RTI FPGA Programming Blockset

Script Interface Reference

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 and the contents hereof are subject to change without notice.

Contents

About This Reference

Content

This reference provides detailed information on the script interface provided by the RTI FPGA Programming Blockset.

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.

Introduction to the Script Interface of the RTI FPGA Programming Blockset

Introduction

The RTI FPGA Programming Blockset comes with a script interface with several script functions. You can use the script functions to execute different work steps when you implement a custom FPGA application.

Where to go from here

Information in this section

Basics on the Script Functions

Use

You can use the script functions provided by the script interface, for example, to execute scheduled build processes at night or to automate recurrent processes when you implement FPGA applications. When you handcode an FPGA application, the script interface initializes the handcode framework and creates the FPGA model INI file with the FPGA bitstream.

You use the script interface by entering its syntax directly in the command line of the MATLAB Command Window or by referencing an M file.

Basic syntax

The script interface is a MATLAB function provided by the RTI FPGA Programming Blockset with the following syntax:

```
[ReturnValue] = rtifpga_scriptinterface('Function',Parameter 1,
Parameter 2, ... )
```

ReturnValue Depending on the specified function, the script interface returns a cell array.

'Function' Lets you specify the function of the script interface to be used. For an overview of the available functions, refer to Overview on the Available Script Functions on page 8.

Parameter 1 ... n Depending on the specified function, you have to provide one or several parameters to the script interface. For more information, refer to the description of the script function.

Overview on the Available Script Functions

Introduction

The following overviews give you a quick introduction to the available script functions.

The script functions are grouped by the supported interface libraries:

- Script functions supporting the FPGA interface sublibrary on page 8
- Script functions supporting the processor interface sublibrary on page 9
- Script functions supporting the handcode interface on page 10

Script functions supporting the FPGA interface sublibrary

If you use the RTI blocks of the FPGA Interface sublibrary, you can use the following script functions to execute work steps that are related to this sublibrary.

Script Function	Purpose
AnalyzeFPGAXDATAWriteBus	To trigger the bus analyze of the Simulink bus that is connected to a Buffer64 Out FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.
CopyFPGAXDATAReadBus	To copy an existing FPGA bus topology from a specified Simulink Bus Creator block, subsystem inport block, or subsystem outport block to the Data port of the Buffer64 In FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.
ExecuteFPGAModelAction	To perform an action on the FPGA model (FPGA build, timing analysis, or HDL simulation).
ExportToNewProject	To add a new project to ConfigurationDesk and export the last build result to it.
ExportToRecentProject	To open the recent ConfigurationDesk project and export the last build result to it. If no recent project is available, the script adds a new one.

Script Function	Purpose
FPGABuild	To start the FPGA build process for a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.
FPGAFrameworkUpdate	To reset the block parameters to their initial values or to update the framework.
FPGAGetAllGuiBlockParameters	To get all editable parameters and their valid value ranges in the dialog of a function block.
FPGAGetBlockParameter	To get the value of a parameter in the dialog of a function block.
FPGASetBlockParameter	To set one or several parameters in the dialog of a function block.
GetFPGABlocks	To get all Simulink block handles of the specified block type.
GetFPGAXDATABusSettings	To get the bus settings of a **Buffer64 In**/**Buffer64 Out** FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.
GetMCSubsystems	To get the Simulink subsystems that provide the access for the processor cores of a multicore system. This script function supports only MicroAutoBox III and SCALEXIO system models.
GetTraceSubsystems	To get the Simulink subsystems that are traceable with your experiment software.
PressHelpButton	To open dSPACE Help to show the description of a certain RTI FPGA Programming Blockset function block.
ResetFPGAXDATABus	To clear the bus settings of a **Buffer64 In** FPGA block that use the bus transfer mode. This script function supports only MicroAutoBox III and SCALEXIO system models.
SetFPGAXDATABusSettings	To set the bus settings of a **Buffer64 In**/**Buffer64 Out** FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.
SetMCSubsystems	To set the Simulink subsystems that provide the access for the processor cores of a multicore system. This script function supports only MicroAutoBox III and SCALEXIO system models.
SetTraceSubsystems	To set the Simulink subsystems that are traceable with your experiment software.

For a description of the script functions, refer to Script Functions Supporting the FPGA Interface Sublibrary on page 11.

Script functions supporting the processor interface sublibrary

If you use the RTI blocks of the **Processor Interface** sublibrary, you can use the following script functions to execute work steps that are related to this sublibrary.

Script Function	Purpose
AddIniFile	To add an FPGA model INI file to the **PROC_SETUP_BL** block of a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.
AddProcIfBlock	To add an existing processor interface block to a processor model.
CreateBurnApplication	To create a burn application. This script function supports only MicroAutoBox II and PHS-bus-based system models.

Script Function	Purpose
Deseparation	To combine the processor model with the FPGA model after you separate the FPGA model. This script function supports only MicroAutoBox II, MicroLabBox, or PHS-bus-based system models.
GenerateProcessorInterface	To generate the processor interface of a MicroAutoBox III or SCALEXIO system.
GenerateProcInterfaceBlocks	To generate the processor interface of a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.
GetFPGABlocks	To get all Simulink block handles of the specified block type.
MigrateToModelPortBlocks	To migrate processor interface blocks of the Processor Interface sublibrary to model port blocks of the Model Interface Package for Simulink. This script function supports only MicroAutoBox III and SCALEXIO system models.
RemoveIniFile	To remove an FPGA model INI file from the **PROC_SETUP_BL** block of a MicroAutoBox II, MicroLabBox, or a PHS-bus-based system.
Separation	To separate the processor model from the FPGA model before you start the build process. This script function supports MicroAutoBox II, MicroLabBox, or PHS-bus-based system models.
ShowProcessorInterface	To see where the corresponding processor interface block of the selected FPGA interface block is used in the processor model.

For a description of the script functions, refer to Script Functions Supporting the Processor Interface Sublibrary on page 35.

Script functions supporting the handcode interface

If you use the handcode interface of the RTI FPGA Programming Blockset, the following script functions are required to handcode the FPGA application.

Script Function	Purpose
HCFPGAFrameworkInit	To initialize the handcode FPGA Framework INI file with the specified I/O interface.
HCFPGAModelIniGenerate	To create a new application-specific FPGA model INI file that includes the FPGA bitstream.

For a description of the script functions, refer to Script Functions Supporting the Handcode Interface on page 45.

Script Functions Supporting the FPGA Interface Sublibrary

Introduction

If you use the RTI blocks of the **FPGA Interface** sublibrary, you can use the following script functions to execute work steps that are related to this sublibrary.

