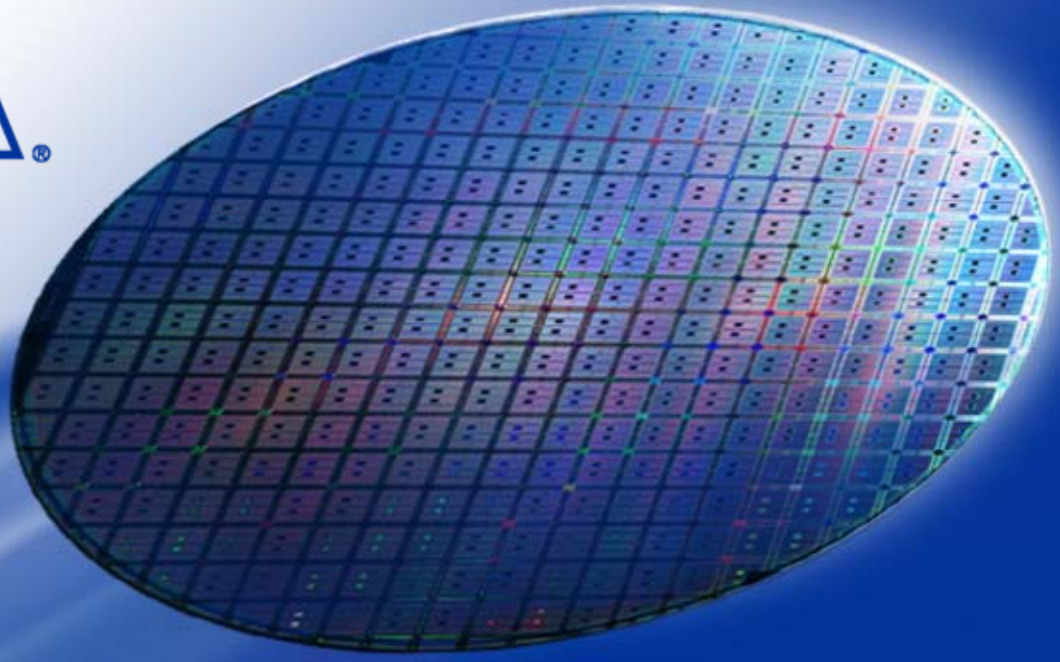


Designing with Synplicity SynplifyPro & Altera's Quartus II Software

Copyright © 2004 Altera Corporation

Outline

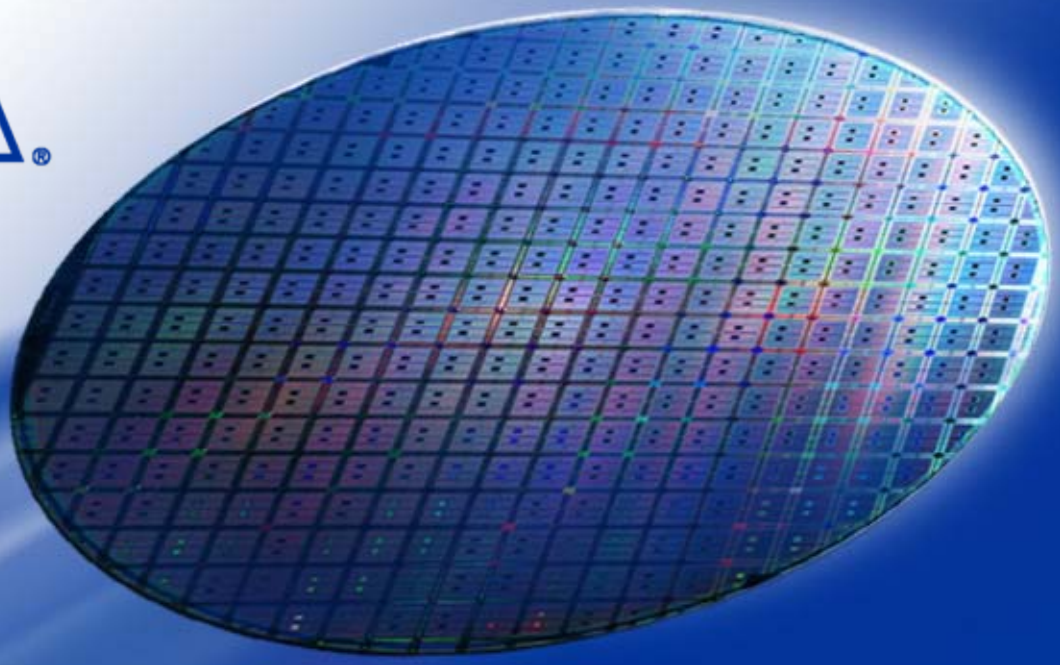
- Introduction to Altera® & Altera Devices
- PLD Design Flow
- Quartus® II / Synplify Pro Overview
- Design & Synthesis with Synplify Pro
  - Exercises 1 & 2
- Quartus II Project Management & Compilation
  - Exercise 3
- NativeLink®
  - Exercise 4
- Optimizing Designs with Synplify Pro & Quartus II
  - Exercises 5 & 6
- TCL Scripting

Copyright © 2004 Altera Corporation

Course Objectives

- Set Up Constraints and Perform HDL Synthesis with Synplify Pro
- Optimize FPGA Designs for Higher Performance
- View & Control Synthesis & Place & Route Results
- Analyze and Debug Designs with Synplify Pro
- Set Up Projects and Compile Designs in Quartus II
- Use Quartus II MegaWizard Plug-In Manager to Generate LPM Functions
- Synthesize and Place & Route a Design Using Black-box
- Use Synplify Pro Driven Design Flow

Copyright © 2004 Altera Corporation



Introduction to Altera & Altera Devices

Copyright © 2004 Altera Corporation

The Programmable Solutions Company®

- Programmable Devices
- Design Software
- Intellectual Property (IP)

Copyright © 2004 Altera Corporation

Introduction to Altera Devices

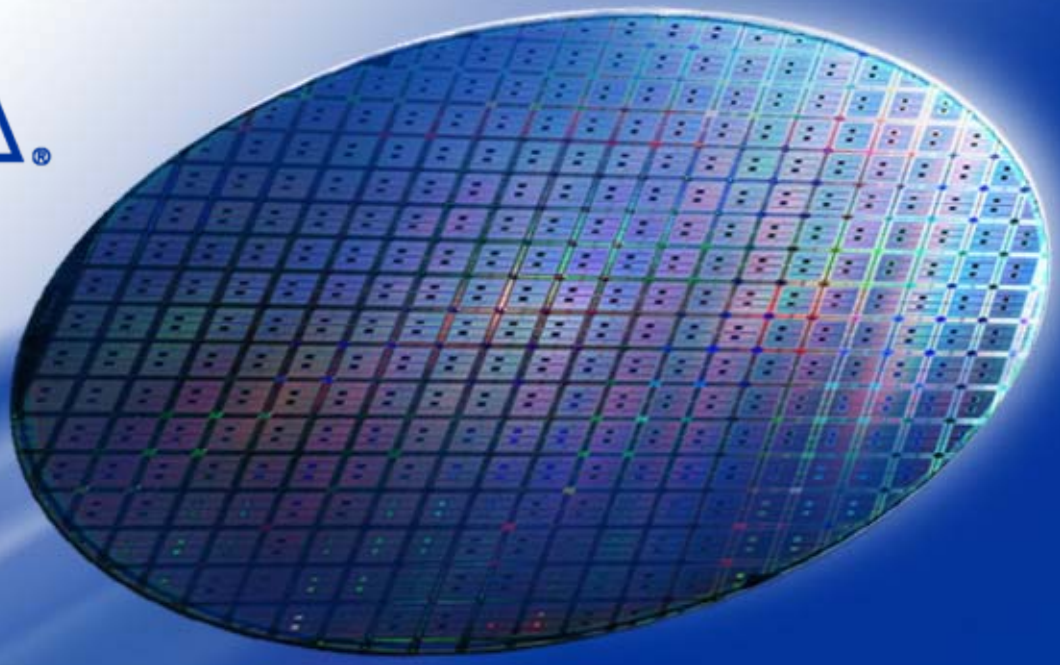
- Programmable Logic Families
  - High & Medium Density FPGAs
    - Stratix™ II, Stratix, APEX™ II, APEX 20K, & FLEX® 10K
  - Low-Cost FPGAs
    - Cyclone™ & ACEX® 1K
  - FPGAs with Clock Data Recovery
    - Stratix GX & Mercury™
  - CPLDs
    - MAX® 7000 & MAX 3000
  - Embedded Processor Solutions
    - Nios™, ExcaliburT™
  - Configuration Devices
    - EPC

Copyright © 2004 Altera Corporation

Introduction to Altera Design Software

- Software & Development Tools:
  - Quartus II
    - Stratix II, Stratix, Stratix GX, Cyclone, APEX II, APEX 20K/E/C, Excalibur, & Mercury Devices
    - FLEX 10K/A/E, ACEX 1K, FLEX 6000, MAX 7000S/AE/B, MAX 3000A Devices
  - Quartus II Web Edition
    - Free Version
    - Not All Features & Devices Included
  - MAX+PLUS® II
    - All FLEX, ACEX, & MAX Devices

Copyright © 2004 Altera Corporation



Typical PLD Design Flow

Copyright © 2004 Altera Corporation

Typical PLD Design Flow

Design Specification

Design Entry/RTL Coding
- Behavioral or Structural Description of Design

RTL Simulation (Quartus II or ModelSim®)
- Functional Simulation
- Verify Logic Model (No Timing Delays)
- May Require Design Edits

Synthesis
- Translate Design into Target Technology Primitives
- Optimization to Meet Required Area & Performance Constraints

Synplicity

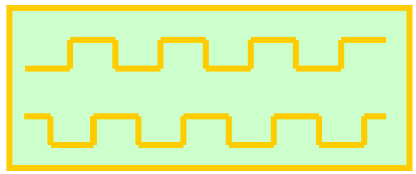
QUARTUS® II

Place & Route
- Map Primitives to Specific Locations Inside Target Technology
- Specify Routing Resources to Be Used
- May Require Design Edits

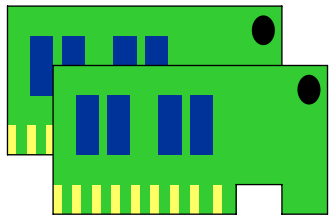
Copyright © 2004 Altera Corporation

Typical PLD Design Flow

Timing Analysis
- Verify Performance Specifications Were Met
- May Require Design Edits



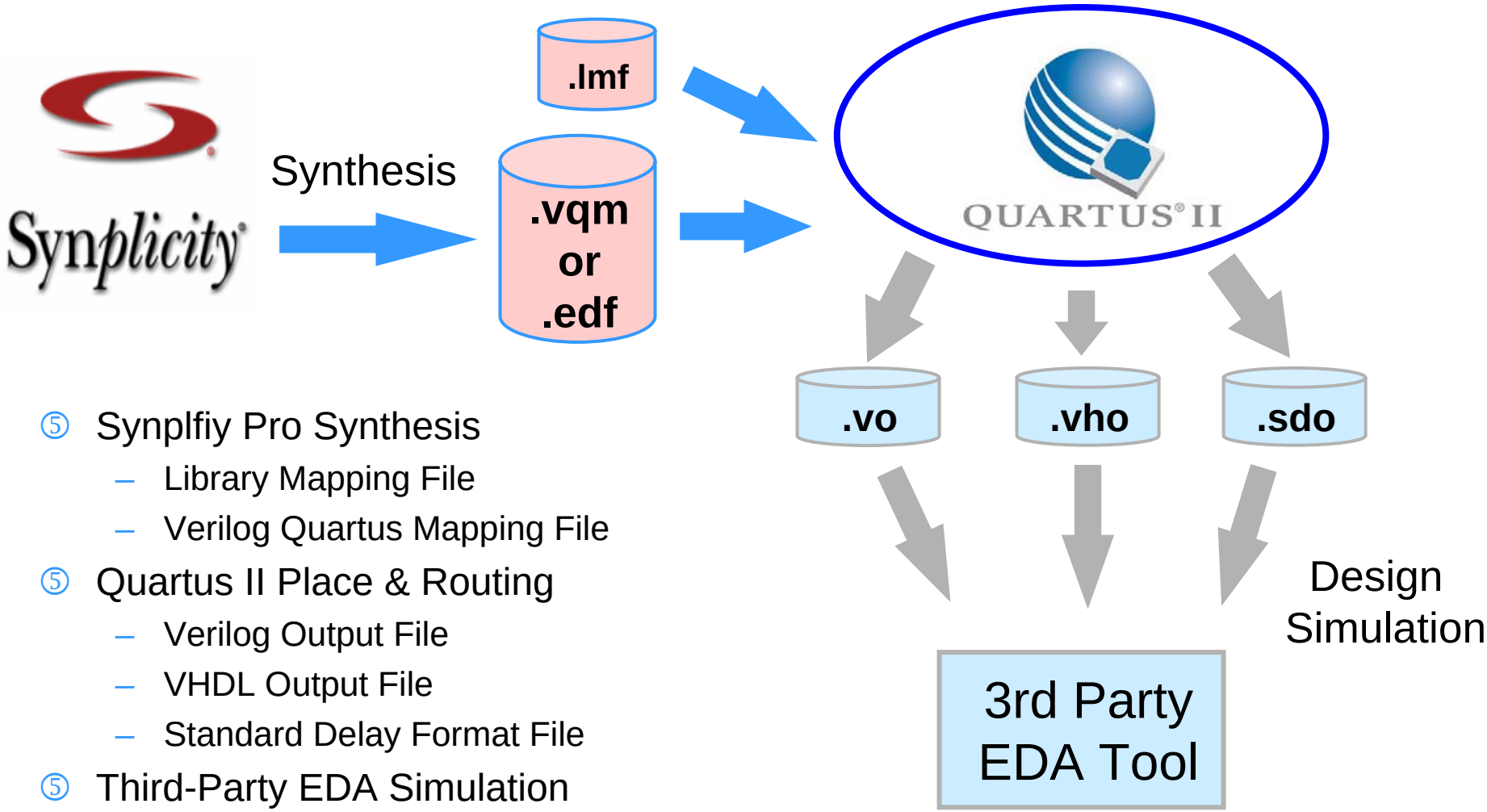
Gate Level Simulation (Quartus II or ModelSim)
- Timing Simulation
- Verify Design Will Work in Target Technology
- May Require Design Edits



PC Board Simulation & Test
- Simulate Board Design
- Program & Test Device on Board

Copyright © 2004 Altera Corporation

Synplify Pro / Quartus II File Flow Overview



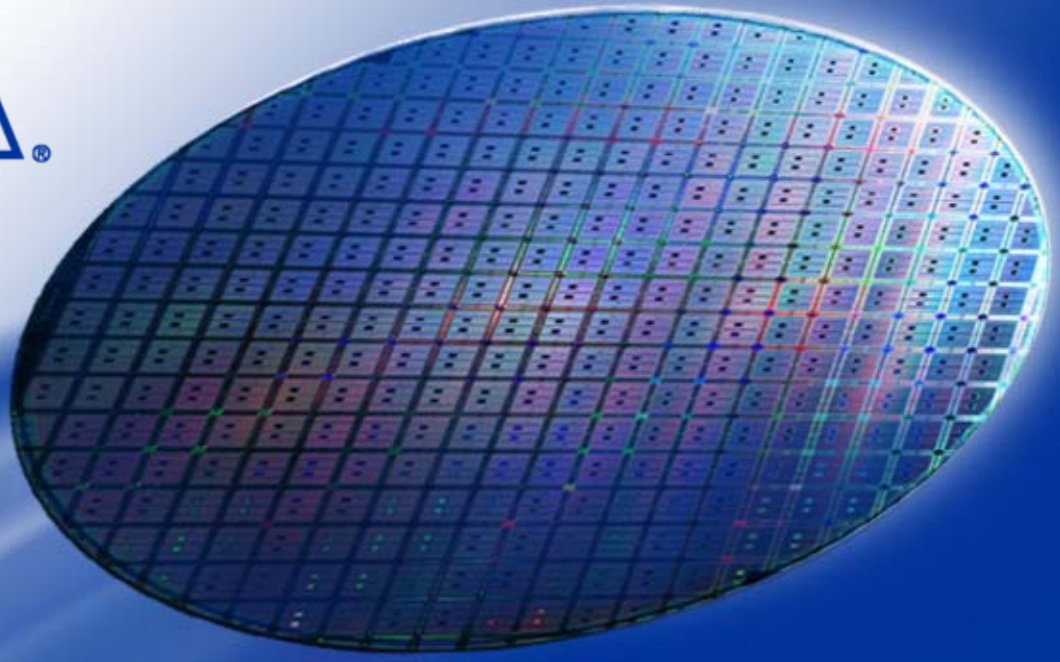
- Synplfiy Pro Synthesis
  - Library Mapping File
  - Verilog Quartus Mapping File
- Quartus II Place & Routing
  - Verilog Output File
  - VHDL Output File
  - Standard Delay Format File
- Third-Party EDA Simulation

Copyright © 2004 Altera Corporation

Why Synthesize with Synplify Pro?

