOS system. [Simulink implementation container files](#)  (SIC files) can be used with VEOS Player.

C

commands

- Add Model to ConfigurationDesk Project:
<ConfigurationDesk Project Name> 63
- Analyze Model in ConfigurationDesk
(Including Task Information) 64
- Analyze Model in ConfigurationDesk (Model
Interface Only) 66
- Create ConfigurationDesk Project From
Model 67
- Create Data Inport Block from Bus Creator
Block 48
- Create Data Output Block from Bus Creator
Block 52
- Delete Selection and Connected Blocks in
ConfigurationDesk 68
- Open a ConfigurationDesk Project 69
- Open Model Interface Blockset 57
- Open Model Separation Setup 58
- Paste and Keep IDs 58
- Save ConfigurationDesk Project 69
- Show Connected Block in
ConfigurationDesk 70
- Show in ConfigurationDesk 71
- Start ConfigurationDesk Build 71
- Update Selected Output Blocks from Input
Signals 60

Common Program Data folder 6

D

Data Inport block

- Block Configuration page 16
- Block Connections page 17
- ConfigurationDesk page 10
- overview 9
- Signal Configuration page 12

Data Output block

- Block Configuration page 24
- Block Connections page 25
- overview 18
- Signal Configuration page 21

Documents folder 6

DSFINISH.M file 76

dsmplib 7

DSPOSTSTARTUP.M file 75

DSSTARTUP.M file 74

F

files

- DSFINISH.M 76
- DSPOSTSTARTUP.M 75
- DSSTARTUP.M 74
- STARTUP.M 73

H

Hardware-Triggered Runnable Function block

- Block configuration page 31
- Block Connections page 32

overview 26

Runnable Function page 29

hook functions

model separation 43

L

Local Program Data folder 6

M

Model Interface Blockset 7

model separation

hook functions 43

Model Separation Setup block 39

overview 37

S

Software-Triggered Runnable Function block

Block configuration page 35

overview 33

Runnable Function page 34

STARTUP.M file 73