Where to go from here

Information in this section

AnalyzeFPGAXDATAWriteBus

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To trigger the bus analyze of the Simulink bus that is connected to a Buffer64 Out FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.

Description

If triggered, the RTI FPGA Programming Blockset analyzes the connected Simulink bus and sets the Data port of the selected Buffer64 Out block to a matching bus topology.

Parameters

Function Lets you specify the function of the script interface to be used. To trigger the bus analyze, you set the parameter to `'AnalyzeFPGAXDATAWriteBus'`.

BlockHandle Lets you specify the Simulink handle of the Buffer64 Out block whose connected Simulink bus you want to analyze.

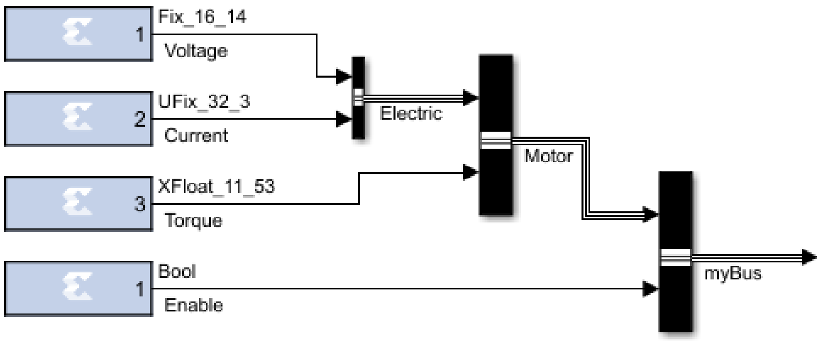
Return value

The structure that represents the analyzed Simulink bus.

The structure has the following fields:

- `name`: Cell array of signals.
  - `name` starts at the signal of the lowest hierarchy level.
  - Every signal has cell arrays for every bus hierarchy level.
  - If a signal has lower hierarchy levels, the cell array is prefixed by empty cells.
- `width`: Vector with the bit widths of the signals.
- `arith_type`: Cell array with the data formats of the signals. The signals can have the following data formats:
  - Signed (2's complement)
  - Unsigned
  - Floating-point
  - Boolean
- `bin_pt`: Vector with the binary point position of the signals.

The following example shows the signals of a Simulink bus and the structure that represents this bus.



```
signals =
  struct with fields:
          name: {{1×4 cell}  {1×4 cell}  {1×4 cell}  {1×4 cell}}
         width: [16 32 64 1]
    arith_type: {'Signed  (2's comp)'  'Unsigned'  'Floating-point'  'Boolean'}
        bin_pt: [14 3 53 0]
```

```
>> signals.name{:}
ans =
  1×4 cell array
    {'Voltage'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {'Current'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
 1×4 cell array
    {0×0 char}     {'Torque'}      {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {0×0 char}     {0×0 char}      {'Enable'}   {'myBus'}
```

Example

This example demonstrates how to analyze the Simulink bus of the most recently selected **Buffer64 Out** block.

```
ReturnValue = rtifpga_scriptinterface('AnalyzeFPGAXDATAWriteBus', gcb)
```

Related topics

Basics

Using Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 📖)

References

GetFPGAXDATABusSettings..26

CopyFPGAXDATAReadBus

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To copy an existing FPGA bus topology from a specified Simulink **Bus Creator** block, subsystem inport block, or subsystem outport block to the **Data** port of the **Buffer64 In** FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.

Parameters

Function Lets you specify the function of the script interface to be used. To copy an FPGA bus topology from a Simulink block, you set the parameter to `'CopyFPGAXDATAReadBus'`.

BlockHandle Lets you specify the Simulink handle of the **Buffer64 In** block whose **Data** port you want to configure.

CopyBlockHandle Lets you specify the Simulink handle of the Simulink **Bus Creator** block, subsystem inport block, or subsystem outport block whose bus topology you want to copy.

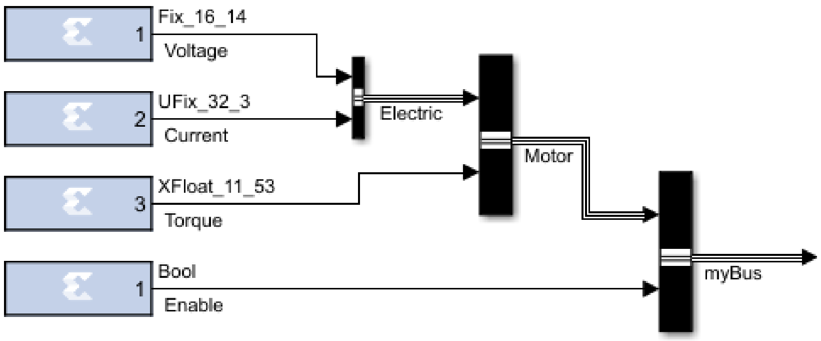
Return value

The structure that represents the bus topology copied to the **Buffer64 In** block.

The structure has the following fields:

- `name`: Cell array of signals.
  - `name` starts at the signal of the lowest hierarchy level.
  - Every signal has cell arrays for every bus hierarchy level.
  - If a signal has lower hierarchy levels, the cell array is prefixed by empty cells.
- `width`: Vector with the bit widths of the signals.
- `arith_type`: Cell array with the data formats of the signals. The signals can have the following data formats:
  - Signed (2's complement)
  - Unsigned
  - Floating-point
  - Boolean
- `bin_pt`: Vector with the binary point position of the signals.

The following example shows the signals of a Simulink bus and the structure that represents this bus.



```
signals =
  struct with fields:
          name: {{1×4 cell}  {1×4 cell}  {1×4 cell}  {1×4 cell}}
         width: [16 32 64 1]
    arith_type: {'Signed  (2's comp)'  'Unsigned'  'Floating-point'  'Boolean'}
        bin_pt: [14 3 53 0]
```

```
>> signals.name{:}
ans =
  1×4 cell array
    {'Voltage'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {'Current'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
 1×4 cell array
    {0×0 char}     {'Torque'}      {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {0×0 char}     {0×0 char}      {'Enable'}   {'myBus'}
```

Example

This example demonstrates how to copy the bus topology of a Bus Creator block to the most recently selected Buffer64 In block.

```
CopyBlockHandle = get_param('Bus Creator','handle')
ReturnValue = rtifpga_scriptinterface('CopyFPGAXDATAReadBus',gcb,CopyBlockHandle)
```

Related topics

Basics

Using Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 📖)

References

SetFPGAXDATABusSettings.. 30

ExecuteFPGAModelAction

Syntax

```
rtifpga_scriptinterface('Function',SetupBlockHandle,'ModelAction')
```

Purpose

To perform an action on the FPGA model (FPGA build, timing analysis, or HDL simulation).