- Better Quality of Results
  - Better Fmax and smallest LE usage
  - Synplicity dedicated to synthesis and simulation
  - Synplify Pro maps directly to Altera primitives
  - Automatic Altera megafunction use (+, -, *, RAM, etc…)
- Better Error Messages and Easier Debugging
  - Better verilog and VHDL error messages
- Some Assignment Passing Capabilities

Copyright © 2004 Altera Corporation



Quartus II / Synplify Pro Overview

Copyright © 2004 Altera Corporation

Quartus II Development System

⑤Fully-Integrated Design Tool

- Multiple Design Entry Methods
- Logic Synthesis
- Place & Route
- Simulation
- Timing & Power Analysis
- Device Programming

Copyright © 2004 Altera Corporation

More Features

- MegaWizard® & SOPC Builder Design Tools
- LogicLock™ Optimization Tool
- NativeLink® 3rd-Party EDA Tool Integration
- Integrated Embedded Software Development
- SignalTap® II & SignalProbe™ Debug Tools
- Windows, Solaris, HPUX, & Linux Support
- Node-Locked & Network Licensing Options
- Revision Control Interface

Copyright © 2004 Altera Corporation

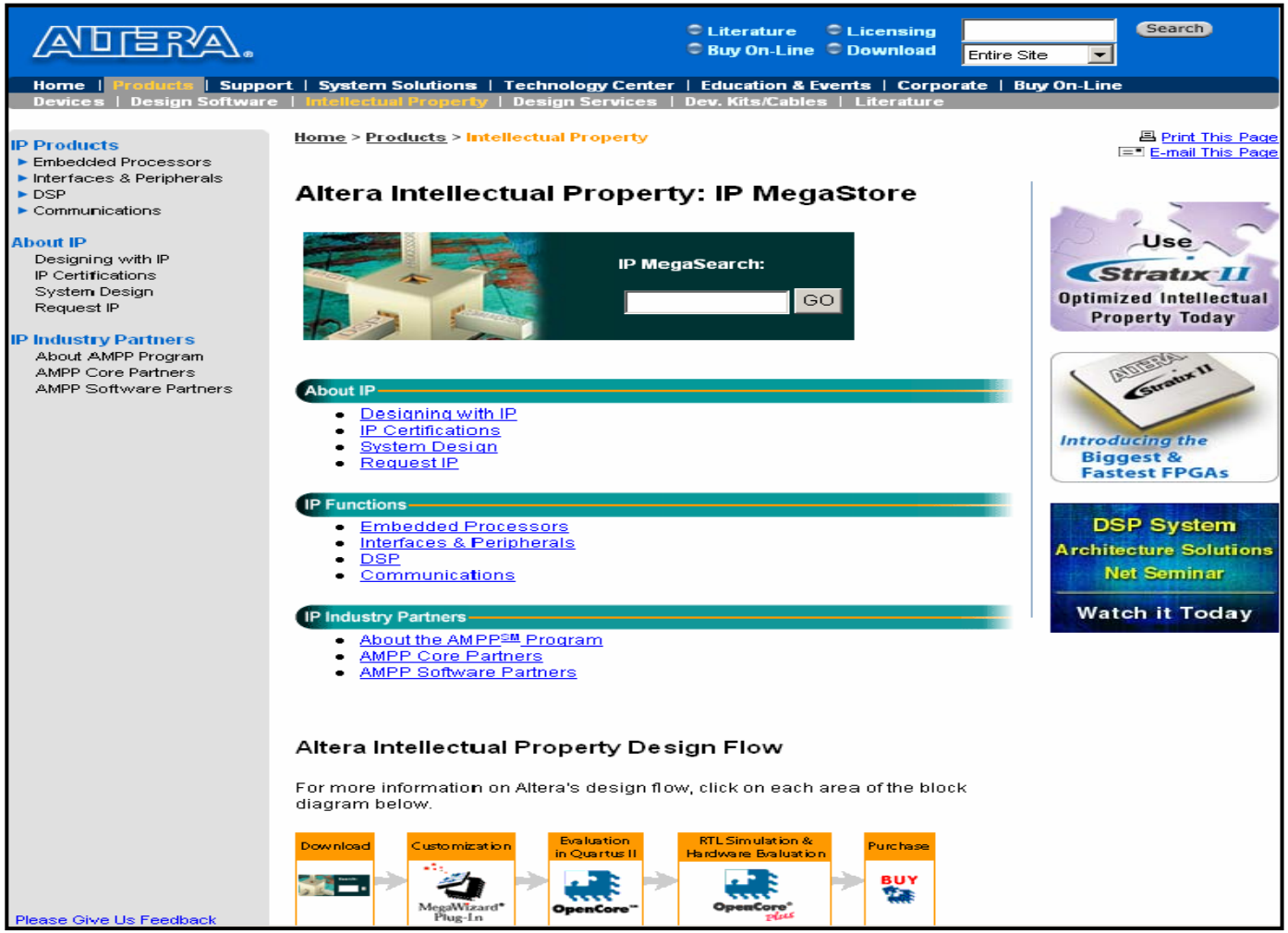
Intellectual Property (IP) Support

- MegaCore® Program
  - Staffed with Altera employees
  - MegaCore group develops functions that support the PCI bus interface, DSP applications, and communication applications
- AMPP℠ Program (Altera Megafunction Partners Program)
  - AMPP partners are 3rd party IP developers who design IP optimized for Altera devices
- Both programs support OpenCore which allows designers to evaluate megafunctions prior to licensing

Visit www.altera.com ⟹ Intellectual Property ⟹

Copyright © 2004 Altera Corporation

IP MegaStore™



Copyright © 2004 Altera Corporation

System-Level Verification

- SignalTap logic analyzer offers system-level verification of devices running at speed
  - Verification times are reduced
  - Megafunction provided free of charge with Quartus II
- SignalTap works together with Quartus II and the MasterBlaster™ or ByteBlasterMV™ communication cables

Copyright © 2004 Altera Corporation

Internet-Enabled Support

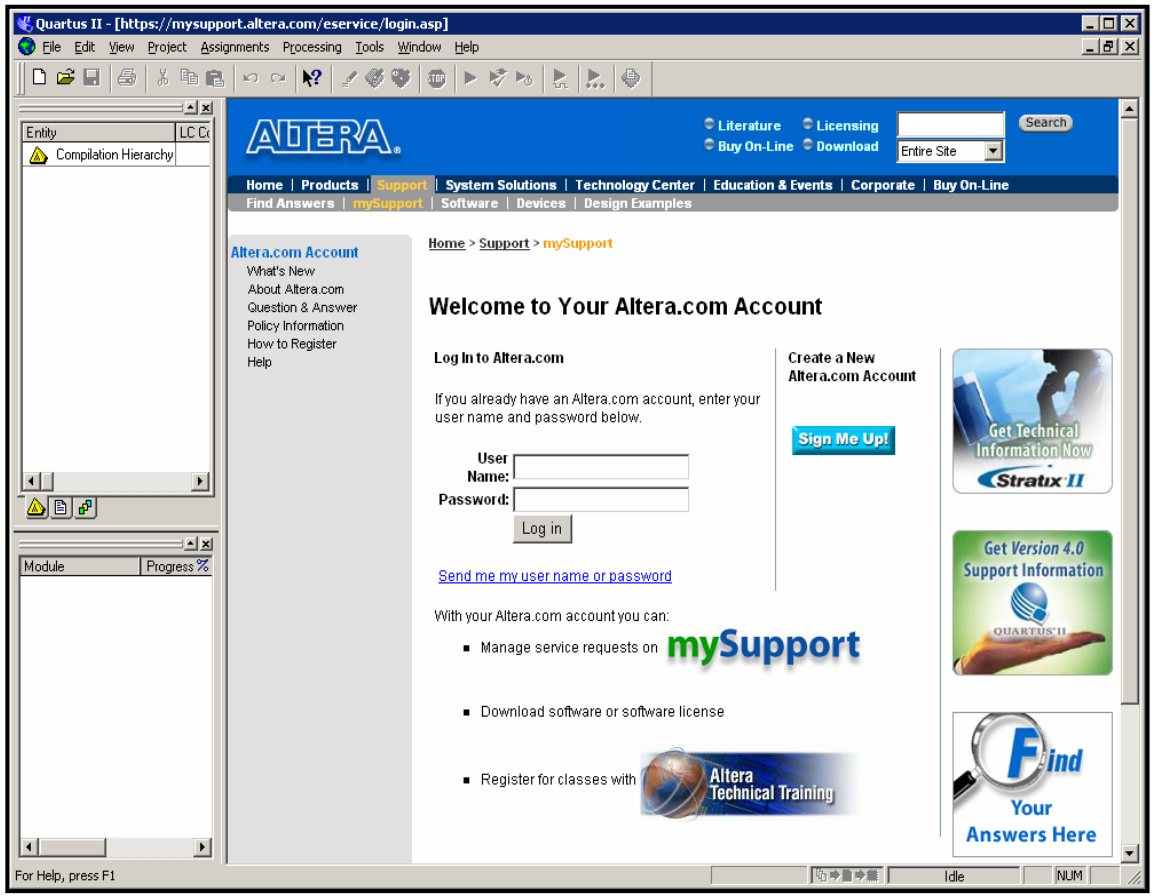
- Register users
- Download software updates
- Obtain license files
- Submit and monitor service requests with Altera Applications
- Evaluate and purchase MegaCore functions
- Search Atlas® database and other technical support pages

Copyright © 2004 Altera Corporation

Internet Capabilities

⑤ Launch Altera web support from within Quartus II

– Help > Altera on the Web>



Copyright © 2004 Altera Corporation

What's New In Quartus II 4.0?

- Stratix II Support
- RTL Viewer
- MAX+PLUS II "Look & Feel"
- SignalTap II Advanced Triggering
- Chip Editor Enhancements
- Single Assignment File (.QSF)
- Revisions
- Balanced Optimization Technique
- Auto Fitter Effort
- New Design Space Explorer Interface
- Early Power Estimation

Copyright © 2004 Altera Corporation

Questions about Quartus II?

- Quartus II on-line Help has the answers
  - Contains the complete up-to-date information on Quartus II
  - Provides tips on how to effectively work with Quartus II tools
  - Provides answers and examples
- To access on-line Help
  - Help menu content:
    - Starting Projects
    - Creating and editing files
      - Using AHDL, VHDL, Verilog
      - Libraries: logic functions, Megafunctions, Primitives
    - Using other EDA tools
    - Programming or configuring a device
  - Press F1 or Shift-F1 to access context sensitive help

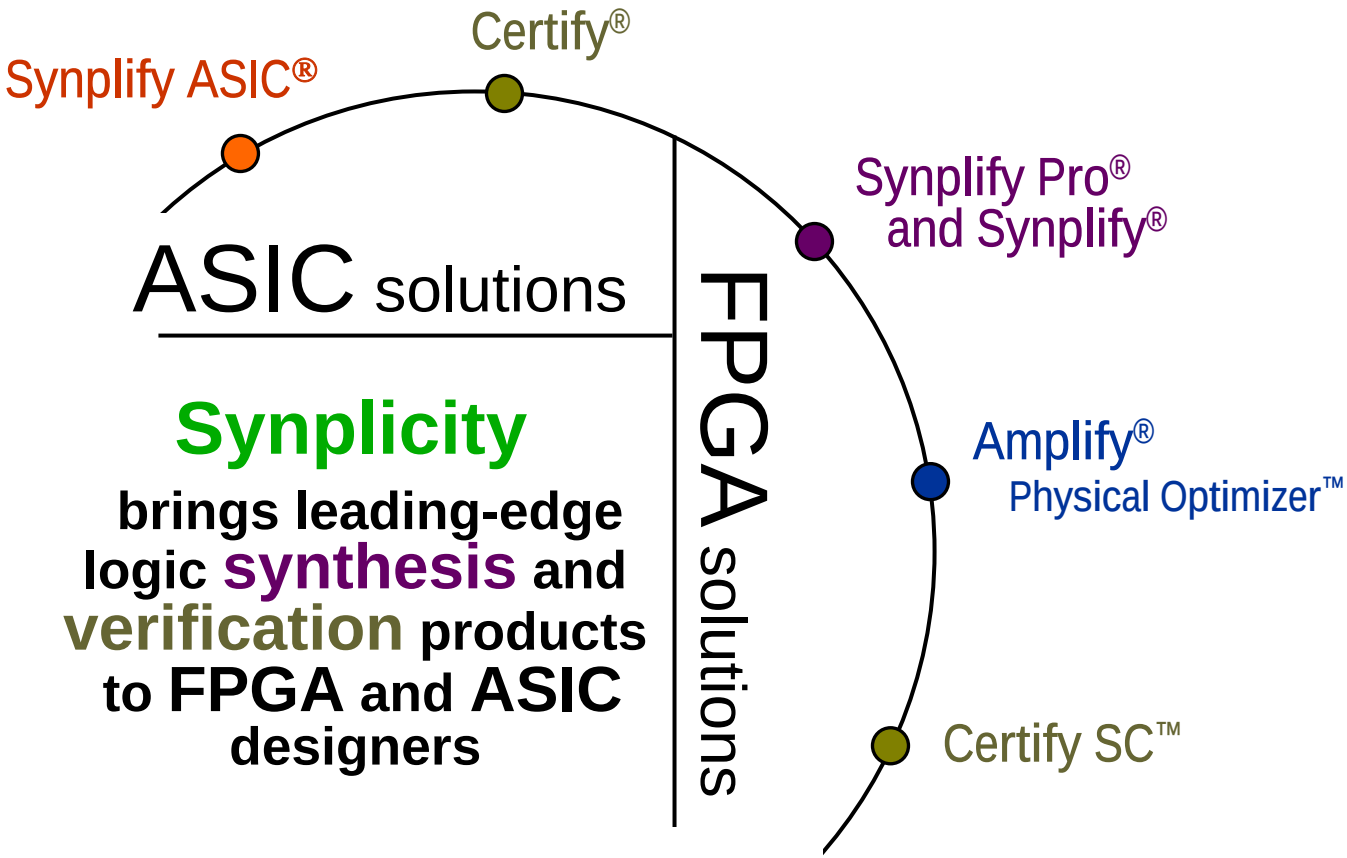
Copyright © 2004 Altera Corporation

Synplify Pro FPGA Synthesis Product

- Synplify Pro Is a Synthesis Tool Produced by Synplicity
- Synplify Can Synthesize both Verilog and VHDL Designs
  - Can Synthesized Mixed Designs
- Supported on Multiple Platforms
  - Windows XP / Windows 2000 / Windows ME / Windows NT / Windows ’98
  - Sun OS / Solaris
  - HP-UX

Copyright © 2004 Altera Corporation

Synplicity's Current Products



Copyright © 2004 Altera Corporation

Synplicity Product Comparisons

Amplify™ Physical Optimizer™

Physical Synthesis for FPGAs

- **Highest Circuit Performance**
- **Fastest Timing Closure**
- **Option to Synplify Pro**

Syn*plify*® Pro

- **Challenging Designs**
- **Complex Projects**
- **The Ultimate in FPGA Synthesis**

