Generating HDL Code from Simulink

OEM: MathWorks • Code: MW-GENERATING-HDL-CODE-FROM-SIMULINK

COURSE DESCRIPTION

This two-day course shows how to generate HDL code from a Simulink® model using HDL Coder™. Preparing Simulink models for HDL code generation Generating HDL code and testbench for a compatible Simulink model Performing speed and area optimizations Modeling streaming architectures using explicit control signals Integrating existing code and IP Verifying generated HDL code using testbench and co-simulation

AUDIENCE PROFILE

- IT professionals and technical staff seeking hands-on training in this course's subject area.

PREREQUISITES

- Signal Processing with Simulink or equivalent experience using Simulink

COURSE OBJECTIVES

By the end of this course, participants will be able to:

- Prepare a Simulink model for HDL code generation. Generate HDL code and testbench for simple models that require no optimization.
- Establish correspondence between generated HDL code and specific Simulink blocks in the model. Use the Fixed-Point Tool to finalize the model's fixed-point architecture.
- Generate HDL code for multirate designs. Understand different modeling strategies to implement multirate designs.
- Use pipelines to meet design timing requirements. Use specific hardware implementations and share resources to optimize the area.
- Model hardware-friendly streaming architectures using explicit control signals. Include timing and area optimizations manually and ensure backpressure propagation.
- Implement floating-point values and operations in your HDL code.
- Incorporate existing HDL code in your design using the blackbox interface. Parameterize HDL code to increase reusability and readability.

COURSE OUTLINE

Module 1: Day 1 of 2

Preparing Simulink Models for HDL Code Generation

- Preparing Simulink models for HDL code generation
- Generating HDL code
- Generating a test bench
- Verifying generated HDL code with an HDL simulator

Fixed-Point Precision Control

- Fixed-point scaling and inheritance
- Fixed-Point Designer workflow
- Fixed-Point Tool
- Command-line interface

Generating HDL Code for Multirate Models

- Preparing a multirate model for generating HDL code
- Generating HDL code with single or multiple clock pins
- Verifying multirate designs with co-simulation
- Designing a simplified streaming interface for multirate applications

Module 2: Day 2 of 2

Optimizing Generated HDL Code

- Generating HDL code with the HDL Workflow Advisor
- Meeting timing requirements via pipelining
- Choosing specific hardware implementations for compatible Simulink blocks
- Sharing FPGA/ASIC resources in subsystems
- Verifying that the optimized HDL code is bit-true cycle-accurate
- Mapping Simulink blocks to dedicated hardware resources on the FPGA

Modeling and Optimizing Streaming Architectures

- Model a fully parallel streaming architecture
- Insert pipeline registers into a clock rate model
- Understand the modeling steps from a parallel to serial architecture
- Ensure correct stall behavior with valid/ready handshaking

Using Native Floating Point

- Why and when to use native floating-point
- Target-independent HDL code generation with HDL Coder
- Fixed-point vs. floating-point comparison
- Optimization of floating-point implementations

Interfacing External HDL Code with Generated HDL

- Interfacing external HDL code
- Increase code reusability and readability