Parameters

Function Lets you specify the function of the script interface to be used. To perform an action on the FPGA model, you set the parameter to `'ExecuteFPGAModelAction'`.

SetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

ModelAction Lets you specify the action you want to perform. Possible settings:

- `'FPGA Build'`

  Starts the build process for the FPGA model. See the MATLAB Command Window to follow the progress of the build process. As the result, you can find the FPGA model INI file in `<FPGABuildFolder>/<ModelName>_rtiFPGA/ini`. The name of a FPGA model INI file is `<FPGAApplicationName>_<ApplicationID>.ini`.

  If the FPGA application provides traceable signals, the build process opens a temporary model `<FPGAModelName>_rtiFPGAtmp` to build the application. The build process closes the temporary model at the end.

- `'Timing Analysis'`

  If the build process has detected timing problems, you can execute the timing analysis to analyze the timing behavior of your FPGA subsystem. This timing analysis considers the custom FPGA model without the framework of the platform that is automatically added to your FPGA model during the build process. Timing Analysis starts the timing analysis utility from the Xilinx System Generator. For more information, refer to the *System Generator for DSP User Guide* from Xilinx.

- `'HDL Simulation'`

  Starts the offline simulation of the entire Simulink model based on generated HDL code. The simulation itself uses utilities from the Xilinx Blockset to create an HDL test bench. The test bench compares Simulink simulation results for the FPGA subsystem to those of the Xilinx Vivado Simulator based on the generated HDL code for the FPGA subsystem and therefore checks the correctness of the generated code. The simulation results are presented as text and waveform. For more information, refer to the *System Generator for DSP User Guide* from Xilinx.

Return value

No return value.

Example

This example demonstrates how to start the build process of the DemoFPGApipt1 model.

```
SetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
rtifpga_scriptinterface('ExecuteFPGAModelAction',SetupBlockHandle,'FPGA Build')
```

ExportToNewProject

Syntax

```
rtifpga_scriptinterface('Function',
FPGASetupBlockHandle,'ConfigurationDeskProjectName')
```

Purpose

To add a new project to ConfigurationDesk and export the last build result to it.

Description

The script performs the following steps:

- Starts ConfigurationDesk.
- Creates a new project in the root folder of ConfigurationDesk.
- Exports the build result of the current FPGA application as a custom function block type to the project.

  If the entire model contains other subsystems with FPGA models for other FPGA boards, the framework also exports the build results of these FPGA models.
- Adds instances of the added custom function block types to the signal chain.

The script performs the following steps only if the processor model can be separated and the processor interface is implemented:

- Separates and saves the processor models according to the settings of the Model Separation Setup block.

  In ConfigurationDesk, the processor models are implemented outside ConfigurationDesk in Simulink models.
- Exports the model interface of the processor models to ConfigurationDesk.
- Maps the function ports of the added custom functions to the model ports of the processor models (behavior model).

Parameters

Function Lets you specify the function of the script interface to be used. To add a new project to ConfigurationDesk, you set the parameter to `'ExportToNewProject'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

ConfigurationDeskProjectName Lets you specify the name of the new ConfigurationDesk project to be created.

Return value

No return value.

Example

This example demonstrates how to add a new 'New_Project' project to ConfigurationDesk and to export the build result of the DemoFPGApipt1 FPGA application to the new created project.

```
FPGASetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
rtifpga_scriptinterface('ExportToNewProject',FPGASetupBlockHandle,'New_Project')
```

Related topics

HowTos

How to Export Build Results and Processor Models to ConfigurationDesk Projects (RTI FPGA Programming Blockset Guide 📖)

ExportToRecentProject

Syntax

```
rtifpga_scriptinterface('Function',
FPGASetupBlockHandle,'ConfigurationDeskProjectName')
```

Purpose

To open the recent ConfigurationDesk project and export the last build result to it. If no recent project is available, the script adds a new one.

Description

The recent project is the project that is selected on the ConfigurationDesk Interface page of the FPGA_SETUP_BL block dialog.

The script performs the following steps:

- Starts ConfigurationDesk.
- Opens a project in the root folder of ConfigurationDesk.
- Exports the build result of the current FPGA application as a custom function block type to the project.

  If the entire model contains other subsystems with FPGA models for other FPGA boards, the framework also exports the build results of these FPGA models.
- Adds instances of the added custom function block types to the signal chain.

The script performs the following steps only if the processor model can be separated and the processor interface is implemented:

- Separates and saves the processor models according to the settings of the Model Separation Setup block.

  In ConfigurationDesk, the processor models are implemented outside ConfigurationDesk in Simulink models.

- Exports the model interface of the processor models to ConfigurationDesk.
- Maps the function ports of the added custom functions to the model ports of the processor models (behavior model).

Parameters

Function Lets you specify the function of the script interface to be used. To open an existing ConfigurationDesk project and export the last build result to it, you set the parameter to `'ExportToRecentProject'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

ConfigurationDeskProjectName Lets you specify the name of an existing ConfigurationDesk project to export the FPGA application to it.

Return value

No return value.

Example

This example demonstrates how to to export the build result of the DemoFPGApipt1 FPGA application to the 'Recent_Project' in ConfigurationDesk.

```
FPGASetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
rtifpga_scriptinterface('ExportToRecentProject',FPGASetupBlockHandle,'Recent_Project')
```

Related topics

HowTos

How to Export Build Results and Processor Models to ConfigurationDesk Projects (RTI FPGA Programming Blockset Guide 📖)

FPGABuild

Syntax

```
[ListOfAppIDs, ListOfiniFileRefs, ProcModelPath, Errorcode] =
rtifpga_scriptinterface('Function',ProcModelHandle)
```

Purpose

To start the FPGA build process for a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.

Parameters

Function Lets you specify the function of the script interface to be used. To start the FPGA build process, you set the parameter to `'FPGABuild'`.

ProcModelHandle Lets you specify the Simulink handle of the the processor model. The script will start an FPGA build process for any FPGA subsystems found in the specified processor model.

Return value

ListOfAppIDs List of the ApplicationIDs of the generated FPGA model INI files.

ListOfiniFileRefs List of the generated FPGA model INI files including their path.

ProcModelPath Path of the processor model from where the script starts to build FPGA applications.

Errorcode

- 0: Operation finished successfully.
- Not 0: Operation terminated with an error.

Example

This example demonstrates how to start the FPGA build process of the FPGA models in the the processor model DemoFPGApipt1.

```
ProcModelHandle = get_param('DemoFPGApipt1','handle')
[ListOfAppIDs, ListOfiniFileRefs, ProcModelPath, Errorcode] = rtifpga_scriptinterface('FPGABuild',ProcModelHandle)
```

Related topics

HowTos

How to Build FPGA Applications (MicroAutoBox II, MicroLabBox, PHS-Bus-Based System) (RTI FPGA Programming Blockset Guide 🕮)

FPGAFrameworkUpdate

Syntax

```
rtifpga_scriptinterface('Function',ModelHandle,'Reinit')
```

Purpose

To reset the block parameters to their initial values or to update the framework.