Syn*plify*®

- **Fast**
- **Easy to Use**
- **Excellent Results**

Copyright © 2004 Altera Corporation

Synplify Pro Features

- HDL Analyst
- Text Editor
- Behavior Extraction Synthesis Technology (B.E.S.T.)
- Synthesis Constraint Optimization Environment (SCOPE) Editor
- FSM (Finite State Machine) Support
  - FSM Compiler / FSM Explorer / FSM Viewer
- Invoke Place & Route via NativeLink
- MultiPoint ™ Synthesis Flow
- Stratix MAC Support

Copyright © 2004 Altera Corporation

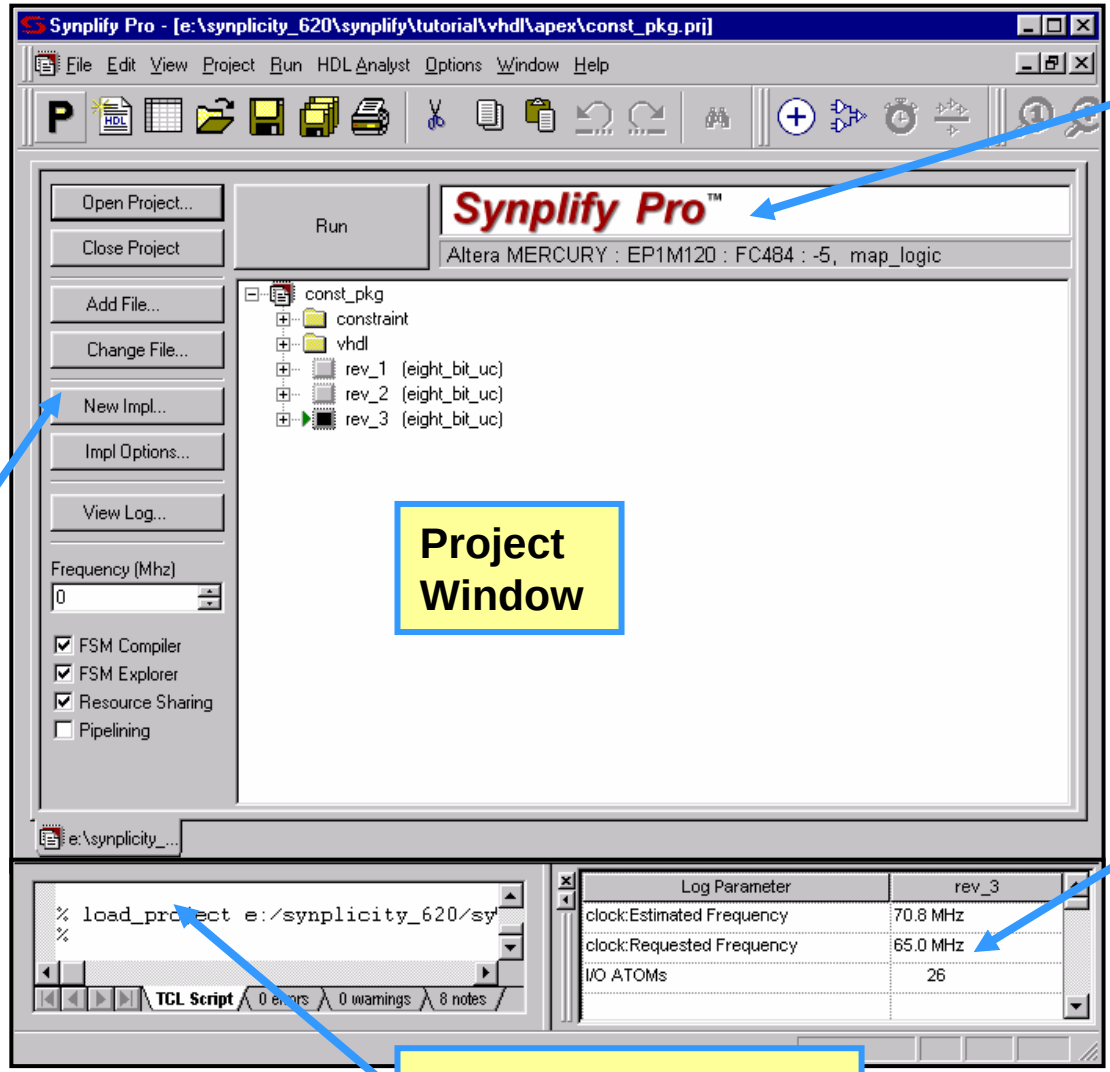
Synplify Pro Versus Synplify (Partial)

Feature	Synplify	Synplify
B.E.S.T.	X	X
Text Editor	X	X
HDL Analyst	X	Optional
FSM Compiler	X	X
FSM Explorer/Viewer	X	
SCOPE	X	X
Log Watch Window	X	
Text Editor Cross-Probing	X	
Multiple Implementations	X	

Note: Please See On-Line Help in Synplify Pro or Synplify for a Complete List of Feature Differences (Help ⇒Synplify Pro vs. Synplify)

Copyright © 2004 Altera Corporation

Synplify Pro User Interface



Status Bar

Project Window

Button Panel

Log Watch Window

TCL Script & Messages Window

Copyright © 2004 Altera Corporation

Getting Help with Synplify Pro

⑤ On-line Help

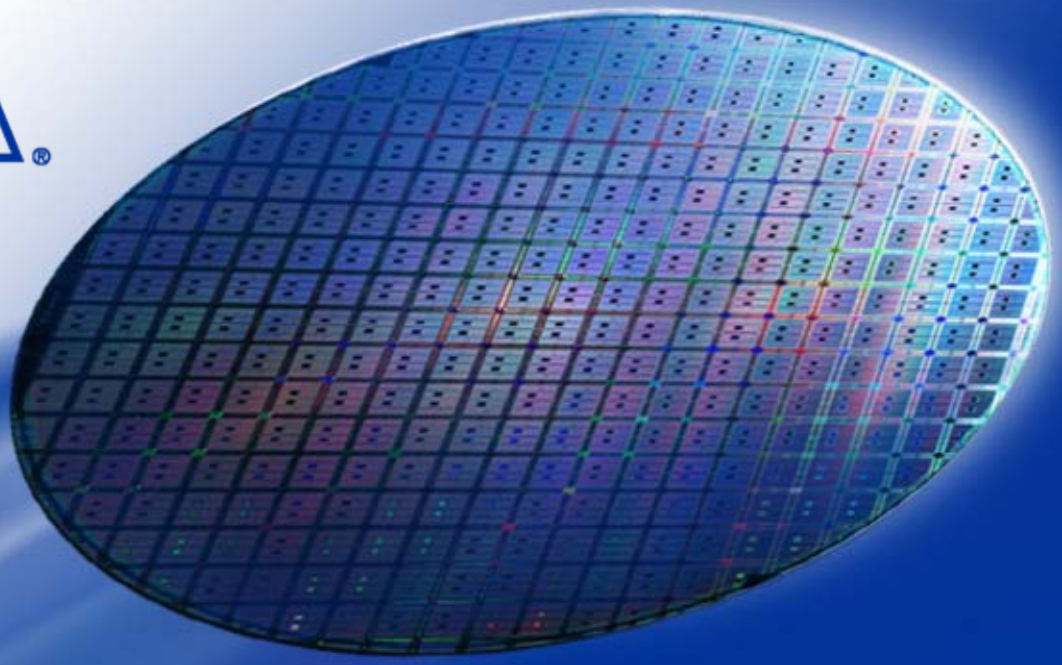
- Select Help->Help, or [F1] Function Key from the Synplify Pro UI

⑤ Synplify Pro Documents

(Start menu ⇒Synplicity ⇒Synplify Pro Documents)

- Synplify Pro User Guide & Tutorial (synplify_ug.pdf)
  - ③Overview of User Interface and Synthesis Guidelines
- Synplify Pro Reference Manual (synpro_ref.pdf)
  - ③Details on the User Interface
  - ③VHDL and Verilog Basics & Support
- Synplify Pro Release Notes (synpro_rnotes.pdf)

Copyright © 2004 Altera Corporation



Designing with Synplify Pro

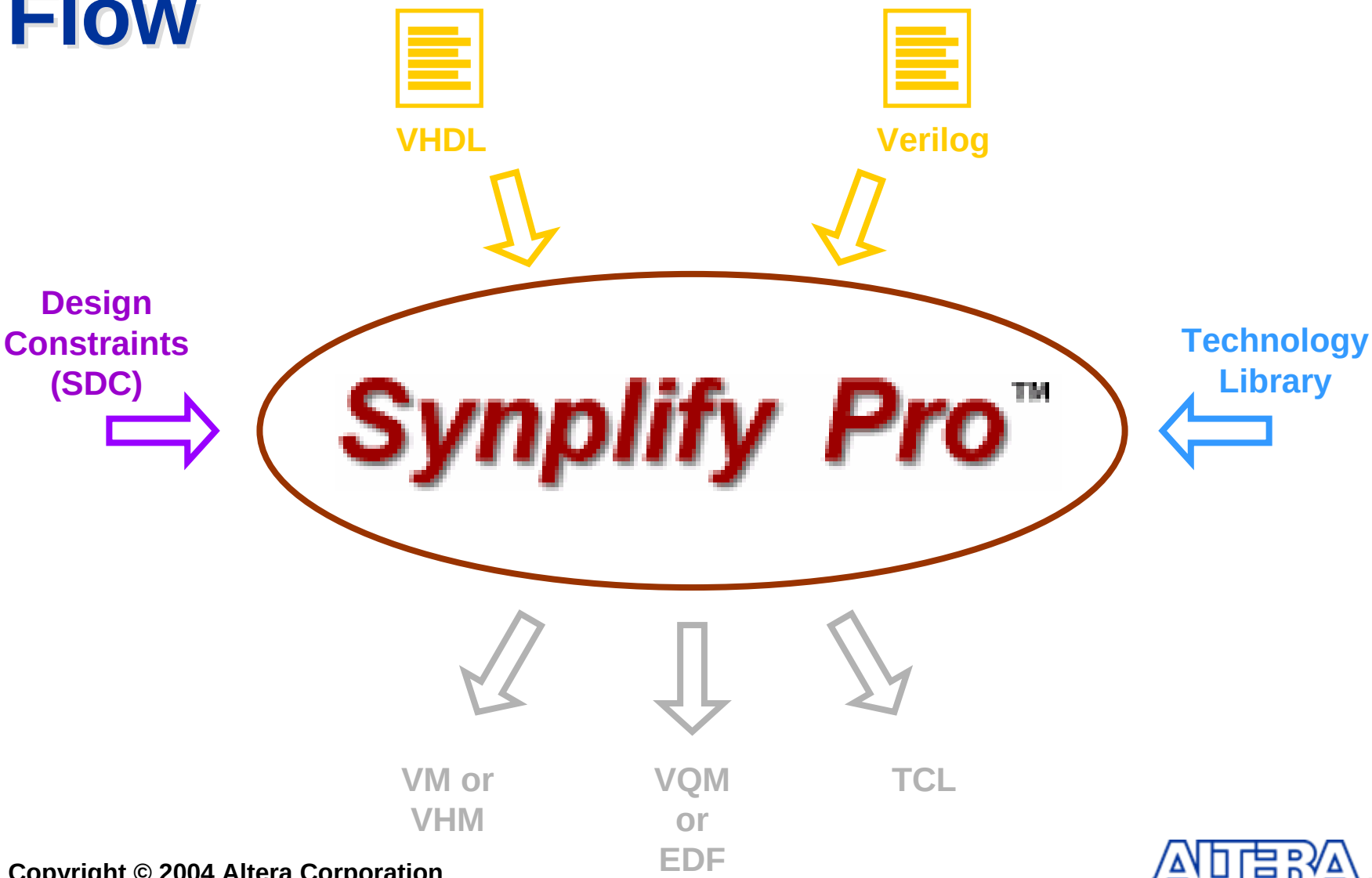
Copyright © 2004 Altera Corporation

Agenda

- ⑤Design Flow & Synthesis Overview
- ⑤Entering & Checking HDL Code
- ⑤Simple Synthesis Steps
- ⑤Using SCOPE Editor
- ⑤Timing Constraints

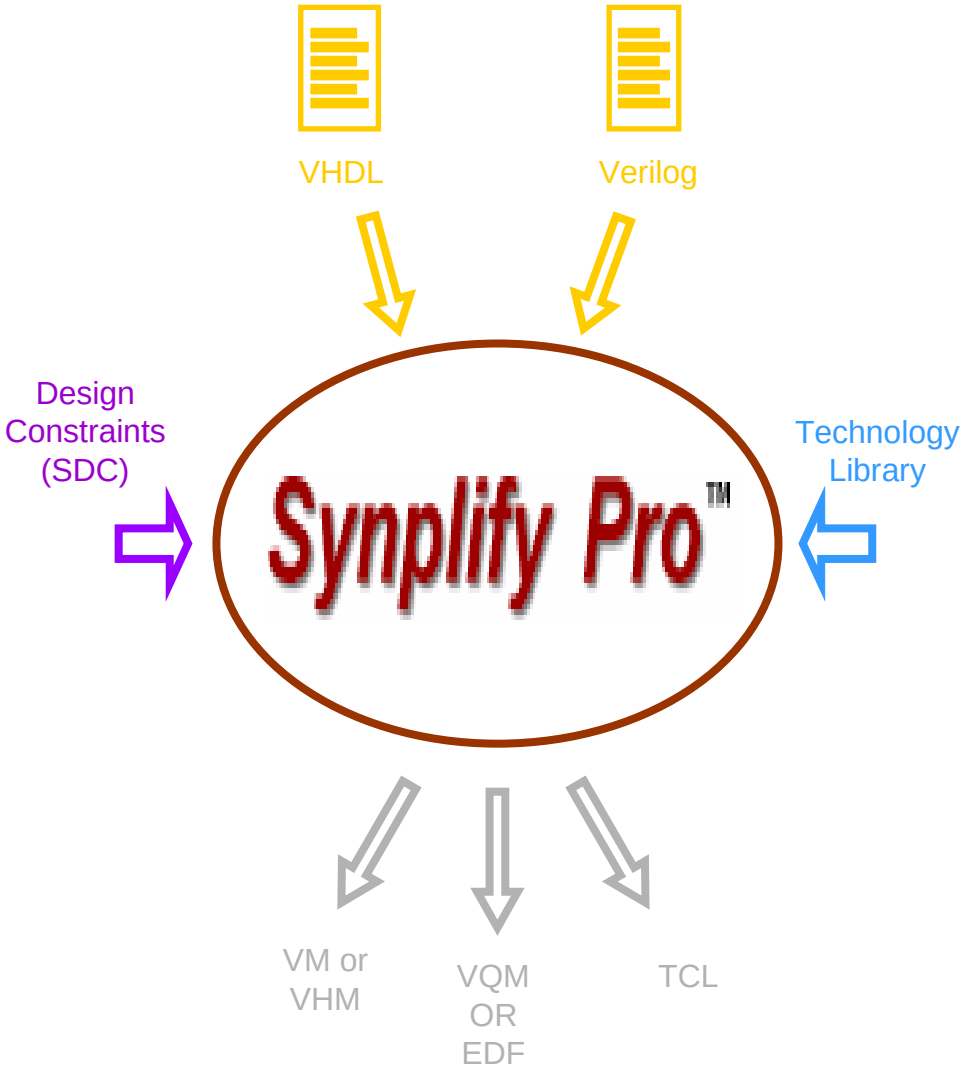
Copyright © 2004 Altera Corporation

Synplify Pro Input/Output File Flow



Copyright © 2004 Altera Corporation

Synplify Pro Inputs



- Design Files
  - VHDL
  - Verilog
- Technology Library
  - Building Blocks (Primitives) Specific to Targeted Device Family
- Design Constraints (SDC)
  - Timing Constraints
  - Synthesis Control Attributes

Copyright © 2004 Altera Corporation

The Synthesis Process

- Compilation
  - Technology Independent
- Mapping
  - Technology Specific

Copyright © 2004 Altera Corporation

Compilation

- Syntax Checking of HDL Code
- HDL Conversion to Technology-Independent Netlist
- Extraction of High-Level Structures by B.E.S.T. Algorithms
  - RAM/ROM
  - Arithmetic & Logical Operators
- FSM Extraction & Optimization
- Resource Sharing
- RTL Optimizations

