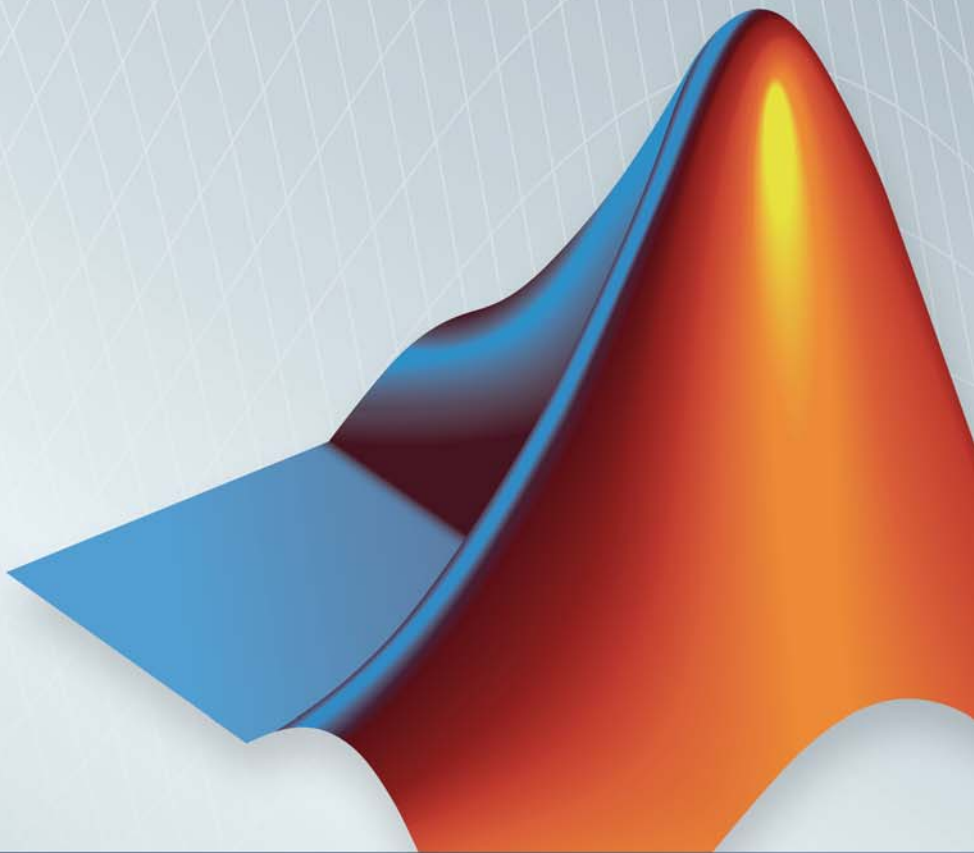


# HDL Coder™

## Reference

**R2013a**



# MATLAB®



## How to Contact MathWorks



[www.mathworks.com](http://www.mathworks.com)  
[comp.soft-sys.matlab](mailto:comp.soft-sys.matlab)  
[www.mathworks.com/contact\\_TS.html](http://www.mathworks.com/contact_TS.html)

Web  
Newsgroup  
Technical Support



[suggest@mathworks.com](mailto:suggest@mathworks.com)  
[bugs@mathworks.com](mailto:bugs@mathworks.com)  
[doc@mathworks.com](mailto:doc@mathworks.com)  
[service@mathworks.com](mailto:service@mathworks.com)  
[info@mathworks.com](mailto:info@mathworks.com)

Product enhancement suggestions  
Bug reports  
Documentation error reports  
Order status, license renewals, passcodes  
Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



The MathWorks, Inc.  
3 Apple Hill Drive  
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

*HDL Coder™ Reference*

© COPYRIGHT 2013 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

### Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

### Patents

MathWorks products are protected by one or more U.S. patents. Please see [www.mathworks.com/patents](http://www.mathworks.com/patents) for more information.