Description

With this function you can manually update the framework or reset the block parameters.

If you use FPGA applications with a newer version of the RTI FPGA Programming Blockset, the framework will automatically update itself to the current framework version.

Parameters

Function Lets you specify the function of the script interface to be used. To update the framework, you set the parameter to `'FPGAFrameworkUpdate'`.

ModelHandle Lets you specify the Simulink handle of the model. The script updates all subsystems that are contained in the model.

Reinit Lets you reset the block parameters to their initial values. This parameter is optional. If you do not specify this parameter, the block parameters will be unchanged.

Return value

No return value.

Example

This example demonstrates how to reset the block parameters to their initial values.

```
ModelHandle = get_param('DemoFPGApipt1','handle')
rtifpga_scriptinterface('FPGAFrameworkUpdate',ModelHandle,'Reinit')
```

FPGAGetAllGuiBlockParameters

Syntax

```
[[Parameter],[Range]] =
rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To get all editable parameters and their valid value ranges in the dialog of a function block.

Parameters

Function Lets you specify the function of the script interface to be used. To get all editable parameters and their valid value ranges, you set the parameter to `'FPGAGetAllGuiBlockParameters'`.

BlockHandle Lets you specify the Simulink handle of the function block whose parameters you want to read.

Return value

Parameter Array with the editable parameters with the current setting.

Range Array with the editable parameters with the valid value ranges.

Example

This example demonstrates how to get all editable parameters and their valid value ranges in the dialog of the FPGA_XDATA_READ_BL2 function block.

```
BlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_XDATA_READ_BL2','handle')
[Parameter,Range] = rtifpga_scriptinterface('FPGAGetAllGuiBlockParameters',BlockHandle)
```

Related topics

References

FPGAGetBlockParameter........ 23
FPGASetBlockParameter........ 24
GetFPGABlocks........ 25

FPGAGetBlockParameter

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
BlockHandle,'Parameter')
```

Purpose

To get the value of a parameter in the dialog of a function block.

Parameters

Function Lets you specify the function of the script interface to be used. To get the value of a parameter, you set the parameter to `'FPGAGetBlockParameter'`.

BlockHandle Lets you specify the Simulink handle of the function block whose parameters you want to read.

Parameter Lets you enter the name of the parameter from which you want to read the value.

Return value

Array with the name of the parameter and the read value.

Example

This example demonstrates how to read the value of the Format parameter of a FPGA_XDATA_READ_BL block.

```
BlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_XDATA_READ_BL2','handle')
ReturnValue = rtifpga_scriptinterface('FPGAGetBlockParameter',BlockHandle,'Format')
```

Related topics

References

FPGAGetAllGuiBlockParameters........ 22
FPGASetBlockParameter........ 24
GetFPGABlocks........ 25

FPGASetBlockParameter

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
BlockHandle,'Parameter','Value')
```

Purpose

To set one or several parameters in the dialog of a function block.

Parameters

Function Lets you specify the function of the script interface to be used. To set one or several parameters, you set the parameter to `'FPGASetBlockParameter'`.

BlockHandle Lets you specify the Simulink handle of the function block whose parameters you want to set.

Parameter Lets you specify the name of the parameter you want to set a new value. You can set more than one parameter by providing several parameter names in an array.

Tip

Use the script function `FPGAGetAllGuiBlockParameters` to get all editable parameters and their valid value ranges in the dialog of an function block.

Value Lets you specify the new value for the parameter in the dialog. If you set more than one parameter, you have to provide the new values in an array in the same sequence as you specify the parameters to be changed.

Return value

Array with the set values.

Examples

The following example demonstrates how to set the Format parameter of a FPGA_XDATA_READ_BL block to unsigned.

```
BlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_XDATA_READ_BL2','handle')
ReturnValue = rtifpga_scriptinterface('FPGASetBlockParameter',BlockHandle,'Format','unsigned')
```

The following example demonstrates how to set the Format and the Channel name parameters of a FPGA_XDATA_READ_BL block.

```
BlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_XDATA_READ_BL2','handle')
ReturnValue = rtifpga_scriptinterface('FPGASetBlockParameter',BlockHandle,{'Format','CustomName'},
{'signed','NewChannel'})
```

Related topics

References

FPGAGetAllGuiBlockParameters.. 22
FPGAGetBlockParameter.. 23
GetFPGABlocks.. 25

GetFPGABlocks

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
ModelHandle,'BlockType')
```

Purpose

To get all Simulink block handles of the specified block type.

Parameters

Function Lets you specify the function of the script interface to be used. To get the value of a parameter, you set the parameter to `'GetFPGABlocks'`.

ModelHandle Lets you specify the Simulink handle of the subsystem whose block handles you want to read.

BlockType Lets you enter the block type whose block handles you want to read.

The following block types are supported:

- `FPGASetupBlock`: FPGA_SETUP_BL block
- `FPGAXDATARead`: FPGA_XDATA_READ_BL block
- `FPGAXDATAWrite`: FPGA_XDATA_WRITE_BL block
- `FPGAIORead`: FPGA_IO_READ_BL block
- `FPGAIOWrite`: FPGA_IO_WRITE_BL block
- `FPGAInt`: FPGA_INT_BL block
- `PROCSetupBlock`: PROC_SETUP_BL block
- `PROCXDATARead`: PROC_XDATA_READ_BL block
- `PROCXDATAWrite`: PROC_XDATA_WRITE_BL block
- `PROCInt`: PROC_INT_BL block

Return value

Array with the Simulink handles of all blocks of the specified type.

Example

This example demonstrates how to get the Simulink handles of all FPGA_IO_READ_BL blocks in the specified model.

```
ModelHandle = get_param('DemoFPGApipt1/DemoFPGApipt1','handle')
ReturnValue = rtifpga_scriptinterface('GetFPGABlocks',ModelHandle,'FPGAIORead')
```

GetFPGAXDATABusSettings

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To get the bus settings of a Buffer64 In/Buffer64 Out FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.

Description

You cannot use a bus topology of the processor model in an FPGA model, because bus topologies of the processor model do not include the FPGA data types. For instructions, refer to How to Use Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 🕮).

Parameters

Function Lets you specify the function of the script interface to be used. To get the bus settings, you set the parameter to `'GetFPGAXDATABusSettings'`.

BlockHandle Lets you specify the Simulink handle of the Buffer64 In/Buffer64 Out block whose bus settings you want to get.