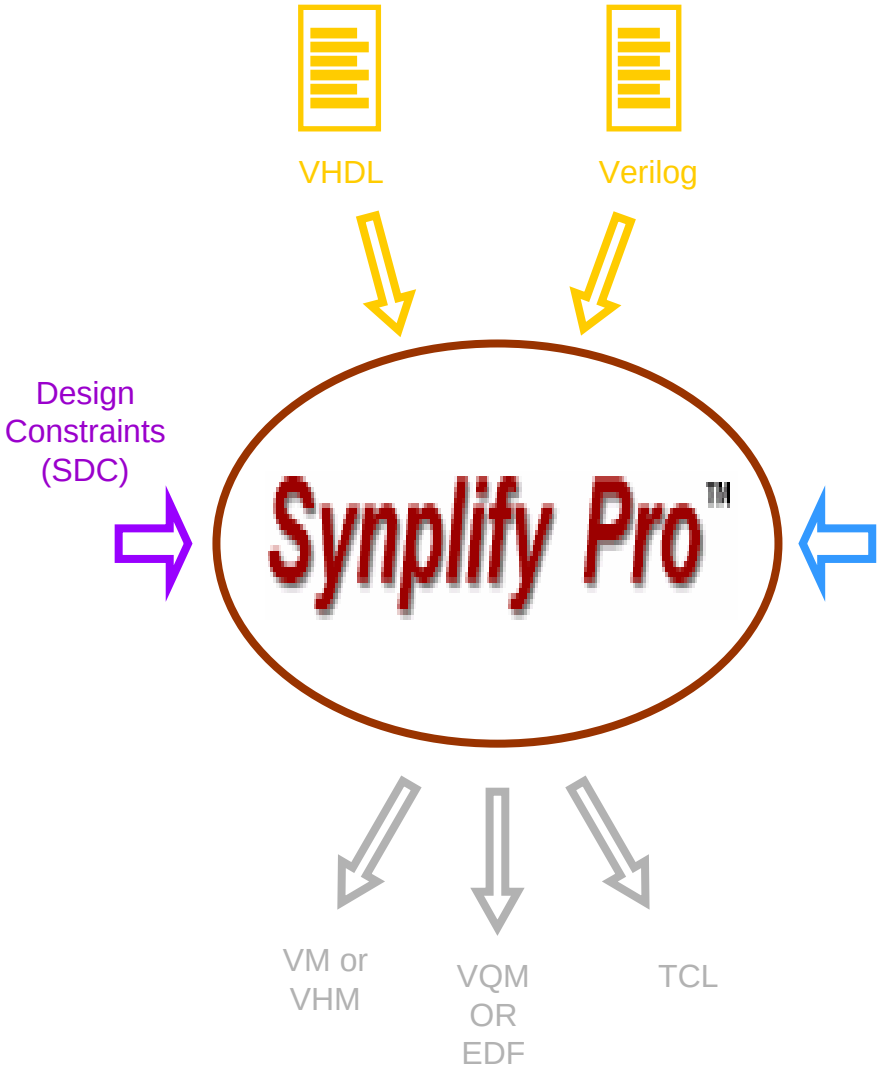
Copyright © 2004 Altera Corporation

Mapping

- Technology-Independent Netlist as Input
- Mapping to Technology-Specific Structures
  - Look-Up Tables
  - Registers
  - MAC's (Stratix/StratixII)
- B.E.S.T. Mapping to Technology Specific Resources
  - RAM/ROM ⇒ESBs
- Timing Driven Synthesis & Optimization
  - Logic Replication
  - Buffering

Copyright © 2004 Altera Corporation

Synplify Pro Outputs



- VQM or EDF File
  - Design Representation
  - Technology-Specific Netlist
  - VQM (Stratix, Stratix II, APEX II, All APEX, Mercury, Excalibur)
  - EDF (All FLEX, ACEX, MAX)
- TCL Files
  - Used with Execute & Control Quartus II
  - Used to Pass Design Constraints to Quartus II
  - More Details Discussed Later
- VM, VHM Files
  - Verilog & VHDL Post-Synthesis Netlist Files
  - For Post-Synthesis Simulation

Copyright © 2004 Altera Corporation

Synplify Pro Text Editor

Create New HDL File

- Open File
  - File -> New
  - File -> Open an Existing File
- Integrated with Synthesis Environment
  - Allows Cross-Probing with Log File
  - Allows Syntax & Synthesis Checking on the HDL Source Code
- Column Editing (Alt-Left Mouse Button)
- Block Commenting (Alt-C)
- Automatic Keyword Completion (Hit Esc Key)
- HDL Syntax Coloring
- Line Numbering

Copyright © 2004 Altera Corporation

Supported Verilog95 Constructs

- All operators

  (+, -, *, /, %, <, >, <=, >=, ==, !=, ===, !==, &&, ||, !, ~, &, ~&, |, ~|, ^~, ~^, ^, <<, >>, ?:, {}, {{}})

  [Note: / and % supported for compile-time constants and constant powers of 2]
- Behavioral statements

  if-else-if, case, casex, casez, for, repeat, while, forever, begin, end, fork, join
- Procedural assignments

  =, <= [Note: '<=' cannot be mixed with '=' for the same register]
- Compiler directives

  `define, `ifdef, `else, `endif, `include

Copyright © 2004 Altera Corporation

Supported Verilog2001 Constructs

- Additional constructs on top of Verilog95 support
- ANSI C style module declarations
  - Combining both port and data type declarations in the port list of modules, functions and tasks
- Sensitivity list
  - Can combine comma and “or” to separate signals
  - Can use @* or @(*) to include all signals in procedural block
- Signed arithmetic
  - *reg* and *net* data types can be declared using the reserved keyword *signed*
  - Signed operations can be performed for any vector length
  - Integer literals can also be specified using **‘sradix** or **size’s<radix>**

Copyright © 2004 Altera Corporation

More Supported Verilog2001 Constructs

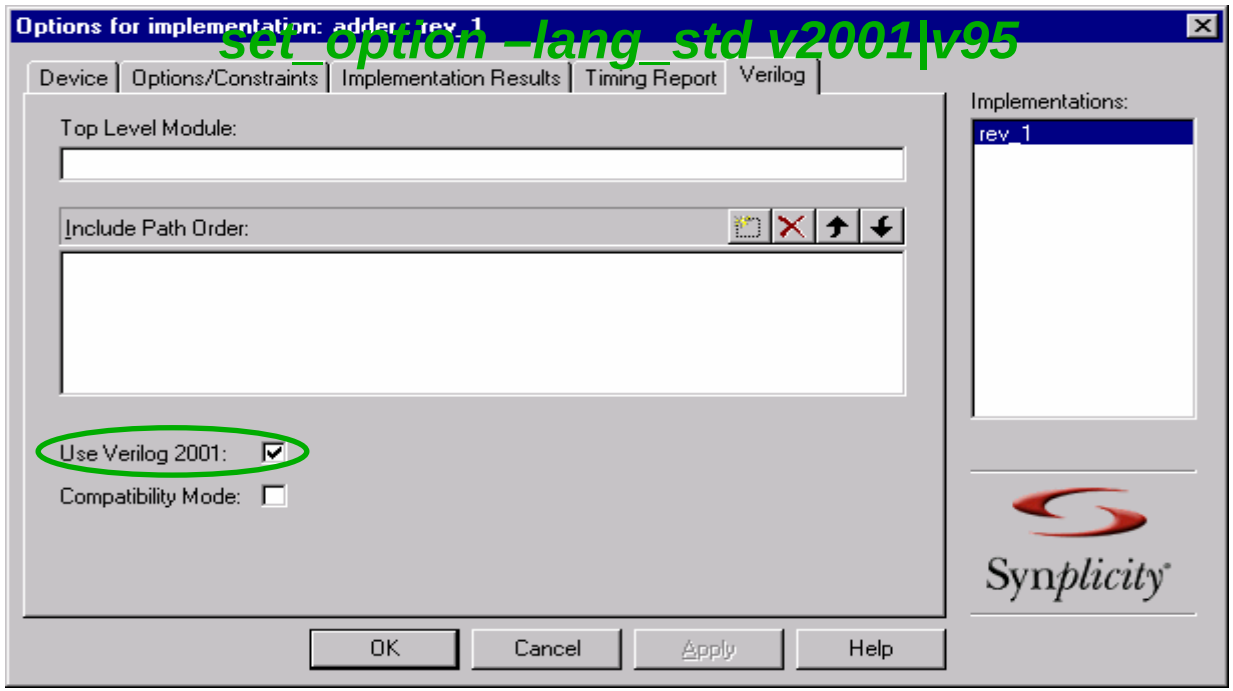
- Explicit inline parameter declaration
  - Assign parameter value by name in instantiation
  - No need to specify all parameter values
  - Parameters in any order
- $signed and $unsigned support
- ** operator supported only for base 2
- Generate loops and conditional generate

Copyright © 2004 Altera Corporation

Verilog 2001 Support in Synplify Pro

⑤Switch in Implementation Options

- ON by default for new projects
- Can be implemented in the project file as



Copyright © 2004 Altera Corporation

Parameter Override in Verilog

⑤Parameter override using # and ***defparam*** (down one level of hierarchy only)

Using "#" to override parameter

```
module sqrtb(z,a);
  parameter asize=4;
  output [(asize-1):0] z;
  input [(asize-1):0] a;
...
endmodule
```

```
module sqrterr(e, a);
  output [7:0] e;
  input [7:0] a;
...
  sqrtb #(8)  sq1(.z(e), .a(a));
```

Using "defparam" to override parameter

```
module sqrterr(e, a);
  output [7:0] e;
  input [7:0] a;
...
  sqrtb sq1(.z(e), .a(a));
  defparam sq1.asize = 8;
```

Copyright © 2004 Altera Corporation

Supported VHDL Standards

⑤ Synplify Pro software supports a synthesizable subset of VHDL'93 (IEEE 1076), and the following IEEE library packages:

- std_logic_1164
- numeric_std
- std_logic_unsigned
- std_logic_signed
- std_logic_arith

Note : The VHDL compiler follows the VHDL'93 coding style, this should be remembered especially during post-synthesis simulations.

Copyright © 2004 Altera Corporation

Supported VHDL Language Constructs

- Inputs, outputs and inouts of the following types
  - ***std_logic*, *std_logic_vector*, *integer*, *positive*, *signed*, *unsigned*, *records*, *arrays* & user-defined types**
- All operators
  - Logical

    AND, NAND, OR, NOR, XOR, XNOR
  - Arithmetic

    +, -,*, **, /, rem, mod
  - Relational

    >, <, =, >=, /=, <=
  - Shift Operators

    Shift_Left, Shift_right, Rotate_Left, Rotate_right,SHL, SHR, SHL, SHR
- Behavioral statements

  if-else-if, case, if-generate, for-loop, for-generate, when
- Assignments

  <=[for signals], :=[for variables]

Copyright © 2004 Altera Corporation

Generics

⑤Generics are supported

```
entity sqrtb is
  generic (asize : integer := 4);
  port( z : out bit_vector(asize-1 downto 0);
        a : in bit_vector(asize-1 downto 0));
end sqrtb;

architecture arch1 of sqrtb
…
end arch1;
```

```
entity sqrterr is
  port( e : out bit_vector(7 downto 0);
        a : in bit_vector(7 downto 0));
end sqrterr;

architecture arch2 of srterr is
  begin
  sq1 : sqrtb generic map (asize => 8)
                port map( z => e, a => a);
   …
end arch2;
```

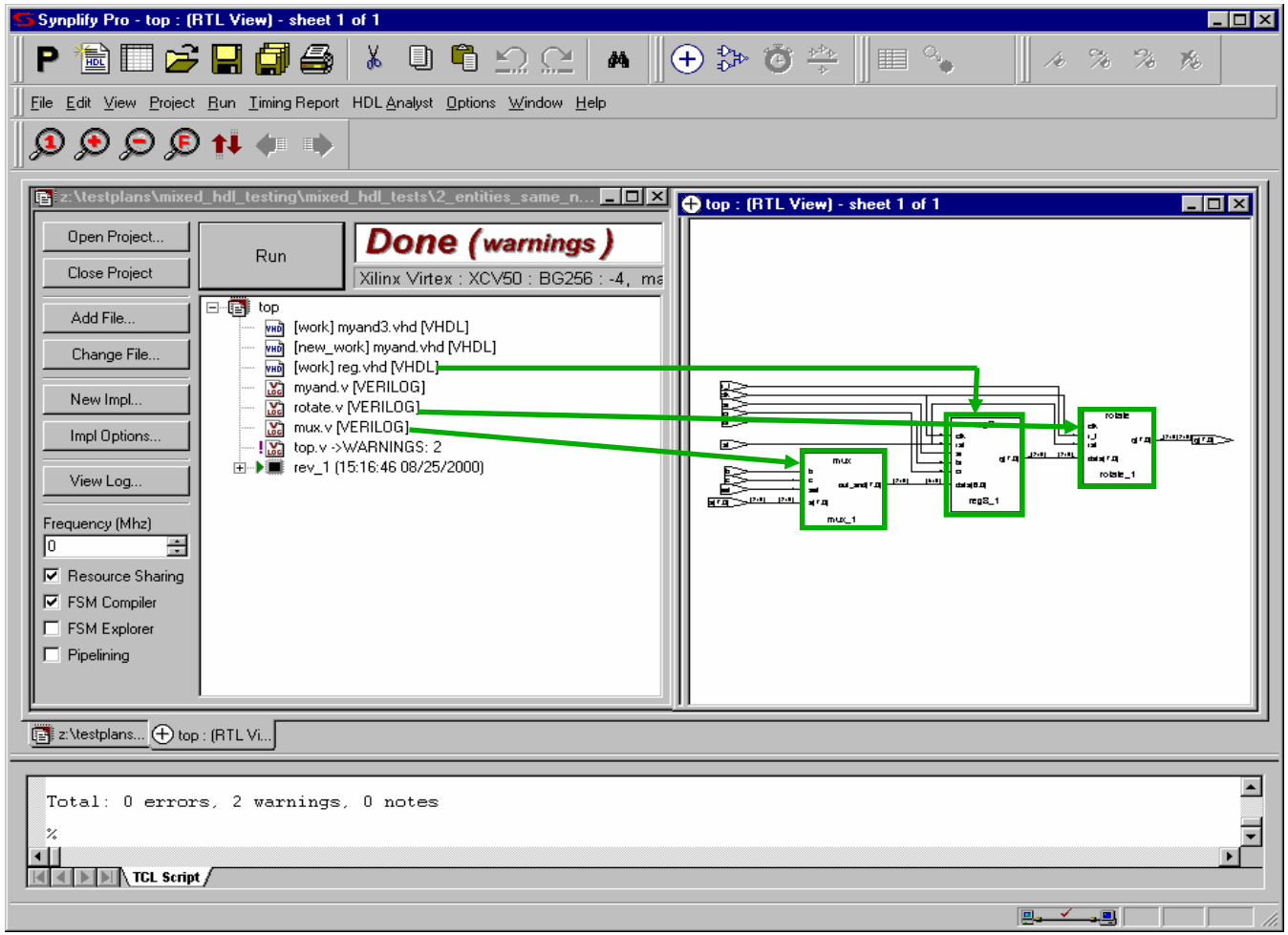
Copyright © 2004 Altera Corporation

Mixed Verilog/VHDL

- Mixed HDL design support
  - **Can add both Verilog and VHDL files to project**
- The top module/entity name has to be specified in the Implementation options
  - If a Verilog file instantiates a VHDL design, the Verilog language rules apply
  - Similarly if a VHDL file instantiates a Verilog design, the VHDL language rules apply
- Current limitations
  - **Generics or parameters cannot be passed across language boundaries**
  - **VHDL user-defined types for ports cannot be used across language boundaries**

