Intel® FPGA Technical Training

Quartus Prime Pro High End Features

Course Description

This course provides all practical know-how needed to start designing with Quartus Prime Pro edition.

The course starts with features comparison between the Standard and Pro editions and how to migrate from one to the other.

The course continues by introducing the Interface Planner tool and how to use it efficiently in complex designs.

The second day introduces the Platform Designer, incremental optimization, and block-based design. Then Timing Analyzer tool introduced to write SDC files and analyze timing.

The course ends with scripting methods to automate design tasks and processes.

The training embeds 50% practical labs that covers the theory.

Course Duration

2 days

Goals

1. Migrate from Standard to Pro edition
2. Use the Interface Planner for complex designs
3. Be familiar with Timing Analyzer new features for timing analysis
4. Automate your design using scripting

Intended Users

Digital hardware engineers and FPGA team leaders, who would like migrate to Quartus Pro and/or using Arria 10/Stratix 10 FPGAs

Previous Knowledge

Quartus Prime Standard

FPGA Design

Course Material

1. Intel official course book
2. Course labs handbook
3. Quartus Prime Pro

Table of Contents

Day #1

- **Features Comparison & Migration: Standard vs Pro**
  - Intel Quartus Prime software key benefits
  - Why Pro edition?
  - Quartus Prime design software features
  - Migration from Standard to Pro edition
    - Back up and isolate your project
    - How do I migrate?
    - Update assignments to reference instances only
    - Instance name focus in the Project Navigator
    - Check Entity name use in sdc and related scripts
    - Pro edition functioning and non-functioning scripts
    - Synthesis generated nodes might change names
    - LogicLock region comparison
    - Non-rectangular regions
    - Parent-child region constraints
    - Overlapping regions with different assigned instances
    - Why regenerate IP?
    - Migrating between device families

- **Early Design Planning Using Interface Planner**
  - FPGA implementation phases
  - The need for interface planning
  - Interface planning before
  - Problem: periphery interfaces are complex
  - Previous flow: Pin Planner and Assignment Editor
  - Better solution: interface-based assignment
  - Introducing the Interface Planner
  - Interface Planner in the compilation flow
  - Interface Planner usage flow
    - Create and synthesize representative design
    - Sources of initial design
    - Using a design from the design store
    - Virtual pins
    - Open and initialize Interface Planner
    - Initialization summary report
    - Check imported assignment and update plan
    - Plan tab chip view – click and drag
    - List available locations
    - Chip view results
    - Cross-highlight placed elements
    - Continuous legality syncing
    - Link info
    - Automatic placement
    - Package view results
    - Individual pins
    - Clock network floorplanning
    - Save/Load/Reset Plan
    - Interface Planner reporting
    - Additional reports
    - Validate I/O plan
    - Write out constraints file
    - Source into project & compile
    - Board design resources and interface handoff
    - Pin Planner vs Interface Planner

- **Lab #1:** Migrate a Design and Perform Early Design Planning

Day #2

❖ **Implementation with Incremental Optimization & Block-Based Design**

- Platform Designer
  - Platform Designer in the FPGA design flow
  - PD advantage: automatic interconnect generation
  - Target PD applications
  - Features
  - Design reuse possibilities
  - Project association
  - The PD GUI
  - Platform Designer vs Platform Designer Standard
  - Additional Platform Designer Pro features
- Incremental Optimization
  - Fitter stages and Incremental Optimization overview
  - Checkpoints and snapshots
  - Early placement
  - Per-stage compilation benefits
  - Using the multi-stage fitter
  - Per-stage fitter reports contents
  - Incremental Optimization features availability
  - Per-stage fitter reports
  - Per-stage timing analysis
  - Timing analyzer launch from compilation dashboard
  - Plan checkpoint
  - Place checkpoint
  - Using post-place timing analysis
  - Scenarios for using post-place timing analysis
  - Route checkpoint
- Block-Based Design: Partitions & Logic Lock regions
  - What are user-defined design partitions?
  - Design partitions example
  - What can be a partition?
  - Core vs periphery partitions
  - Resource classification
  - Setting design entities as partitions
  - Design partitions window
  - Design partition planner
  - No optimizations across boundaries for synthesis

- What are Logic Lock regions?
- Logic Lock regions in Chip Planner
- Logic Lock regions window
- Logic Lock regions assignment syntax
- Merging Logic Lock regions
- Non-rectangular regions in the QSF file
- Hierarchical region constraints
- Overlapping regions with different assigned instances
- Routing regions
- Routing type
- Logic Lock region comparison: Standard vs Pro

- Incremental Block-Based Design
  - Example uses
  - Top-down design model flow
  - Understanding preservation level
  - Partitin Netlist types
  - Compilation flow
  - Empty assignment
  - Empty vs final partitions
  - Bringing Empty partitions back
  - Using Incremental Block-Based compilation
  - Creating black box wrapper files
  - Quick periphery planning
- Design block reuse
  - Reuse design flows: core reuse
  - Reuse design flows: root partition reuse
  - Developer project flows
  - Export partition
  - Files to hand-off
  - Design partitioning
  - Integrate reused core partition
  - Periphery reuse
  - Compile & create core logic
- Partial reconfiguration
  - What is partial reconfiguration?
  - Partial reconfiguration applications
  - Physical interfaces for partial reconfiguration
  - Partial reconfiguration considerations

❖ **Lab #2:** Use Block-Based Design Methodology

- **Timing Analysis**
  - Specify SDC file(s)
  - Timing Analyzer
  - Timing Analyzer GUI
    - Tasks pane
    - Report pane
    - View pane
    - Console pane
  - Basic steps for using timing analyzer
  - Generate timing netlist
  - Constrain directly in console
  - Constraining
  - Generate timing reports
  - Advanced clock pessimism
  - Pessimism removal background
  - Advanced clock pessimism removed
  - EOL and PSIJ
  - Report clock waveforms
  - Clock Domain Crossing (CDC) viewer
  - Live filtering
  - CDC viewer details

❖ **Lab #3:** Examine the New Features in the Timing Analyzer

- **Scripting & Automating Your Flow**
  - Quartus Prime command-line support
  - Benefits of command-line executables
  - Options and reports
  - Typical design flow
  - Integrated into the GUI
  - Typical use scenarios
  - Getting help – command line options
  - Qhelp utility
  - Tcl for expanded control
  - Other ways to use Tcl control
  - Tcl packages
  - Using ::quartus::flow to fully automate your flow
  - Assignment files
  - Using Tcl as part of your QSF file
  - Example scenario
  - Assignment precedence
  - Writing settings files

- Archive or restore a project example
- Fit a design using multiple seeds example
- Platform designer executables
- Simulation script generation

❖ **Lab #4:** Scripting & Automating