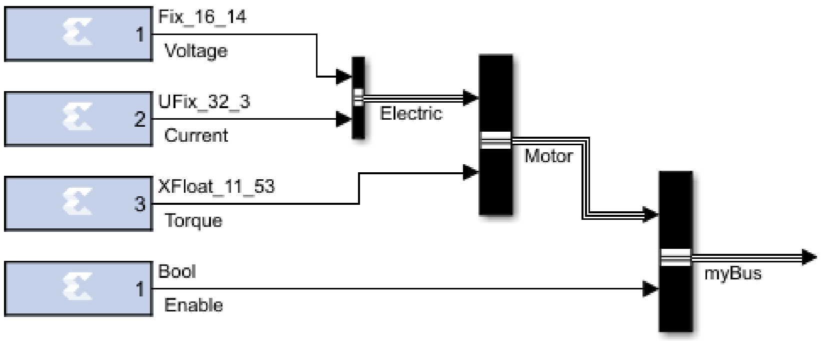
Return value

The structure that represents the read bus settings.

The structure has the following fields:

- `name`: Cell array of signals.
  - `name` starts at the signal of the lowest hierarchy level.
  - Every signal has cell arrays for every bus hierarchy level.
  - If a signal has lower hierarchy levels, the cell array is prefixed by empty cells.
- `width`: Vector with the bit widths of the signals.
- `arith_type`: Cell array with the data formats of the signals. The signals can have the following data formats:
  - Signed (2's complement)
  - Unsigned
  - Floating-point
  - Boolean
- `bin_pt`: Vector with the binary point position of the signals.

The following example shows the signals of a Simulink bus and the structure that represents this bus.



```
signals =
  struct with fields:
          name: {{1×4 cell}  {1×4 cell}  {1×4 cell}  {1×4 cell}}
         width: [16 32 64 1]
    arith_type: {'Signed  (2's comp)'  'Unsigned'  'Floating-point'  'Boolean'}
        bin_pt: [14 3 53 0]
```

```
>> signals.name{:}
ans =
  1×4 cell array
    {'Voltage'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {'Current'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
 1×4 cell array
    {0×0 char}     {'Torque'}      {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {0×0 char}     {0×0 char}      {'Enable'}   {'myBus'}
```

Example

This example demonstrates how to get the bus settings of the most recently selected **Buffer64 In**/**Buffer64 Out** FPGA block.

```
ReturnValue = rtifpga_scriptinterface('GetFPGAXDATABusSettings', gcb)
```

Related topics

Basics

Using Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 📖)

References

SetFPGAXDATABusSettings..30

GetMCSubsystems

Syntax

```
[ReturnValue] =
rtifpga_scriptinterface('Function',FPGASetupBlockHandle)
```

Purpose

To get the Simulink subsystems that provide the access for the processor cores of a multicore system. This script function supports only MicroAutoBox III and SCALEXIO system models.

Description

With this function you can get the subsystems that provide the processor interface to the different processor models of a multicore processor application. For more information, refer to Modeling FPGA Applications Supporting Multicore Processor Applications (RTI FPGA Programming Blockset Guide 🕮).

Parameters

Function Lets you specify the function of the script interface to be used. To get the subsystems for multicore support, you set the parameter to `'GetMCSubsystems'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

Return value

Array with the Simulink subsystems for multicore support.

Example

This example demonstrates how to get the subsystems with multicore support.

```
FPGASetupBlockHandle = get_param('.../FPGA_SETUP_BL1','handle')
ReturnValue = rtifpga_scriptinterface('GetMCSubsystems',FPGASetupBlockHandle)
```

Related topics

Basics

Aspects on FPGA Applications Supporting Multicore Processor Applications (RTI FPGA Programming Blockset Guide 🕮)

References

SetMCSubsystems.. 33

GetTraceSubsystems

Syntax

```
[ReturnValue] =
rtifpga_scriptinterface('Function',FPGASetupBlockHandle)
```

Purpose

To get the Simulink subsystems that are traceable with your experiment software.

Description

FPGA signal tracing is not supported by a .PHS-bus-based system

Parameters

Function Lets you specify the function of the script interface to be used. To get the subsystems that are traceable, you set the parameter to `'GetTraceSubsystems'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

Return value

Array with the Simulink subsystems that are traceable.

Example

This example demonstrates how to get the subsystems that are traceable.

```
FPGASetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
ReturnValue = rtifpga_scriptinterface('GetTraceSubsystems',FPGASetupBlockHandle)
```

Related topics

Basics

Basics on Tracing FPGA Signals (RTI FPGA Programming Blockset Guide 📖)

References

SetTraceSubsystems.. 34

PressHelpButton

Syntax

```
rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To open dSPACE Help to show the description of a certain RTI FPGA Programming Blockset function block.

Parameters

Function Lets you specify the function of the script interface to be used. To open dSPACE Help, you set the parameter to `'PressHelpButton'`.

BlockHandle Lets you specify the Simulink handle of the function block whose description you want to open.

Return value

No return value.

Example

This example demonstrates how to open dSPACE Help to show the description of the FPGA_XDATA_READ_BL2 function block in the DemoFPGApipt1 FPGA model.

```
BlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_XDATA_READ_BL2','handle')
rtifpga_scriptinterface('PressHelpButton',BlockHandle)
```

SetFPGAXDATABusSettings

Syntax

```
rtifpga_scriptinterface('Function',BlockHandle,Signals)
```

Purpose

To set the bus settings of a Buffer64 In/Buffer64 Out FPGA block. This script function supports only MicroAutoBox III and SCALEXIO system models.

Parameters

Function Lets you specify the function of the script interface to be used. To set the bus settings, you set the parameter to `'SetFPGAXDATABusSettings'`.

BlockHandle Lets you specify the Simulink handle of the Buffer64 In/Buffer64 Out block whose bus settings you want to set.

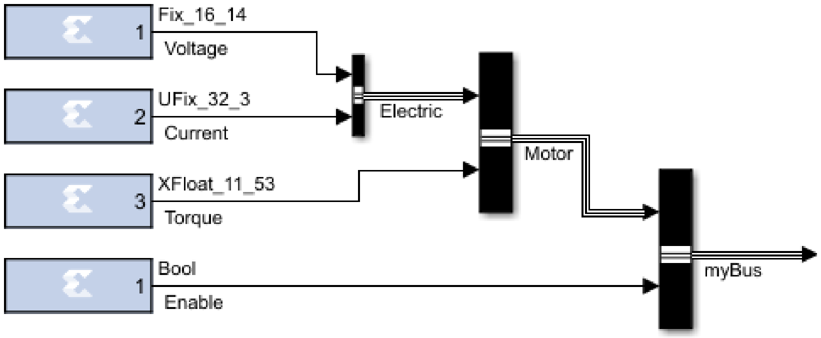
Signals Lets you specify a structure that represents the bus settings.

The structure has the following fields:

- `name`: Cell array of signals.
  - `name` starts at the signal of the lowest hierarchy level.
  - Every signal has cell arrays for every bus hierarchy level.
  - If a signal has lower hierarchy levels, the cell array is prefixed by empty cells.
- `width`: Vector with the bit widths of the signals.
- `arith_type`: Cell array with the data formats of the signals. The signals can have the following data formats:
  - Signed (2's complement)
  - Unsigned

  - Floating-point
  - Boolean
- `bin_pt`: Vector with the binary point position of the signals.

The following example shows the signals of a Simulink bus and the structure that represents this bus.