Copyright © 2004 Altera Corporation

Mixed HDL Designs

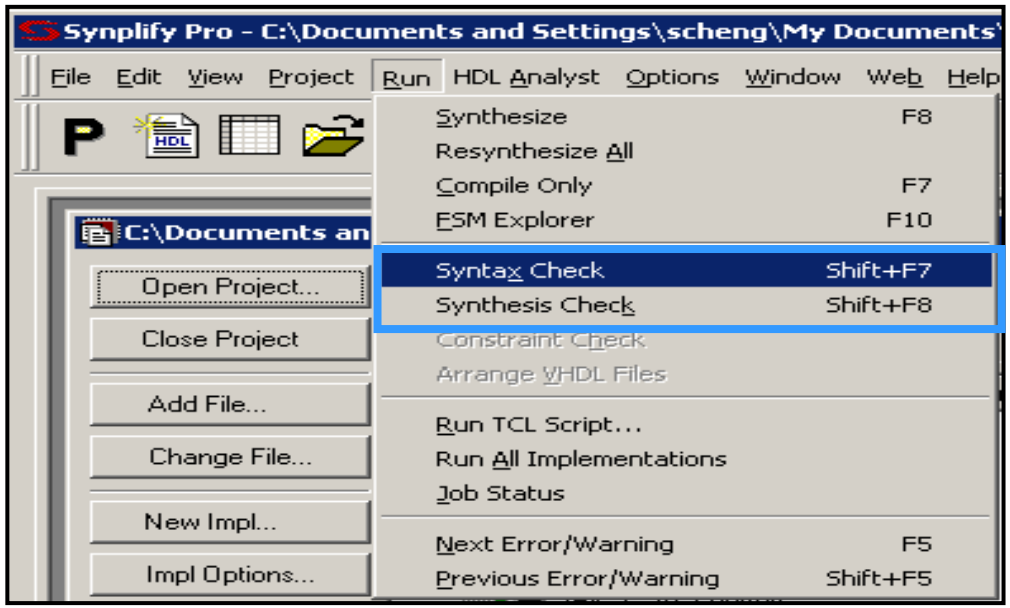


Copyright © 2004 Altera Corporation

Syntax & Synthesis Checking

- Syntax Check (Shift + F7)
  - Looks for Common Syntax Errors like Punctuation and Incorrect Keywords
- Synthesis Check (Shift + F8)
  - Looks for Hardware Related Errors like Incorrectly Coded Flip-Flops

Must Have Text File Open



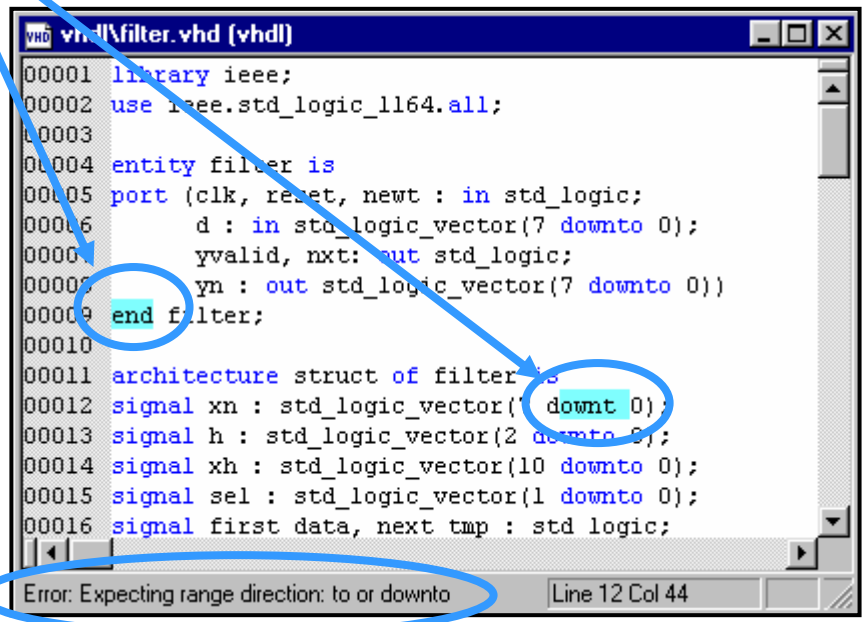
Select Syntax or Synthesis Check

Copyright © 2004 Altera Corporation

Syntax & Synthesis Error Location

Questionable Code Is Highlighted. Click on Highlighted Code &…

Bottom of Text Editor Window Displays Error Message Related to Highlighted Code



Copyright © 2004 Altera Corporation

Basic Synthesis Steps

1) Create Project
2) Add Input Files
3) Create or Edit Implementation
   - Pick Device
   - Global Clock Frequency
4) Click on the

Button
5) View/Analyze Results

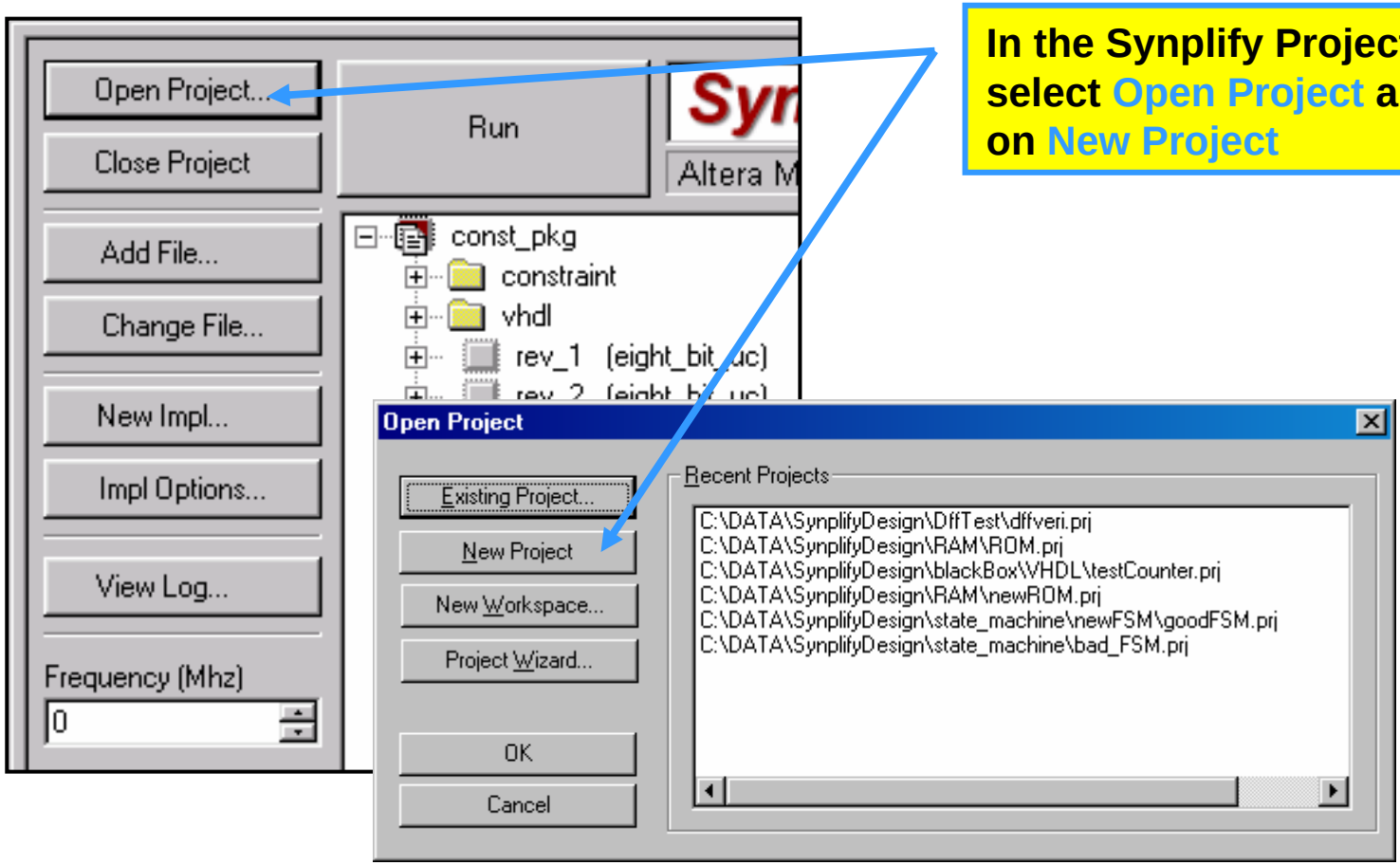
Copyright © 2004 Altera Corporation

1) Synplify Pro Projects

- Must Be Created to Perform Synthesis
- Contain Data Needed for a Particular Design
  - Source (Input) Files
  - Device Options
  - Constraint Files (SDC)
  - Multiple Implementations
  - Compiler & Mapper Options
  - Synthesis Results File
  - Global Frequency
- Are Stored in .PRJ Files
  - ASCII Text File Containing TCL Commands

Copyright © 2004 Altera Corporation

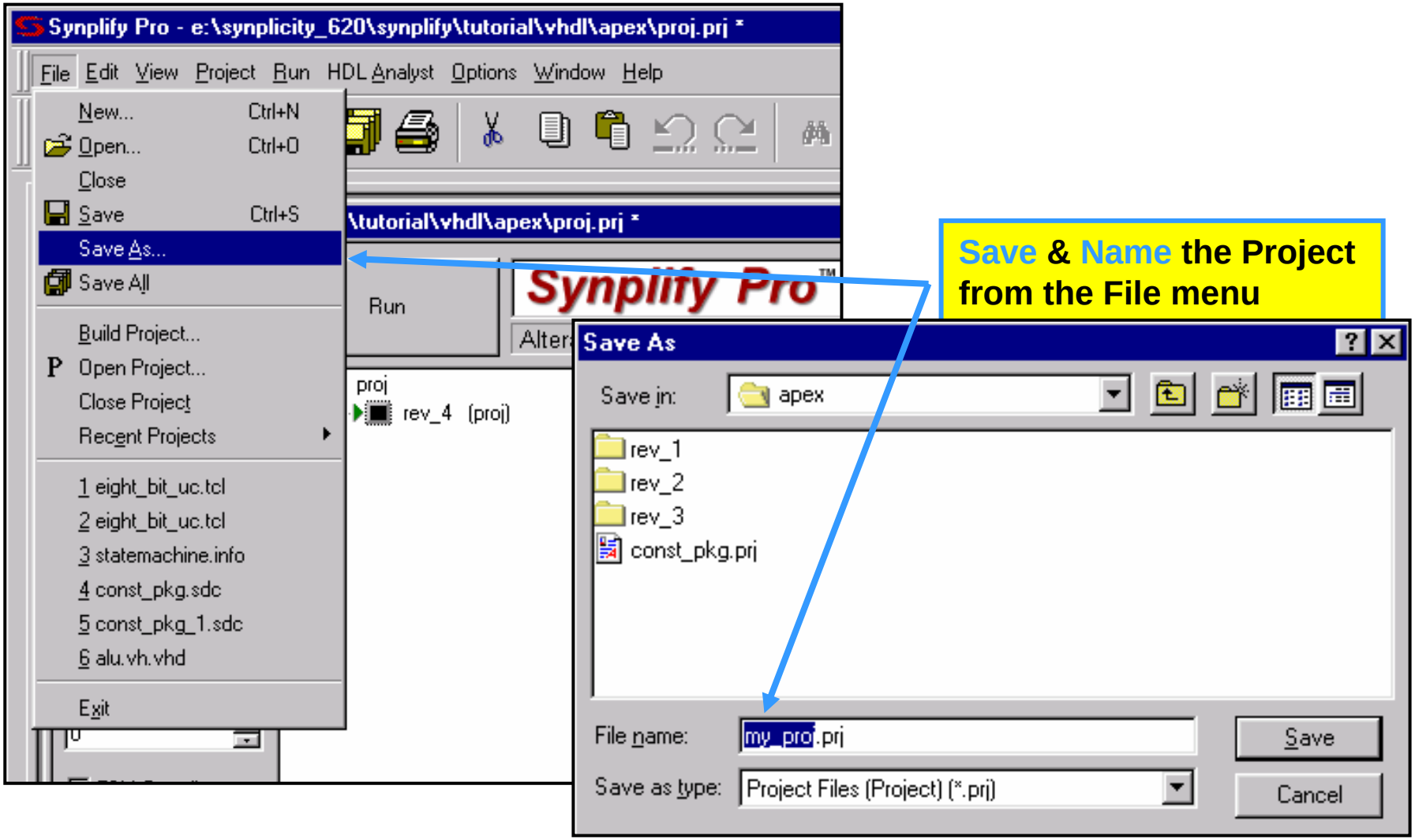
Creating New Projects



In the Synplify Project Window, select Open Project and then click on New Project

Copyright © 2004 Altera Corporation

Creating New Projects (cont.)

