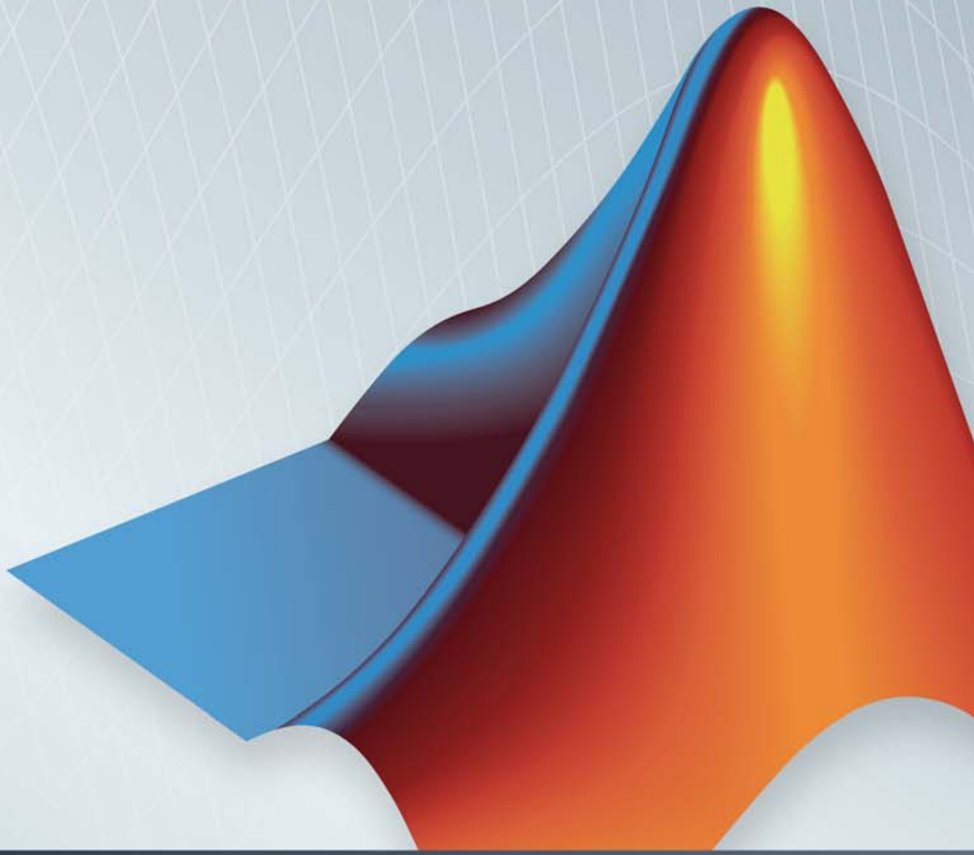
HDL Coder™

Reference

R2013a



MATLAB®

MathWorks®

Revision History

March 2013	Online only	New for Version 3.2 (R2013a)

Contents

Function Reference

codegen

Purpose

Generate HDL code from MATLAB code

Syntax

```
codegen -config hdlcfg matlab_design_name
codegen -config hdlcfg -float2fixed
fixptcfg matlab_design_nam
   e
```

Description

`codegen -config hdlcfg matlab_design_name` generates HDL code from MATLAB® code.

`codegen -config hdlcfg -float2fixed fixptcfg matlab_design_name` converts floating-point MATLAB code to fixed-point code, then generates HDL code.

Input Arguments

hdlcfg - HDL code generation configuration
`coder.HdlConfig`

HDL code generation configuration options, specified as a `coder.HdlConfig` object.

Create a `coder.HdlConfig` object using the HDL `coder.config` function.

matlab_design_name - MATLAB design function name
string

Name of top-level MATLAB function for which you want to generate HDL code.

fixptcfg - Floating-point to fixed-point conversion configuration
`coder.FixptConfig`

Floating-point to fixed-point conversion configuration options, specified as a `coder.FixptConfig` object.

Use `fixptcfg` when generating HDL code from floating-point MATLAB code. Create a `coder.FixptConfig` object using the HDL `coder.config` function.

Examples

Generate Verilog® Code from MATLAB Code

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

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

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
```

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`

Related Examples

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

coder.FixptConfig.addFunctionReplacement

Purpose

Replace floating-point function name with fixed-point function name

Syntax

```
fxptcfg.addFunctionReplacement(floatFn,fixedFn)
```

Description

`fxptcfg.addFunctionReplacement(floatFn,fixedFn)` specifies a function replacement in a `coder.FixptConfig` 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.

Input Arguments

floatFn - Name of floating-point function

`''` (default) | string

Name of floating-point function, specified as a string.

fixedFn - Name of fixed-point function

`''` (default) | string

Name of fixed-point function, specified as a string.

Examples

Specify Function Replacement in Fixed-Point Conversion Configuration Object

Create a fixed-point code configuration object, `fxpCfg`, with a test bench, `myTestbenchName`.

```
fxpCfg = coder.config('fixpt');
fxpCfg.TestBenchName = myTestbenchName;
fxpCfg.addFunctionReplacement('min', 'fi_min');
codegen -float2fixed fxpCfg designName
```

Specify that the floating-point function, `min`, should be replaced with the fixed-point function, `fi_min`.

```
fxpCfg.addFunctionReplacement('min', 'fi_min');
```

When you generate code, the coder replaces instances of `min` with `fi_min` during floating-point to fixed-point conversion.

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

Purpose

HDL `codegen` floating-point to fixed-point conversion configuration object

Description

A `coder.FixptConfig` 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 `-float2fixed` option to pass this object to the `codegen` function.

Construction

`*fixptcfg* = coder.config('fixpt')` creates a `coder.FixptConfig` object for floating-point to fixed-point conversion during HDL code generation.

Properties

DefaultFractionLength

Default fixed-point fraction length.

Values: 4 (default) | positive integer

DefaultWordLength

Default fixed-point word length.

Values: 14 (default) | positive integer

FixPtFileNameSuffix

Suffix for fixed-point file names.

Values: `'_FixPt'` | string

LaunchNumericTypesReport

View the numeric types report after the coder has proposed fixed-point types.

Values: `true` (default) | `false`

LogIOForComparisonPlotting

Enable simulation data logging to plot the data differences introduced by fixed-point conversion.

Values: `true` (default) | `false`

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

`addFunctionReplacement`	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.

```
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”

Purpose

HDL `codegen` configuration object

Description

A `coder.HdlConfig` object contains the configuration parameters that the HDL `codegen` function requires to generate HDL code. Use the `-config` option to pass this object to the `codegen` function.

Construction

`*hdlcfg* = coder.config('hdl')` creates a `coder.HdlConfig` object for HDL code generation.

Properties

Basic

GenerateHDLTestBench

Generate an HDL test bench, specified as a `logical`.

Values: `false` (default) | `true`

HDLCodingStandard

HDL coding standard to follow and check when generating code, specified as a string. Generates a compliance report showing errors, warnings, and messages.

Values: `'None'` (default) | `'Industry'`

HDLLintTool

HDL lint tool script to generate, specified as a string. You must set `HDLCodingStandard` to `'Industry'` to use this property.

Values: `'None'` (default) | `'SpyGlass'` | `'LEDA'`

SimulateGeneratedCode

Simulate generated code, specified as a `logical`.

Values: `false` (default) | `true`

PartitionFunctions

Specify whether to generate instantiable HDL code modules from functions.

Values: `false` (default) | `true`

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 setti
```

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 setti
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”