```
signals =
  struct with fields:
          name: {{1×4 cell}  {1×4 cell}  {1×4 cell}  {1×4 cell}}
         width: [16 32 64 1]
    arith_type: {'Signed  (2's comp)'  'Unsigned'  'Floating-point'  'Boolean'}
        bin_pt: [14 3 53 0]
```

```
>> signals.name{:}
ans =
  1×4 cell array
    {'Voltage'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {'Current'}    {'Electric'}    {'Motor'}    {'myBus'}
ans =
 1×4 cell array
    {0×0 char}     {'Torque'}      {'Motor'}    {'myBus'}
ans =
  1×4 cell array
    {0×0 char}     {0×0 char}      {'Enable'}   {'myBus'}
```

Example

This example demonstrates how to use the SetFPGAXDATABusSettings script function.

In the example, the bus settings of one block are copied to another block:

1. Get the bus settings of the most recently selected Buffer64 In/Buffer64 Out block of the FPGA model.

   ```
   Signals = rtifpga_scriptinterface('GetFPGAXDATABusSettings', gcb)
   ```

2. Click another Buffer64 In/Buffer64 Out block of the FPGA model and set its bus signals.

   ```
   rtifpga_scriptinterface('SetFPGAXDATABusSettings', gcb, Signals)
   ```

Related topics

Basics

Using Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 📖)

References

GetFPGAXDATABusSettings..26

ResetFPGAXDATABus

Syntax

```
rtifpga_scriptinterface('Function',BlockHandle)
```

Purpose

To clear the bus settings of a Buffer64 In FPGA block that use the bus transfer mode. This script function supports only MicroAutoBox III and SCALEXIO system models.

Parameters

Function Lets you specify the function of the script interface to be used. To clear the bus settings, you set the parameter to `'ResetFPGAXDATABus'`.

BlockHandle Lets you specify the Simulink handle of the Buffer64 In block whose Data port you want to clear.

Example

This example demonstrates how to clear the bus settings of the most recently selected Buffer64 In FPGA block.

```
rtifpga_scriptinterface('ResetFPGAXDATABus',gcb)
```

Related topics

Basics

Using Simulink Buses for Modeling the Processor Communication (RTI FPGA Programming Blockset Guide 📖)

References

AnalyzeFPGAXDATAWriteBus..13

SetMCSubsystems

Syntax

```
rtifpga_scriptinterface('Function',FPGASetupBlockHandle, Subsystems
on page 33)
```

Purpose

To set the Simulink subsystems that provide the access for the processor cores of a multicore system. This script function supports only MicroAutoBox III and SCALEXIO system models.

Description

With this function you can set the subsystems that provide the processor interface to the different processor models of a multicore processor application. For more information, refer to Modeling FPGA Applications Supporting Multicore Processor Applications (RTI FPGA Programming Blockset Guide 📖).

Parameters

Function Lets you specify the function of the script interface to be used. To set the subsystems for multicore support, you set the parameter to `'SetMCSubsystems'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

Subsystems Lets you specify an array of subsystems to provide a specific processor interface for each processor core.

Example

This example demonstrates how to set subsystems to support multicore systems.

```
FPGASetupBlockHandle = get_param('.../FPGA_SETUP_BL1','handle')
Subsystems = {'Core1Interface','Core2Interface'}
rtifpga_scriptinterface('SetMCSubsystems',FPGASetupBlockHandle,Subsystems)
```

Related topics

Basics

Aspects on FPGA Applications Supporting Multicore Processor Applications (RTI FPGA Programming Blockset Guide 📖)

References

GetMCSubsystems... 28

SetTraceSubsystems

Syntax

```
rtifpga_scriptinterface('Function',FPGASetupBlockHandle, )
```

Purpose

To set the Simulink subsystems that are traceable with your experiment software.

Description

With this function you make the input and output ports, all internal signals, and buses of the specified subsystems traceable. The script function replaces existing definitions for traceable subsystems.

FPGA signal tracing is not supported by a .PHS-bus-based system

Parameters

Function Lets you specify the function of the script interface to be used. To make subsystems traceable, you set the parameter to `'SetTraceSubsystems'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

Subsystems Lets you specify an array of subsystems that are traceable with your experiment software.

Example

This example demonstrates how to make subsystems traceable.

```
FPGASetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
Subsystems = {'Ki','Kp'}
rtifpga_scriptinterface('SetTraceSubsystems',FPGASetupBlockHandle,Subsystems)
```

Related topics

Basics

Basics on Tracing FPGA Signals (RTI FPGA Programming Blockset Guide 📖)

References

GetTraceSubsystems..29

Script Functions Supporting the Processor Interface Sublibrary

Introduction

If you use the RTI blocks of the Processor Interface sublibrary, you can use the following script functions to execute work steps that are related to this sublibrary.

Where to go from here

Information in this section

AddIniFile

Syntax

```
rtifpga_scriptinterface('Function',
ProcSetupBlockHandle,'PathName','FileName')
```

Purpose

To add an FPGA model INI file to the PROC_SETUP_BL block of a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.

Parameters

Function Lets you specify the function of the script interface to be used. To add an FPGA model INI file to the processor interface block, you set the parameter to `'AddIniFile'`.

ProcSetupBlockHandle Lets you specify the Simulink handle of the PROC_SETUP_BL block that you want to add the FGA model INI file.

PathName Lets you specify the path of the FPGA model INI file to be added.

FileName Lets you specify the FPGA model INI file to be added.

Return value

No return value.

Example

This example demonstrates how to add the `DemoFPGApipt1_<ApplicationID>.ini` file to the DemoFPGApipt1 processor model.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
rtifpga_scriptinterface('AddIniFile',ProcSetupBlockHandle,'<path>','DemoFPGApipt1_<ApplicationID>.ini')
```