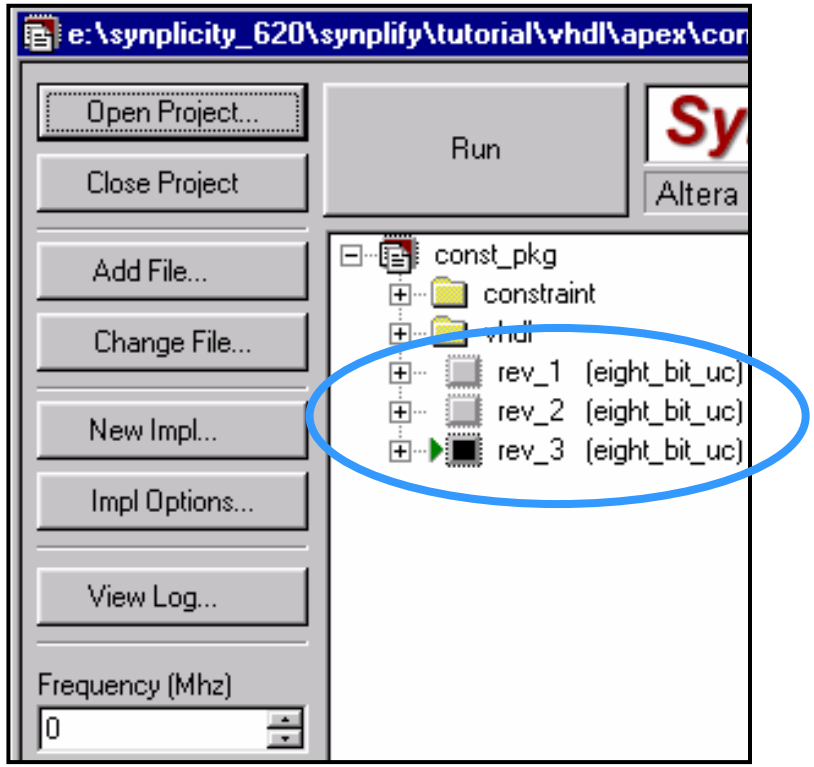


Save & Name the Project from the File menu

Copyright © 2004 Altera Corporation

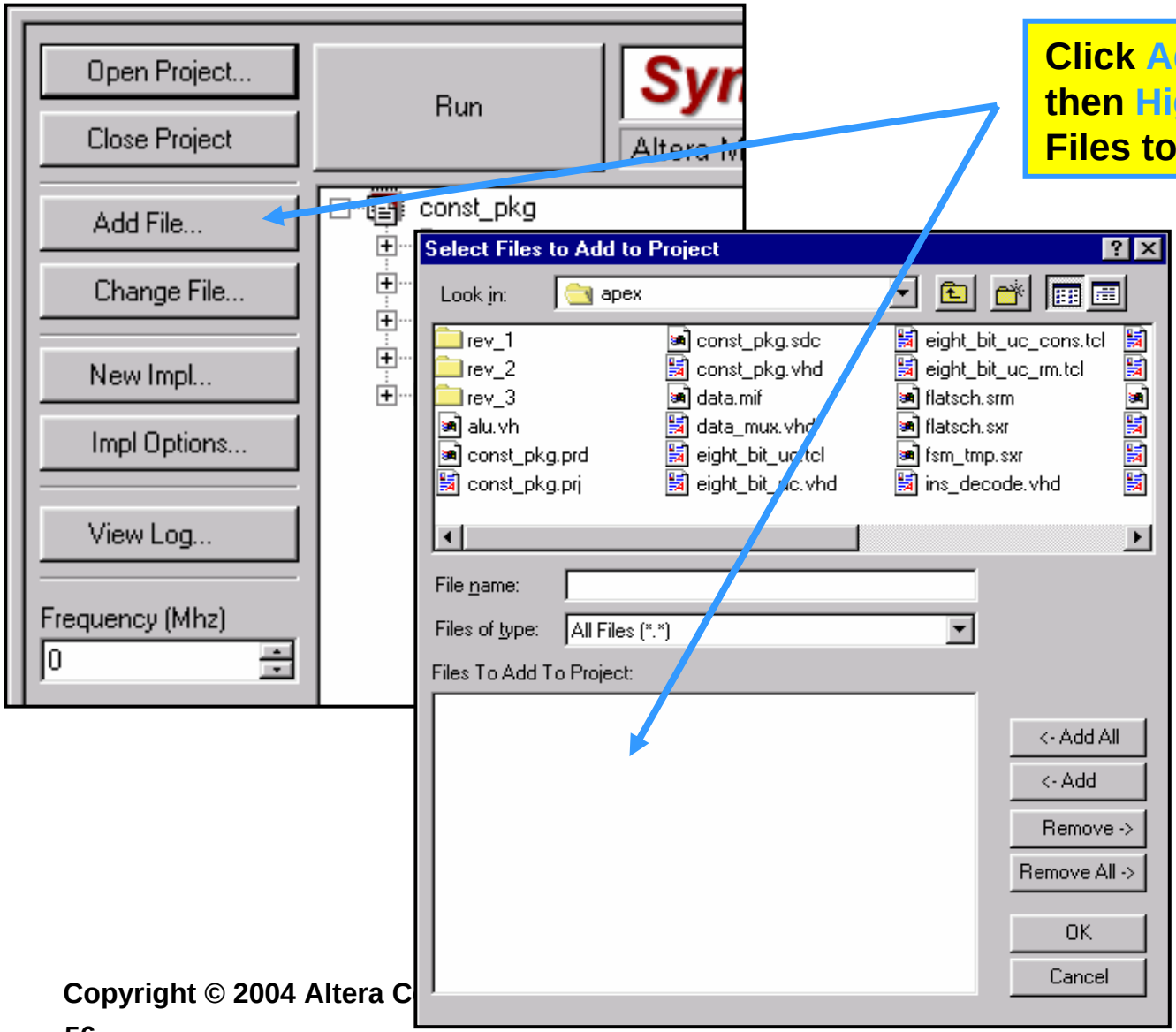
Project Implementations

- Different Versions of Same Project
- Each Can Have Different Synthesis Options & Constraints
  - Device & Speed Grade
  - Clock Frequency
  - Default State Machine Encoding
  - Top-Level Module
- Current Implementation is Highlighted



Copyright © 2004 Altera Corporation

2) Add Input Files



Click Add File Button then Highlight and Add Files to Window

Copyright © 2004 Altera C

HDL File Order

- ⑤ Verilog
  - Last File is Considered to be Top-Level Module
    - ③Order of Remaining Files Unimportant
  - Can Set Top-Level Module as an Implementation Option

- ⑤ VHDL
  - Packages Must Be Listed First
    - ③Order of Remaining Files Unimportant[1]
  - Last File is Considered to be Top-Level Module
  - Can Set Top-Level Module as an Implementation Option

Note:
1) If entity & associated architecture declarations are in separate files, then file containing entity declaration must be first

Copyright © 2004 Altera Corporation

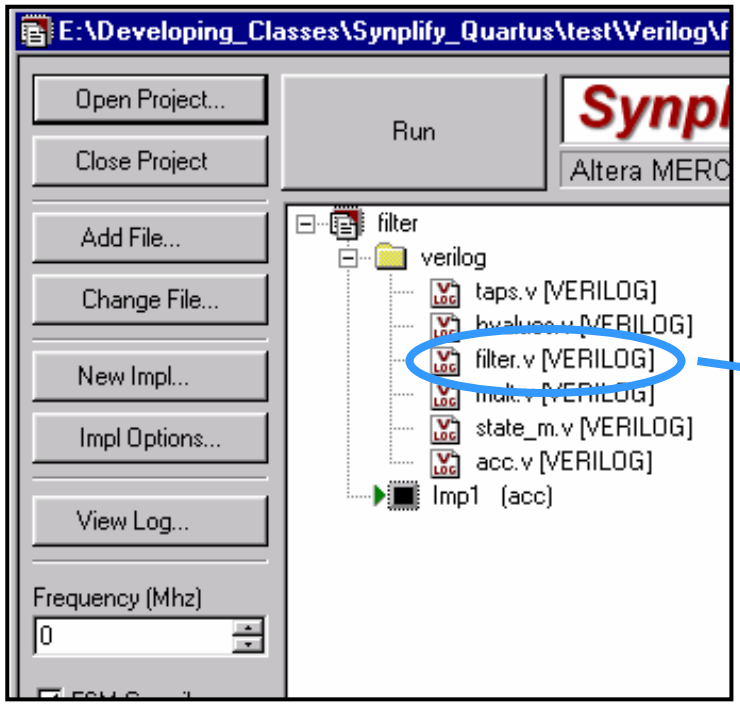
Mixed HDL Designs

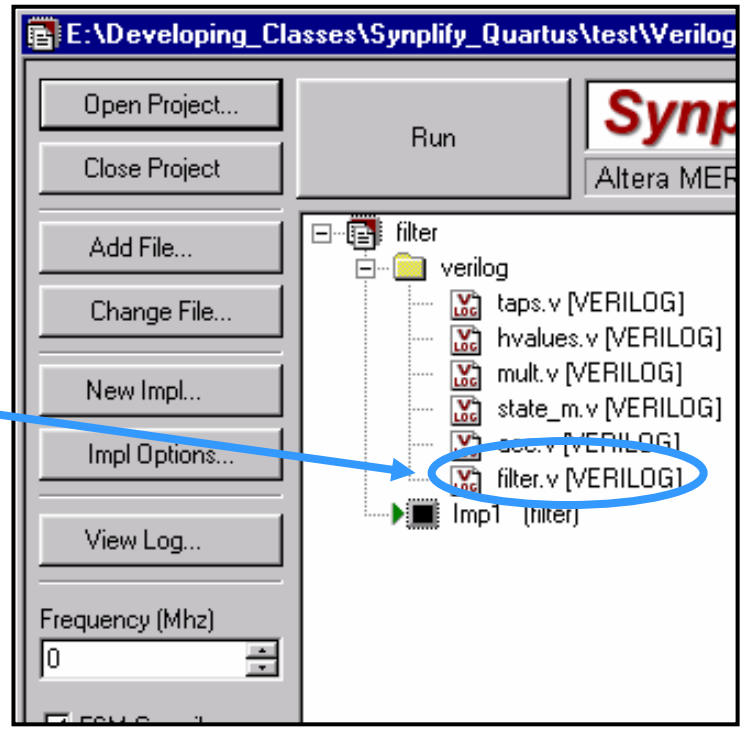
- For Mixed HDL Designs, Use Implementation Option Dialog Box to Set Top-Level
- If a Verilog file instantiates a VHDL design, the Verilog language rules apply.
- If a VHDL file instantiates a Verilog design, the VHDL language rules apply
- Current limitations
  - Generics or parameters cannot be passed across language boundaries
  - VHDL User defined types for ports cannot be used across language boundaries

Copyright © 2004 Altera Corporation

Re-Ordering Files

Just Click & Drag Top-Level File to Bottom of List

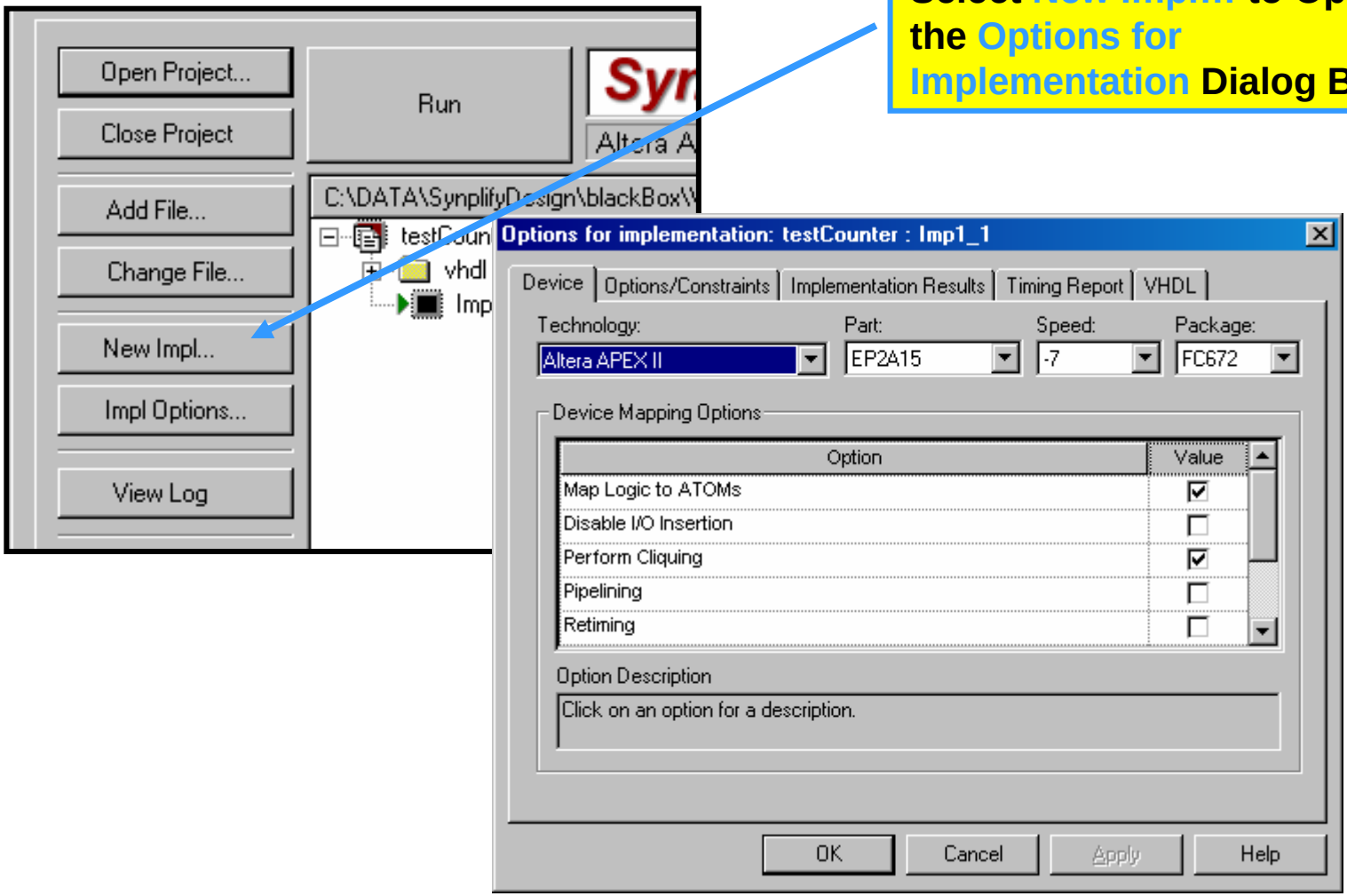




Copyright © 2004 Altera Corporation

3) Create Implementation

Select New Impl… to Open the Options for Implementation Dialog Box



Copyright © 2004 Altera Corporation

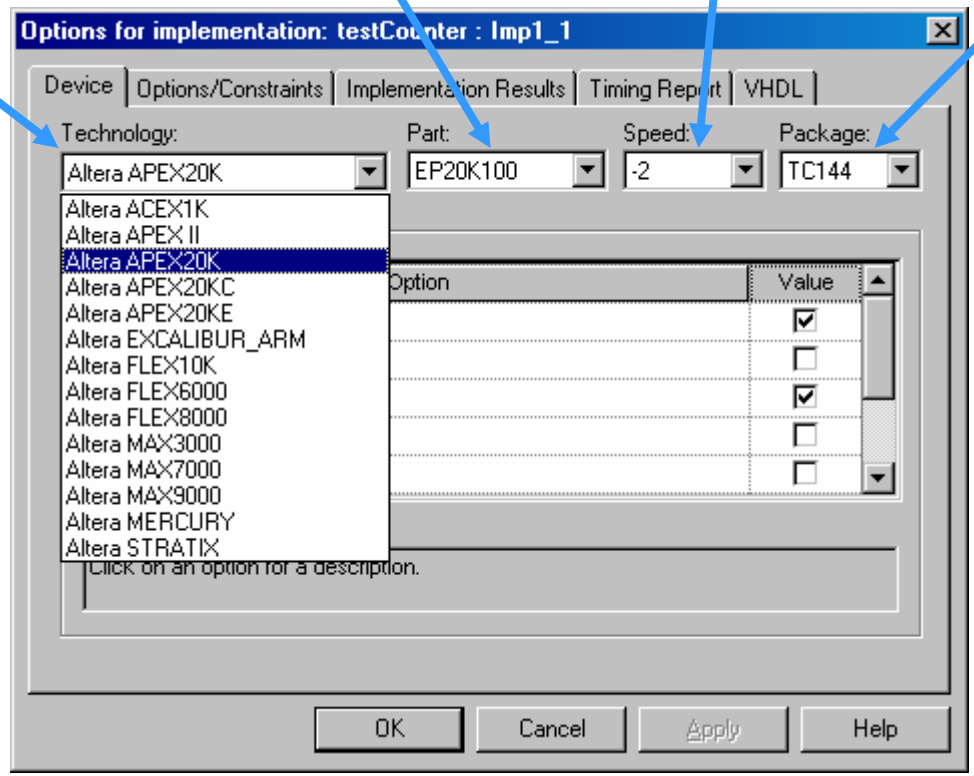
Device Tab ⇒ Select Device

Choose Device Family

Part #

Speed Grade

Package & Pin Count



Copyright © 2004 Altera Corporation

Device Tab ⇒ Device Mapping Options

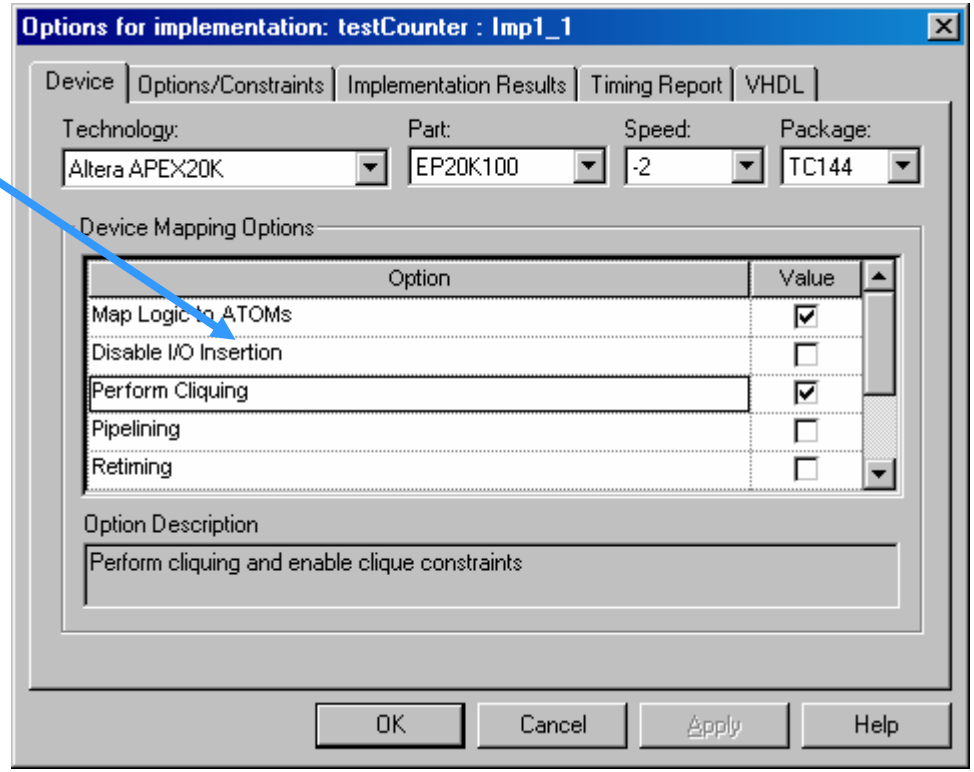
Device Mapping Options

These Options Are Specific to the Targeted Technology. Examples:

Map Logic to ATOMs – Produces ATOM Level Netlist so Quartus II Does Not Have to Perform Synthesis. Recommendation: Leave Checked.