### Revision History

March 2013      Online only      New for Version 3.2 (R2013a)

## Function Reference

**1**

## Class Reference

**2**



# Function Reference

---

# codegen

---

<b>Purpose</b>	Generate HDL code from MATLAB code
<b>Syntax</b>	<pre>codegen -confighdlcfg matlab_design_name codegen -confighdlcfg -float2fixed fixptcfg matlab_design_name e</pre>
<b>Description</b>	<p>codegen -confighdlcfg matlab_design_name generates HDL code from MATLAB® code.</p> <p>codegen -confighdlcfg -float2fixed fixptcfg matlab_design_name converts floating-point MATLAB code to fixed-point code, then generates HDL code.</p>
<b>Input Arguments</b>	<p><b>hdlcfg - HDL code generation configuration</b> coder.HdlConfig</p> <p>HDL code generation configuration options, specified as a coder.HdlConfig object.</p> <p>Create a coder.HdlConfig object using the HDL coder.config function.</p> <p><b>matlab_design_name - MATLAB design function name</b> string</p> <p>Name of top-level MATLAB function for which you want to generate HDL code.</p> <p><b>fixptcfg - Floating-point to fixed-point conversion configuration</b> coder.FixptConfig</p> <p>Floating-point to fixed-point conversion configuration options, specified as a coder.FixptConfig object.</p> <p>Use fixptcfg when generating HDL code from floating-point MATLAB code. Create a coder.FixptConfig object using the HDL coder.config function.</p>

## Examples

### Generate Verilog® Code from MATLAB Code

Create a `coder.HdlConfig` object, `hdlcfg`.

```
hdlcfg = coder.config('hdl'); % Create an 'hdl' config with default settings
```

Set the test bench name. In this example, the test bench function name is `mlhdlc_dti_tb`.

```
hdlcfg.TestBenchName = 'mlhdlc_dti_tb';
```

Set the target language to Verilog.

```
hdlcfg.TargetLanguage = 'Verilog';
```

Generate HDL code from your MATLAB design. In this example, the MATLAB design function name is `mlhdlc_dti`.

```
codegen -config hdlcfg mlhdlc_dti
```

### Generate HDL Code from Floating-Point MATLAB Code

Create a `coder.FixptConfig` object, `fixptcfg`, with default settings.

```
fixptcfg = coder.config('fixpt');
```

Set the test bench name. In this example, the test bench function name is `mlhdlc_dti_tb`.

```
fixptcfg.TestBenchName = 'mlhdlc_dti_tb';
```

Create a `coder.HdlConfig` object, `hdlcfg`, with default settings.

```
hdlcfg = coder.config('hdl');
```

Convert your floating-point MATLAB design to fixed-point, and generate HDL code. In this example, the MATLAB design function name is `mlhdlc_dti`.

```
codegen -float2fixed fixptcfg -config hdlcfg mlhdlc_dti
```

# codegen

---

## See Also

`coder.FixptConfig` | `coder.HdlConfig` | `coder.config`

## Related Examples

- “Generate HDL Code from MATLAB Code Using the Command Line Interface”



**Purpose** Create HDL Coder code generation configuration objects

**Syntax**

```
config_obj = coder.config('hdl')  
config_obj = coder.config('fixpt')
```

**Description** `config_obj = coder.config('hdl')` creates a `coder.HdlConfig` configuration object for use with the HDL codegen function when generating HDL code from MATLAB code.

`config_obj = coder.config('fixpt')` creates a `coder.FixptConfig` configuration object for use with the HDL codegen function when generating HDL code from floating-point MATLAB code. The `coder.FixptConfig` object configures the floating-point to fixed-point conversion.

## Examples **Generate HDL Code from Floating-Point MATLAB Code**

Create a `coder.FixptConfig` object, `fixptcfg`, with default settings.

```
fixptcfg = coder.config('fixpt');
```

Set the test bench name. In this example, the test bench function name is `mlhdlc_dti_tb`.

```
fixptcfg.TestBenchName = 'mlhdlc_dti_tb';
```

Create a `coder.HdlConfig` object, `hdlcfg`, with default settings.

```
hdlcfg = coder.config('hdl');
```

Convert your floating-point MATLAB design to fixed-point, and generate HDL code. In this example, the MATLAB design function name is `mlhdlc_dti`.

```
codegen -float2fixed fixptcfg -config hdlcfg mlhdlc_dti
```