Related topics

HowTos

How to Build Single-Core Processor Applications (MicroAutoBox II, MicroLabBox, PHS-Bus-Based System) (RTI FPGA Programming Blockset Guide 🕮)

AddProcIfBlock

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
'ProcessorModel',ProcInterfaceBlockHandle,Position)
```

Purpose

To add an existing processor interface block to a processor model.

Description

With this method, you can add the processor interface blocks that are generated with the GenerateProcInterfaceBlocks script function to a processor model.

Parameters

Function Lets you specify the function of the script interface to be used. To add an existing processor interface block, you set the parameter to `'AddProcIfBlock'`.

ProcessorModel Lets you specify the name of the target processor model.

ProcInterfaceBlockHandle Lets you specify the Simulink handle of the processor interface block that you want to add to the target processor model.

Position Lets you specify the position of the processor interface block in the processor model as a vector of coordinates in pixels. The origin is the upper-left corner of the processor model.

This parameter is optional.

Return value

Simulink handle of the added processor interface model in the target processor model.

Example

This example demonstrates how to add an generated interface block to the DemoFPGApipt1 processor model.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
FPGASetupBlockHandle = get_param('DemoFPGApipt1/FPGA/FPGA_SETUP_BL1','handle')
GenBlockHandles = rtifpga_scriptinterface('GenerateProcInterfaceBlocks',ProcSetupBlockHandle,FPGASetupBlockHandle,'1')
ReturnValue = rtifpga_scriptinterface('AddProcIfBlock','DemoFPGApipt1',GenBlockHandles(1))
```

Related topics

References

GenerateProcInterfaceBlocks... 40

CreateBurnApplication

Syntax

```
rtifpga_scriptinterface('Function',ProcSetupBlockHandle)
```

Purpose

To create a burn application. This script function supports only MicroAutoBox II and PHS-bus-based system models.

Description

The burn application can be used to download only the FPGA applications to the associated FPGAs according to the specified programming options.

Parameters

Function Lets you specify the function of the script interface to be used. To create a burn application, you set the parameter to `'CreateBurnApplication'`.

ProcSetupBlockHandle Lets you specify the Simulink handle of the PROC_SETUP_BL block of the processor model.

Return value

No return value.

Example

This example demonstrates how to create a burn application.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
rtifpga_scriptinterface('CreateBurnApplication',ProcSetupBlockHandle)
```

Related topics

HowTos

How to Create Burn Applications (RTI FPGA Programming Blockset Guide 📖)

Deseparation

Syntax

```
ReturnValue = rtifpga_scriptinterface('Function',ModelHandle)
```

Purpose

To combine the processor model with the FPGA model after you separate the FPGA model. This script function supports only MicroAutoBox II, MicroLabBox, or PHS-bus-based system models.

Parameters

Function Lets you specify the function of the script interface to be used. To combine the processor model with the separated FPGA model, you set the parameter to `'Deseparation'`.

ModelHandle Lets you specify the Simulink handle of the processor model.

Return value

- 0: Operation finished successfully.
- Not 0: Operation terminated with an error.

Example

This example demonstrates how to combine the processor model DemoFPGApipt1 with the separated FPGA model.

```
ModelHandle = get_param('DemoFPGApipt1','handle')
ReturnValue = rtifpga_scriptinterface('Deseparation',ModelHandle)
```

Related topics

HowTos

How to Build Single-Core Processor Applications (MicroAutoBox II, MicroLabBox, PHS-Bus-Based System) (RTI FPGA Programming Blockset Guide 📖)

GenerateProcessorInterface

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
FPGASetupBlockHandle)
```

Purpose

To generate the processor interface of a MicroAutoBox III or SCALEXIO system.

Parameters

Function Lets you specify the function of the script interface to be used. To generate the processor interface, you set the parameter to `'GenerateProcessorInterface'`.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

Return value

Array with the block handles of the generated processor interface blocks.

Example

This example demonstrates how to generate the processor interface blocks.

```
FPGASetupBlockHandle = get_param('DemoFPGApipt1/DemoFPGApipt1/FPGA_SETUP_BL1','handle')
GenBlockHandles = rtifpga_scriptinterface('GenerateProcessorInterface',FPGASetupBlockHandle)
```

Related topics

HowTos

How to Generate a Processor Interface (RTI FPGA Programming Blockset Guide 📖)

References

GenerateProcInterfaceBlocks.. 40

GenerateProcInterfaceBlocks

Syntax

```
[ReturnValue] = rtifpga_scriptinterface('Function',
ProcSetupBlockHandle,FPGASetupBlockHandle,'BoardNo')
```

Purpose

To generate the processor interface of a MicroAutoBox II, MicroLabBox, or PHS-bus-based system.

Parameters

Function Lets you specify the function of the script interface to be used. To generate the processor interface, you set the parameter to `'GenerateProcInterfaceBlocks'`.

ProcSetupBlockHandle Lets you specify the Simulink handle of the Proc_SETUP_BL block of the processor model.

FPGASetupBlockHandle Lets you specify the Simulink handle of the FPGA_SETUP_BL block of the FPGA model.

BoardNo Lets you specify the FPGA board number.

Return value

Array with the block handles of the generated processor interface blocks.

Example

This example demonstrates how to generate the processor interface blocks.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
FPGASetupBlockHandle = get_param('DemoFPGApipt1/FPGA/FPGA_SETUP_BL1','handle')
GenBlockHandles =
rtifpga_scriptinterface('GenerateProcInterfaceBlocks'Error_Scriptinterfac,ProcSetupBlockHandle,FPGASetupBlockHandle,'1')
```

Related topics

HowTos

How to Generate a Processor Interface (RTI FPGA Programming Blockset Guide 🕮)

References

AddProcIfBlock..37
GenerateProcessorInterface...39

GetFPGABlocks

Purpose

To get all Simulink block handles of the specified block type.

Description

For more information, refer to GetFPGABlocks on page 25.

MigrateToModelPortBlocks

Syntax

```
rtifpga_scriptinterface('Function','Model')
```

Purpose

To migrate processor interface blocks of the Processor Interface sublibrary to model port blocks of the Model Interface Package for Simulink. This script function supports only MicroAutoBox III and SCALEXIO system models.

Description

With this method you can manually migrate processor models to use the model port blocks of the Model Interface Package for Simulink. The migration is a forced migration without conditions.

Before you migrate the processor model, ensure that the following requirements on processor models are fulfilled:

- The processor model is for a MicroAutoBox III or a SCALEXIO system.
- The processor model is open.

Parameters

Function Lets you specify the function of the script interface to be used. To migrate the processor model, you set the parameter to `'MigrateToModelPortBlocks'`.

Model Lets you specify the processor model to be migrated.

Return value

No return value.

Example

This example demonstrates how to migrate the `Example.slx` processor model.