Disable I/O Insertion – Inserts I/O (Input/Output) Pads into Netlist File. Unnecessary in Altera Designs. Recommendation: Leave Checked.

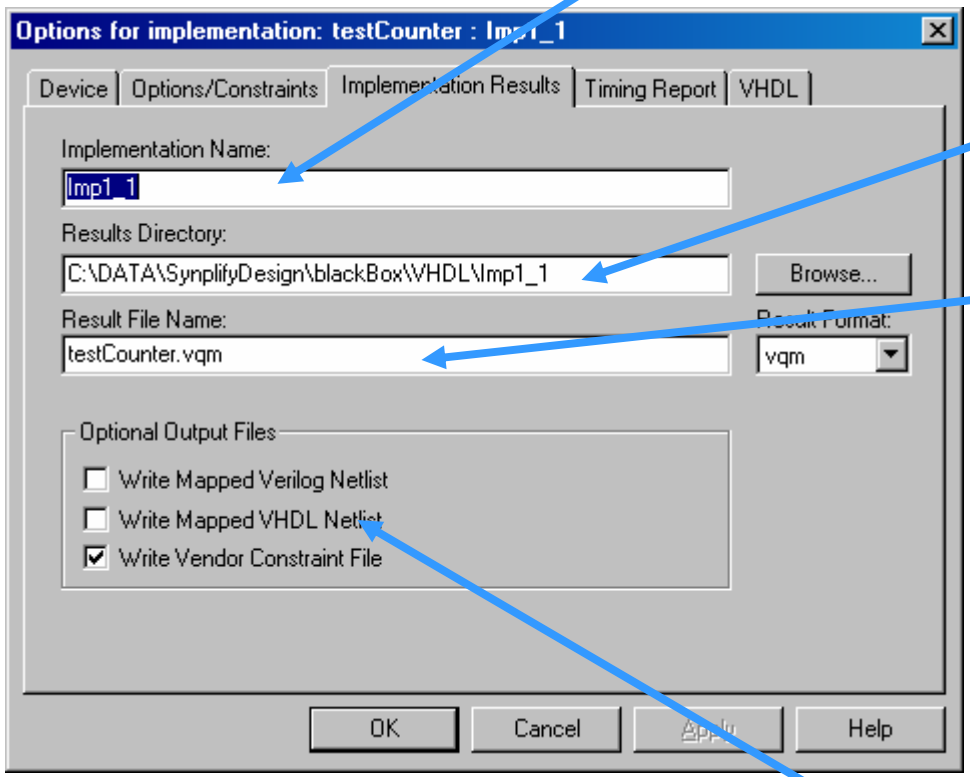
Perform Cliquing – Allows Synplify Pro to Make Grouping Assignments for Place & Route. Recommendation: Only Useful for FLEX 6000.



Copyright © 2004 Altera Corporation

Implementation Results Tab

Name Implementation



Specify the Directory for Output Files

Choose Output File Name…Leave File Format as Default for the Device You Selected

Mapped Verilog Netlist (VM) & **Mapped VHDL Netlist (VHM)** – Post Synthesis, Pre-Place & Route Simulation Netlist

Vendor Constraint File – Allows Synplify Pro to Pass Place & Route to Quartus II

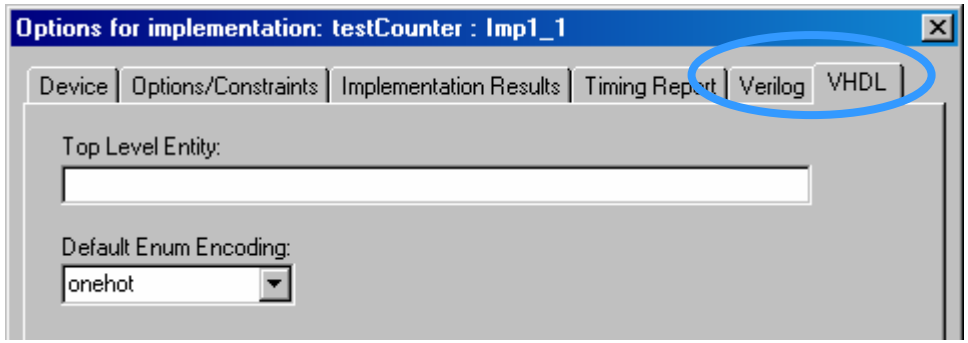
Select Optional Output Files

Copyright © 2004 Altera Corporation

Verilog/VHDL Tabs

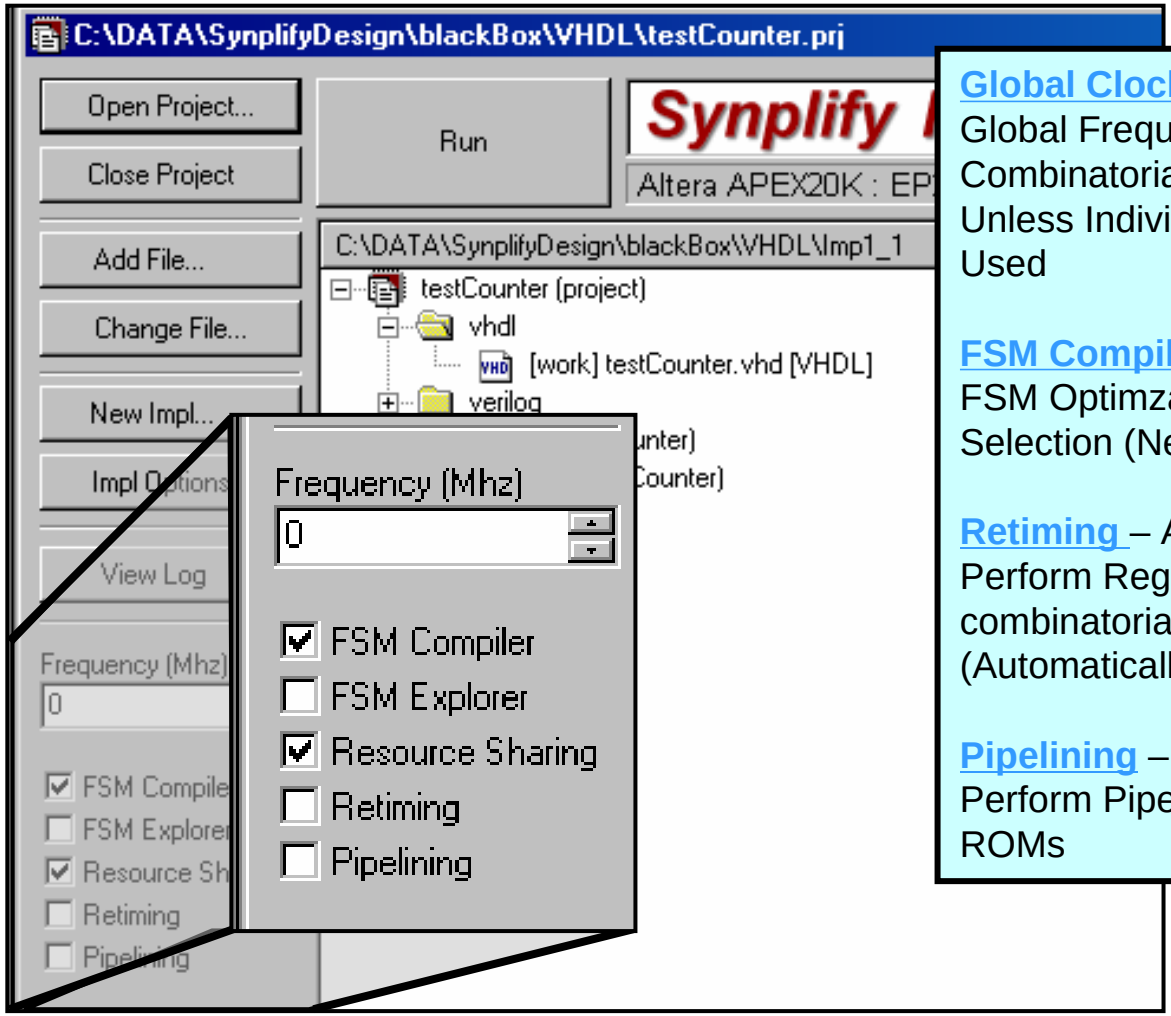
- ⑤ Verilog & VHDL
  - Fill in Top-Level Module
    - ③If Top-Level File Is Not at the Bottom of the File List
    - ③If Top-Level of Mixed HDL Design

- ⑤ VHDL Only
  - Specify Enumerated Date Type Encoding
    - ③Not for State Machines (Discussed Later)



Copyright © 2004 Altera Corporation

Final Basic Synthesis Controls



Global Clock Frequency – Sets Global Frequency to Be Applied to All Combinatorial & Registered Paths Unless Individual Clock Settings Are Used

FSM Compiler/Explorer – Enables FSM Optimzation & Encoding Selection (Next Slides)

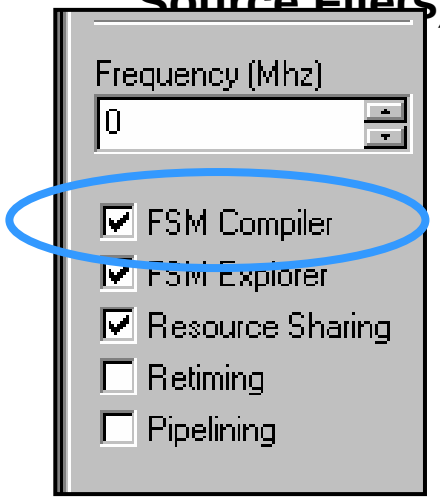
Retiming – Allows Synplfy Pro to Perform Register Balancing across combinatorial elements. (Automatically enables Pipelining)

Pipelining – Allows Synplfy Pro to Perform Pipelining on Multipliers & ROMs

Copyright © 2004 Altera Corporation

FSM Support – FSM Compiler

- ⑤ **FSM Compiler**
  - **Extracts State Machines from HDL Code & Instantiates an FSM Primitive**
  - **Uses Algorithms Specific to Optimizing State Machines**
  - **Removes Redundant & Unreachable States**
  - **Selects Encoding Based on Number of States**
  - **Allows Various Encoding Styles to be Selected without Changing Source File(s)**



Synplify Pro Project Window

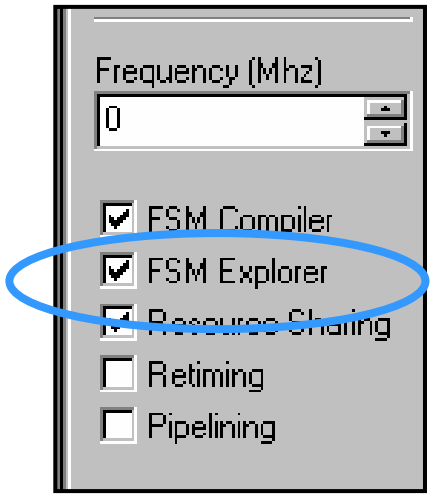
Number of States	Encoding Style
0 – 4	Sequential
5 – 24	One-Hot
24 and Up	Gray

Copyright © 2004 Altera Corporation

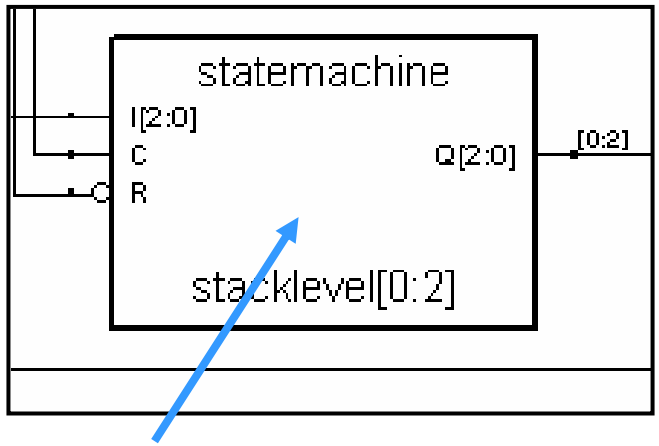
FSM Support – FSM Explorer

⑤ FSM Explorer

- Analyzes All Encoding Styles & Selects the Best Option Based on Other Design Constraints
  - ③Ignores User-Encoded FSM
- May Give Better Results that FSM Compiler but Takes Longer
- FSM Compiler Must Be Executed First



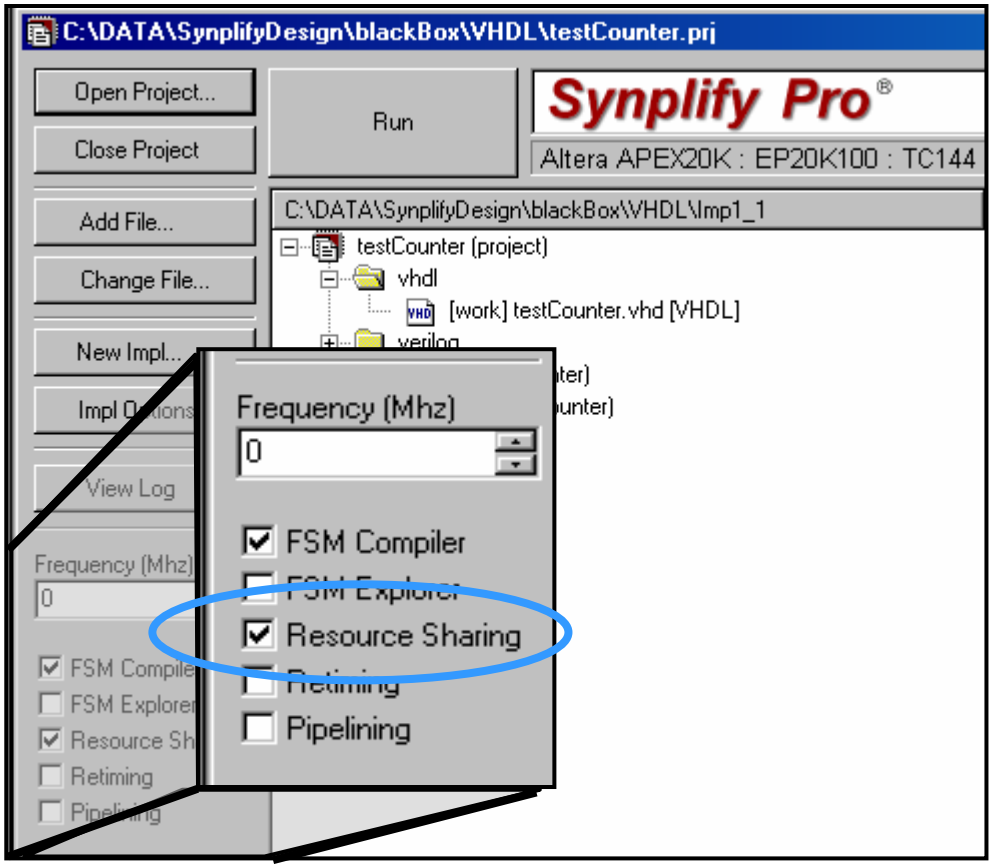
Synplify Pro Project Window



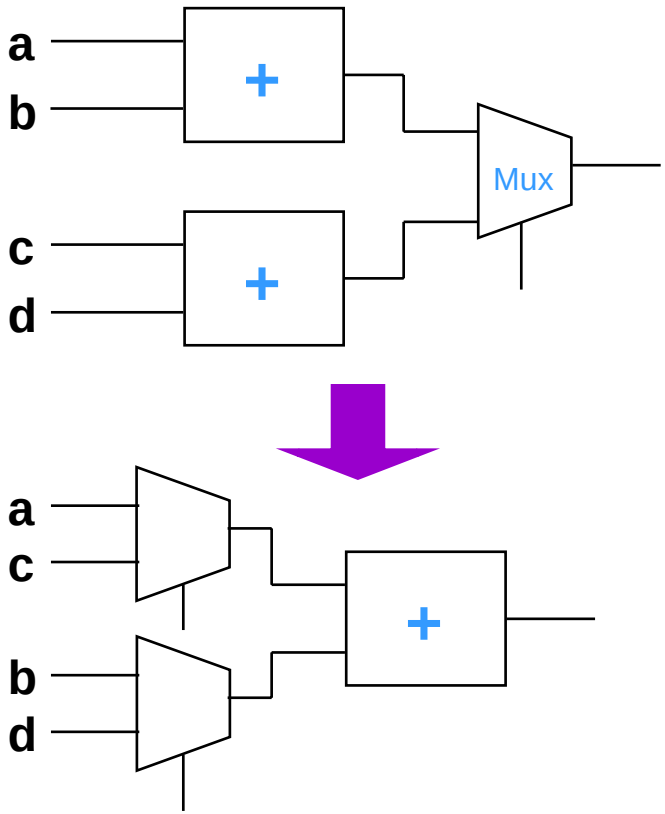
Extracted State Machine

Copyright © 2004 Altera Corporation

Final Basic Synthesis Controls (cont.)