**See Also** `coder.HdlConfig` | `coder.FixptConfig` | `codegen`

# coder.config

---

## **Related Examples**

- “Generate HDL Code from MATLAB Code Using the Command Line Interface”

# coder.FixptConfig.addFunctionReplacement

<b>Purpose</b>	Replace floating-point function name with fixed-point function name
<b>Syntax</b>	<code>fxptcfg.addFunctionReplacement(floatFn, fixedFn)</code>
<b>Description</b>	<code>fxptcfg.addFunctionReplacement(floatFn, fixedFn)</code> specifies a function replacement in a <code>coder.FixptConfig</code> object. During floating-point to fixed-point conversion in the HDL code generation workflow, the coder replaces the specified floating-point function name with the specified fixed-point function name.
<b>Input Arguments</b>	<p><b>floatFn - Name of floating-point function</b> ' ' (default)   string Name of floating-point function, specified as a string.</p> <p><b>fixedFn - Name of fixed-point function</b> ' ' (default)   string Name of fixed-point function, specified as a string.</p>
<b>Examples</b>	<p><b>Specify Function Replacement in Fixed-Point Conversion Configuration Object</b></p> <p>Create a fixed-point code configuration object, <code>fxpCfg</code>, with a test bench, <code>myTestbenchName</code>.</p> <pre>fxpCfg = coder.config('fixpt'); fxpCfg.TestBenchName = myTestbenchName; fxpCfg.addFunctionReplacement('min', 'fi_min'); codegen -float2fixed fxpCfg designName</pre> <p>Specify that the floating-point function, <code>min</code>, should be replaced with the fixed-point function, <code>fi_min</code>.</p> <pre>fxpCfg.addFunctionReplacement('min', 'fi_min');</pre> <p>When you generate code, the coder replaces instances of <code>min</code> with <code>fi_min</code> during floating-point to fixed-point conversion.</p>

# coder.FixptConfig.addFunctionReplacement

---

**Alternatives** You can specify function replacements in the HDL Workflow Advisor. See “Function Replacements”.

**See Also** `coder.FixptConfig` | `coder.config` | `codegen`

# Class Reference

---

# coder.FixptConfig

---

<b>Purpose</b>	HDL codegen floating-point to fixed-point conversion configuration object
<b>Description</b>	A <code>coder.FixptConfig</code> object contains the configuration parameters that the HDL codegen function requires to convert floating-point MATLAB code to fixed-point MATLAB code during HDL code generation. Use the <code>-float2fixed</code> option to pass this object to the codegen function.
<b>Construction</b>	<code>fixptcfg = coder.config('fixpt')</code> creates a <code>coder.FixptConfig</code> object for floating-point to fixed-point conversion during HDL code generation.
<b>Properties</b>	<p><b>DefaultFractionLength</b></p> <p>Default fixed-point fraction length.</p> <p>Values: 4 (default)   positive integer</p> <p><b>DefaultWordLength</b></p> <p>Default fixed-point word length.</p> <p>Values: 14 (default)   positive integer</p> <p><b>FixPtFileNameSuffix</b></p> <p>Suffix for fixed-point file names.</p> <p>Values: <code>'_FixPt'</code>   string</p> <p><b>LaunchNumericTypesReport</b></p> <p>View the numeric types report after the coder has proposed fixed-point types.</p> <p>Values: <code>true</code> (default)   <code>false</code></p> <p><b>LogIOForComparisonPlotting</b></p> <p>Enable simulation data logging to plot the data differences introduced by fixed-point conversion.</p> <p>Values: <code>true</code> (default)   <code>false</code></p>

## **ProposeFractionLengthsForDefaultWordLength**

Propose fixed-point types based on DefaultWordLength.

Values: true (default) | false

## **ProposeWordLengthsForDefaultFractionLength**

Propose fixed-point types based on DefaultFractionLength.

Values: false (default) | true

## **SafetyMargin**

Safety margin percentage by which to increase the simulation range when proposing fixed-point types.

Values: 4 (default) | positive integer

## **TestBenchName**

Test bench function name, specified as a string. You must specify a test bench.