```
rtifpga_scriptinterface('MigrateToModelPortBlocks','Example')
```

RemoveIniFile

Syntax

```
rtifpga_scriptinterface('Function',
,ProcSetupBlockHandle,'Mode','FileName')
```

Purpose

To remove an FPGA model INI file from the PROC_SETUP_BL block of a MicroAutoBox II, MicroLabBox, or a PHS-bus-based system.

Parameters

Function Lets you specify the function of the script interface to be used. To remove an FPGA model INI file, you set the parameter to `'RemoveIniFile'`.

ProcSetupBlockHandle Lets you specify the Simulink handle of the PROC_SETUP_BL block you want to remove the FGA model INI file from.

Mode Lets you specify the mode to remove the FPGA model INI files. If you specify `'single'`, a single FPGA model will be removed. If you specify `'all'`, all FPGA model INI files will be removed.

FileName Lets you specify the FPGA model INI file to be removed. If you remove all FPGA model INI files, this parameter is void.

Return value

No return value.

Examples

This example demonstrates how to remove the `DemoFPGApipt1_<ApplicationID>.ini` file from the DemoFPGApipt1 processor model.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
rtifpga_scriptinterface('RemoveIniFile',ProcSetupBlockHandle,'single','DemoFPGApipt1_<ApplicationID>.ini')
```

This example demonstrates how to remove all FPGA model INI files from the DemoFPGApipt1 processor model.

```
ProcSetupBlockHandle = get_param('DemoFPGApipt1/PROC_SETUP_BL1','handle')
rtifpga_scriptinterface('RemoveIniFile',ProcSetupBlockHandle,'all')
```

Related topics

HowTos

How to Build Single-Core Processor Applications (MicroAutoBox II, MicroLabBox, PHS-Bus-Based System) (RTI FPGA Programming Blockset Guide 🕮)

Separation

Syntax

```
[Errorcode, Path] = rtifpga_scriptinterface('Function',ModelHandle)
```

Purpose

To separate the processor model from the FPGA model before you start the build process. This script function supports MicroAutoBox II, MicroLabBox, or PHS-bus-based system models.

Parameters

Function Lets you specify the function of the script interface to be used. To separate the processor model before you start the build process, you set the parameter to `'Separation'`.

ModelHandle Lets you specify the Simulink handle of the processor model.

Return value

Errorcode

- 0: Operation finished successfully.
- Not 0: Operation terminated with an error.

Path Path to the separated file.

Example

This example demonstrates how to separate the processor model DemoFPGApipt1 from the FPGA model.

```
ModelHandle = get_param('DemoFPGApipt1','handle')
[errorcode, path] = rtifpga_scriptinterface('Separation',ModelHandle)
```

Related topics

HowTos

How to Build Single-Core Processor Applications (MicroAutoBox II, MicroLabBox, PHS-Bus-Based System) (RTI FPGA Programming Blockset Guide 🕮)

ShowProcessorInterface

Syntax

```
rtifpga_scriptinterface('Function',InterfaceBlockHandle)
```

Purpose

To see where the corresponding processor interface block of the selected FPGA interface block is used in the processor model.

Parameters

Function Lets you specify the function of the script interface to be used. To show the corresponding processor interface block, you set the parameter to `'ShowProcessorInterface'`.

InterfaceBlockHandle Lets you specify the Simulink handle of the FPGA_XDATA_READ_BL or FPGA_XDATA_WRITE_BL block.

Return value

No return value.

Example

This example demonstrates how to show the processor interface block of the FPGA_XDATA_READ_BL1 interface block.

```
InterfaceBlockHandle = get_param('DemoFPGApipt1/FPGA/FPGA_XDATA_READ_BL1','handle')
rtifpga_scriptinterface('ShowProcessorInterface',InterfaceBlockHandle)
```

Script Functions Supporting the Handcode Interface

Introduction

If you use the handcode interface of the RTI FPGA Programming Blockset, the following script functions are required to handcode the FPGA application.

Where to go from here

Information in this section

HCFPGAFrameworkInit

Syntax

```
rtifpga_scriptinterface('Function',PathName,FileName)
```

Purpose

To initialize the handcode FPGA Framework INI file with the specified I/O interface.

Description

With this script function you can automatically configure the FPGA code with the I/O interface you specified. For more information on specifying the I/O interface, refer to Specifying the FPGA I/O Interface (RTI FPGA Programming Blockset Handcode Interface Guide 🕮).

Parameters

Function Lets you specify the function of the script interface to be used. To initialize the handcode FPGA Framework INI file with the specified I/O interface, you set the parameter to `'HCFPGAFrameworkInit'`.

PathName Lets you specify the path of the handcoded FPGA Framework INI file with the specified I/O interface.

FileName Lets you specify the name of the handcode FPGA framework INI file without file name extension.

Return value

No return value.

Example

This example demonstrates how to initialize the handcode DS5203 (7K325) with onboard I/O framework with a specified I/O interface. The path of the framework is `D:\Work\FPGAApplications`.

```
rtifpga_scriptinterface('HCFPGAFrameworkInit','D:\Work\FPGAApplications\DS5203',
'hc_fpga_framework_ini_DS5203_XC7K325T')
```

Related topics

Basics

Configuring the FPGA Code With the Specified I/O Interface (RTI FPGA Programming Blockset Handcode Interface Guide 📖)

HCFPGAModelIniGenerate

Syntax

```
rtifpga_scriptinterface('Function',PathName,FileName)
```

Purpose

To create a new application-specific FPGA model INI file that includes the FPGA bitstream.

Description

With this script function you can use the script interface to create a new FPGA model INI file after you built the bitstream. For more information, refer to Generating an FPGA Model INI File (RTI FPGA Programming Blockset Handcode Interface Guide 📖).

Parameters

Function Lets you specify the function of the script interface to be used. To create a new application-specific FPGA model INI file, you set the parameter to `'HCFPGAModelIniGenerate'`.

PathName Lets you specify the path of the handcoded FPGA Framework INI file with the handcoded FPGA functionality.

FileName Lets you specify the name of the handcode FPGA framework INI file without file name extension.

Return value

No return value.

Example

This example demonstrates how to create a new FPGA model INI file that includes the bitstream.

The framework with the handcoded FPGA functionality is the **DS5203 (7K325) with onboard I/O** framework. The path of the framework is `D:\Work\FPGAApplications`.

```
rtifpga_scriptinterface('HCFPGAModelIniGenerate','D:\Work\FPGAApplications\DS5203',
'hc_fpga_framework_ini_DS5203_XC7K325T')
```

Related topics

Basics

Generating an FPGA Model INI File (RTI FPGA Programming Blockset Handcode Interface Guide 🕮)