Values: '' (default) | string

## **Methods**

<code>addFunctionReplacement</code>	Replace floating-point function name with fixed-point function name
-------------------------------------	---

## **Examples**

### **Generate HDL Code from Floating-Point MATLAB Code**

Create a `coder.FixptConfig` object, `fixptcfg`, with default settings.

```
fixptcfg = coder.config('fixpt');
```

Set the test bench name. In this example, the test bench function name is `mlhdlc_dti_tb`.

```
fixptcfg.TestBenchName = 'mlhdlc_dti_tb';
```

Create a `coder.HdlConfig` object, `hdlcfg`, with default settings.

# coder.FixptConfig

---

```
hdlcfg = coder.config('hdl');
```

Convert your floating-point MATLAB design to fixed-point, and generate HDL code. In this example, the MATLAB design function name is `mlhdlc_dti`.

```
codegen -float2fixed fixptcfg -config hdlcfg mlhdlc_dti
```

## Alternatives

You can also generate HDL code from MATLAB code using the HDL Workflow Advisor. For more information, see “HDL Code Generation from a MATLAB Algorithm”.

## See Also

`coder.HdlConfig` | `coder.config` | `codegen`

## Related Examples

- “Generate HDL Code from MATLAB Code Using the Command Line Interface”



<b>Purpose</b>	HDL codegen configuration object
<b>Description</b>	A <code>coder.HdlConfig</code> object contains the configuration parameters that the HDL codegen function requires to generate HDL code. Use the <code>-config</code> option to pass this object to the codegen function.
<b>Construction</b>	<code>hdlcfg = coder.config('hdl')</code> creates a <code>coder.HdlConfig</code> object for HDL code generation.
<b>Properties</b>	<p><b>Basic</b></p> <p><b>GenerateHDLTestBench</b></p> <p>Generate an HDL test bench, specified as a logical.</p> <p>Values: <code>false</code> (default)   <code>true</code></p> <p><b>HDLCodingStandard</b></p> <p>HDL coding standard to follow and check when generating code, specified as a string. Generates a compliance report showing errors, warnings, and messages.</p> <p>Values: <code>'None'</code> (default)   <code>'Industry'</code></p> <p><b>HDLLintTool</b></p> <p>HDL lint tool script to generate, specified as a string. You must set <code>HDLCodingStandard</code> to <code>'Industry'</code> to use this property.</p> <p>Values: <code>'None'</code> (default)   <code>'SpyGlass'</code>   <code>'LEDA'</code></p> <p><b>SimulateGeneratedCode</b></p> <p>Simulate generated code, specified as a logical.</p> <p>Values: <code>false</code> (default)   <code>true</code></p> <p><b>PartitionFunctions</b></p> <p>Specify whether to generate instantiable HDL code modules from functions.</p> <p>Values: <code>false</code> (default)   <code>true</code></p>

## **SimulationIterationLimit**

Maximum number of simulation iterations during test bench generation, specified as an integer. This property affects only test bench generation, not simulation during fixed-point conversion.

Values: unlimited (default) | positive integer

## **SimulationTool**

Simulation tool name, specified as a string.

Values: 'ModelSim' (default) | 'ISIM'

## **SynthesisTool**

Synthesis tool name, specified as a string.

Values: 'Xilinx ISE' (default) | 'Altera Quartus II'

## **SynthesisToolChipFamily**

Synthesis target chip family name, specified as a string.

Values: 'Virtex4' (default) | string

## **SynthesisToolDeviceName**

Synthesis target device name, specified as a string.

Values: 'xc4vsx35' (default) | string

## **SynthesisToolPackageName**

Synthesis target package name, specified as a string.

Values: 'ff668' (default) | string

## **SynthesisToolSpeedValue**

Synthesis target speed, specified as a string.

Values: '-10' (default) | string

## **SynthesizeGeneratedCode**

Synthesize generated code or not, specified as a logical.

Values: false (default) | true

## **TargetLanguage**

Target language, specified as a string.

Values: 'VHDL' (default) | 'Verilog'

## **TestBenchName**

Test bench function name, specified as a string. You must specify a test bench.

Values: '' (default) | string

## **Cosimulation**

### **GenerateCosimTestBench**

Generate a cosimulation test bench or not, specified as a logical.

Values: false (default) | true

### **SimulateCosimTestBench**

Simulate generated cosimulation test bench, specified as a logical. This option is ignored if GenerateCosimTestBench is false.

Values: false (default) | true

### **CosimClockEnableDelay**

Time (in clock cycles) between deassertion of reset and assertion of clock enable.

Values: 0 (default)

### **CosimClockHighTime**

The number of nanoseconds the clock is high.

Values: 5 (default)

### **CosimClockLowTime**

The number of nanoseconds the clock is low.

Values: 5 (default)

## **CosimHoldTime**

The hold time for input signals and forced reset signals, specified in nanoseconds.

Values: 2 (default)

## **CosimLogOutput**

Log and plot outputs of the reference design function and HDL simulator.

Values: false (default) | true

## **CosimResetLength**

Specify time (in clock cycles) between assertion and deassertion of reset.

Values: 2 (default)

## **CosimRunMode**

HDL simulator run mode during simulation, specified as a string. When in Batch mode, you do not see the HDL simulator GUI, and the HDL simulator automatically shuts down after simulation.

Values: Batch (default) | GUI

## **CosimTool**

HDL simulator for the generated cosim test bench, specified as a string.

Values: ModelSim (default) | Incisive

## **FPGA-in-the-loop**

### **GenerateFILTestBench**

Generate a FIL test bench or not, specified as a logical.

Values: false (default) | true

## **SimulateFILTestBench**

Simulate generated cosimulation test bench, specified as a logical. This option is ignored if `GenerateCosimTestBench` is false.

Values: false (default) | true

## **FILBoardName**

FPGA board name, specified as a string. You must override the default value and specify a valid board name.

Values: 'Choose a board' (default) | string

## **FILBoardIPAddress**

IP address of the FPGA board, specified as a string. You must enter a valid IP address.

Values: 192.168.0.2 (default) | string

## **FILBoardMACAddress**

MAC address of the FPGA board, specified as a string. You must enter a valid MAC address.

Values: 00-0A-35-02-21-8A (default) | string

## **FILAdditionalFiles**

List of additional source files to include, specified as a string. Separate file names with a semi-colon (";").

Values: '' (default) | string

## **FILLogOutputs**

Log and plot outputs of the reference design function and FPGA.

Values: false (default) | true

## **Examples**

### **Generate Verilog Code from MATLAB Code**

Create a `coder.HdlConfig` object, `hdlcfg`.

```
hdlcfg = coder.config('hdl'); % Create an 'hdl' config with default settings
```

Set the test bench name. In this example, the test bench function name is `mlhdlc_dti_tb`.

```
hdlcfg.TestBenchName = 'mlhdlc_dti_tb';
```

Set the target language to Verilog.

```
hdlcfg.TargetLanguage = 'Verilog';
```

Generate HDL code from your MATLAB design. In this example, the MATLAB design function name is `mlhdlc_dti`.

```
codegen -config hdlcfg mlhdlc_dti
```

## **Generate Cosim and FIL Test Benches**

Create a `coder.FixptConfig` object with default settings and provide test bench name.

```
fixptcfg = coder.config('fixpt');  
fixptcfg.TestBenchName = 'mlhdlc_sfir_tb';
```

Create a `coder.HdlConfig` object with default settings and set enable rate.

```
hdlcfg = coder.config('hdl'); % Create an 'hdl' config with default settings  
hdlcfg.EnableRate = 'DUTBaseRate';
```

Instruct MATLAB to generate a cosim test bench and a FIL test bench. Specify FPGA board name.

```
hdlcfg.GenerateCosimTestBench = true;  
hdlcfg.FILBoardName = 'Xilinx Virtex-5 XUPV5-LX110T development board';  
hdlcfg.GenerateFILTestBench = true;
```

Perform code generation, Cosim test bench generation, and FIL test bench generation.

```
codegen -float2fixed fixptcfg -config hdlcfg mlhdlc_sfir
```

## Alternatives

You can also generate HDL code from MATLAB code using the HDL Workflow Advisor. For more information, see “HDL Code Generation from a MATLAB Algorithm”.

## See Also

[coder.FixptConfig](#) | [coder.config](#) | [codegen](#)

## Related Examples

- “Generate HDL Code from MATLAB Code Using the Command Line Interface”