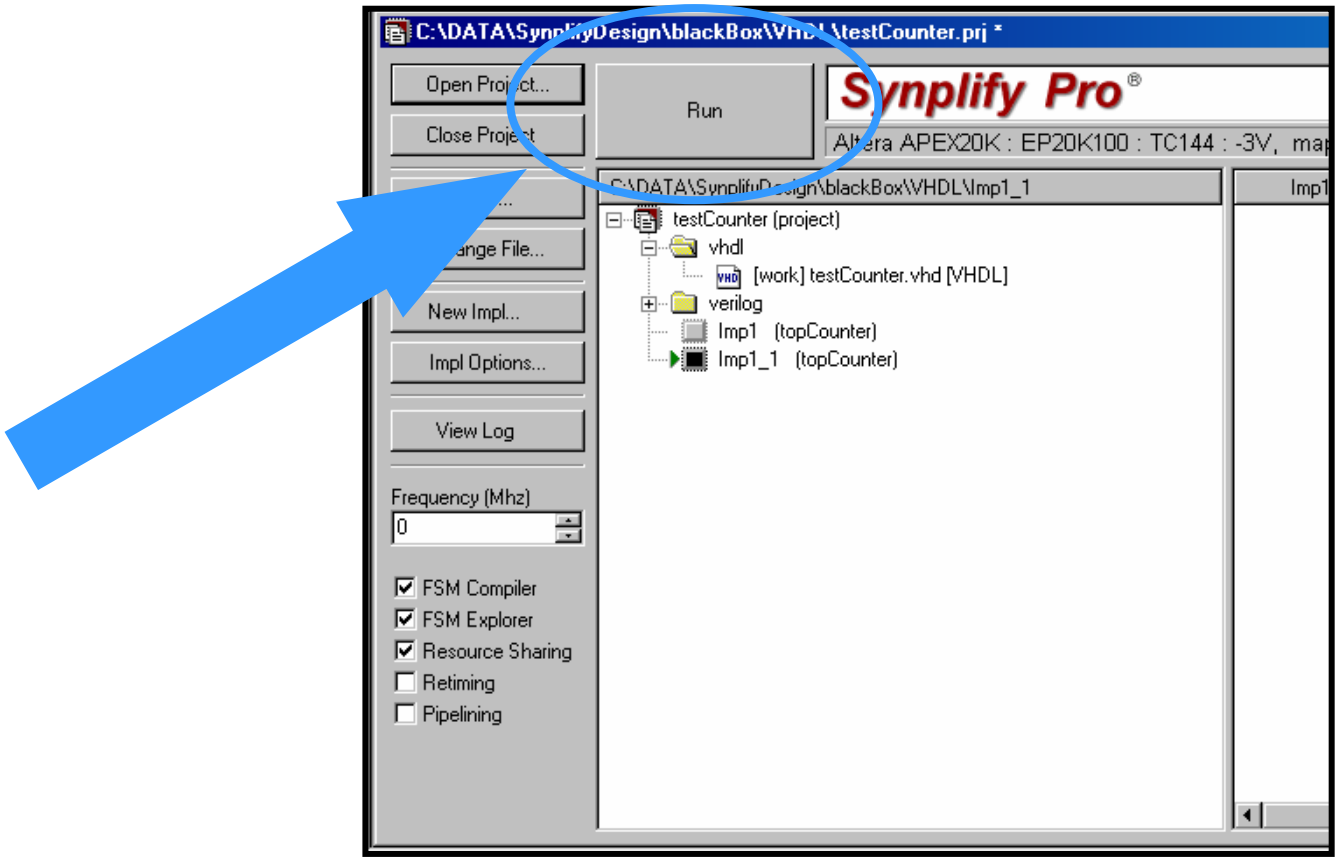


Resource Sharing – Uses Less Device Resources if Larger Logic Blocks (Arithmetic Operators) Can Be Shared



Copyright © 2004 Altera Corporation

4) Run Synthesis



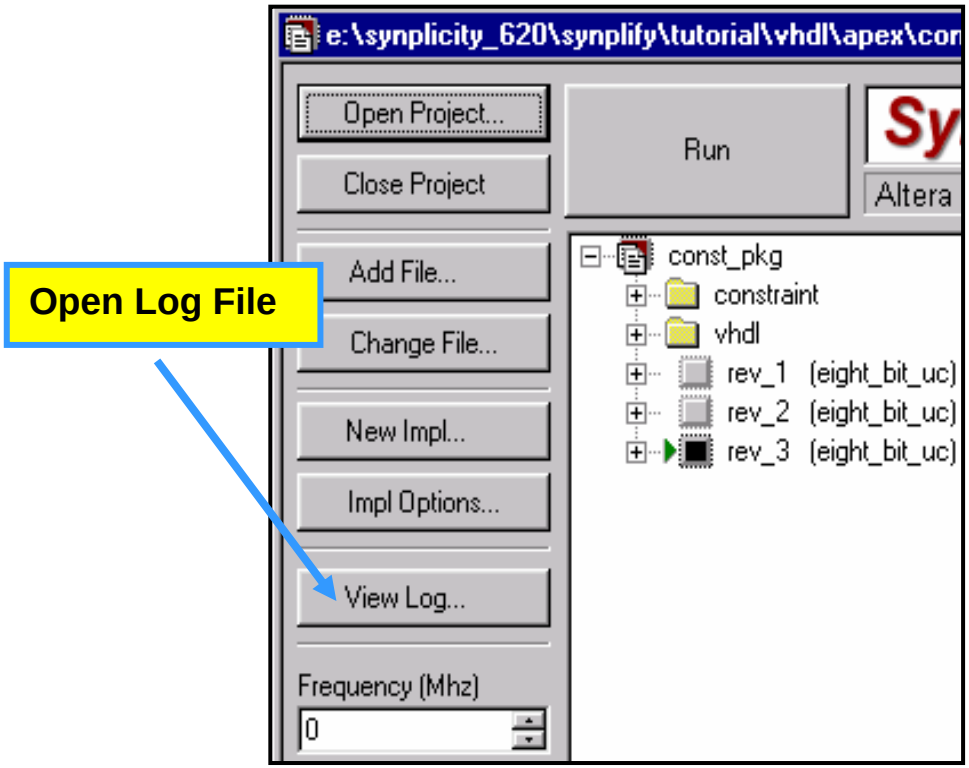
Copyright © 2004 Altera Corporation

5) Viewing/Analyzing Results

- Log File
- Log Watch Window
- HDL Analyst
  - RTL View
  - Technology View
- FSM Viewer

Copyright © 2004 Altera Corporation

Log File



- Sections
  - Compiler Report
  - Mapper Report
  - Timing Report
  - Resource Usage Report
- Messages
  - Errors (@E)
  - Warnings (@W)
  - Notes (@N)

Copyright © 2004 Altera Corporation

Log File – Compiler Report

- System & Software Information
- Project Information
  - Source Files
  - Top-Level Module
- Design Information
  - HDL Syntax & Synthesis Check
  - Unused Input Warnings
  - Redundant Logic Removal
  - RAM/ROM/FSM Extraction
  - Black Boxes

rev_5\filter.srr (log)

```
00001 $ Start of Compile
00002 #Thu May 17 16:42:18 2001
00003
00004 Synplicity Verilog Compiler, version 6.2.0, Build 083R, built Feb 28 20
00005 Copyright (C) 1994-2000, Synplicity Inc.  All Rights Reserved
00006
00007 @I::"e:\developing_classes\synplify_quartus\test\verilog\taps.v"
00008 @I::"e:\developing_classes\synplify_quartus\test\verilog\hvalues.v"
00009 @I::"e:\developing_classes\synplify_quartus\test\verilog\mult.v"
00010 @I::"e:\developing_classes\synplify_quartus\test\verilog\state_m.v"
00011 @I::"e:\developing_classes\synplify_quartus\test\verilog\acc.v"
00012 @I::"e:\developing_classes\synplify_quartus\test\verilog\filter.v"
00013 Verilog syntax check successful!
00014 File e:\developing_classes\synplify_quartus\test\verilog\taps.v changed
00015 Selecting top level module filter
00016 Synthesizing module taps
00017 Synthesizing module hvalues
00018 Synthesizing module state_m
00019 @N:"e:\developing_classes\synplify_quartus\test\verilog\state_m.v":18:1
00020 Extracted state machine for register filter
00021 State machine has 5 reachable states with original encodings of:
00022    000
00023    001
00024    010
```

Line 11 Col 1

Note: Green text indicates cross-probing to source code possible.

Copyright © 2004 Altera Corporation

Log File – Mapper Report

- Target Technology
- SCOPE Editor Information
  - Constraint Files Read
  - Attributes Assigned
- Design Information
  - Design Flattening (“Automatic Dissolve”)
  - Counter Extraction
  - FSM Implementation
  - Explicit & Inferred Clocks
  - Buffered Nets
  - Logic Replication

rev_5\filter.srr (log)

```
00032 Synplicity Altera Technology Mapper, version 6.2.0, Build 083R, built
00033 Copyright (C) 1994-2000, Synplicity Inc.  All Rights Reserved
00034 Automatic dissolve at startup in view:work.filter(verilog) of u5(acc)
00035 Automatic dissolve at startup in view:work.filter(verilog) of u4(mult
00036 Automatic dissolve at startup in view:work.filter(verilog) of u2(hval
00037 Encoding state machine work.state_m(verilog)-filter_h.filter[4:0]
00038 original code -> new code
00039    000 -> 00001
00040    001 -> 00010
00041    010 -> 00100
00042    011 -> 01000
00043    100 -> 10000
00044 Automatic dissolve during optimization of view:work.filter(verilog) c
00045 Found clock clk with period 10ns
00046
```

Line 40 Col 15

Copyright © 2004 Altera Corporation

Log File – Timing[1] Report

```
rev_5\filter.srr (log)
00049 Found clock clk with period 10ns
00050
00051          ##### START TIMING REPORT #####
00052
00053          Performance Summary
00054         **********************
00055
00056           Requested     Estimated     Requested     Estimated
00057 Clock      Frequency     Frequency     Period        Period        Slack
00058 ----------------------------------------------------------------------
00059 clk        100.0 MHz     95.2 MHz      10.0          10.5          -0.5
00060 ======================================================================
00061
00062          Interface Information
00063         ************************
00064
00065 Input Ports:
00066
00067 Port       Reference        User           Arrival     Required
00068 Name       Clock            Constraint     Time        Time         Slacl
00069 ----------------------------------------------------------------------
Line 268 Col 1
```

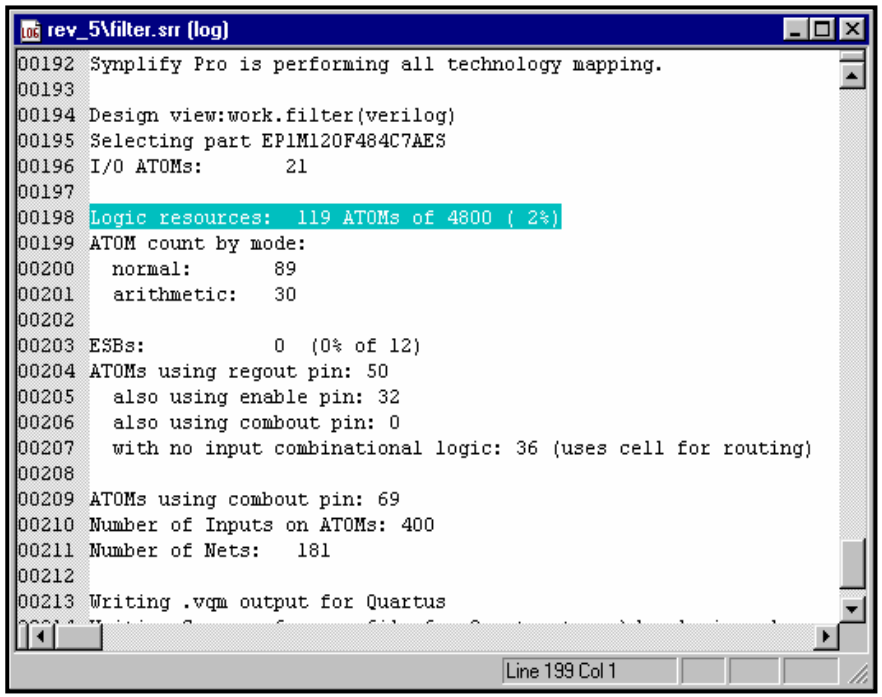
- ⑤ Required & Estimated Clock Frequencies
- ⑤ Required & Estimated Top-Level I/O Timing
  - Bi-directional Appears as Both Input & Output
- ⑤ Critical Path Analysis

Note:

1) Timing values are based on estimation. Actual timing will come from Quartus II after place & route.

Copyright © 2004 Altera Corporation

Log File – Resource Usage Report



- I/O Usage
- Logic Elements (LEs) Usage
- Embedded System Blocks (ESBs) Usage
  - RAM/ROM
  - Logic
- I/O Cell Register Usage
- Number of Counters
- Part/Package Information
- Device Utilization Percentage

Copyright © 2004 Altera Corporation

Please go to Exercise 1

Copyright © 2004 Altera Corporation

Exercise 1 Review

⑤ Easy Synthesis Steps

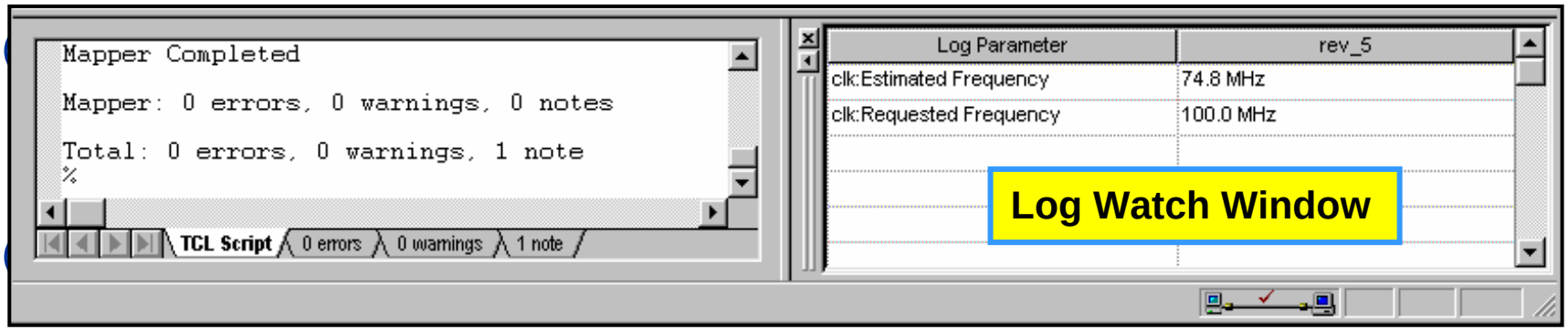
1. Create Project
2. Add Files
3. Create/Edit Implementation
   - Select Device
   - Enter Global Clock Frequency
4. Hit Run Button
5. View Log File

Copyright © 2004 Altera Corporation

Log Watch Window

⑤ Displays Log File Parameters in Tabular Format

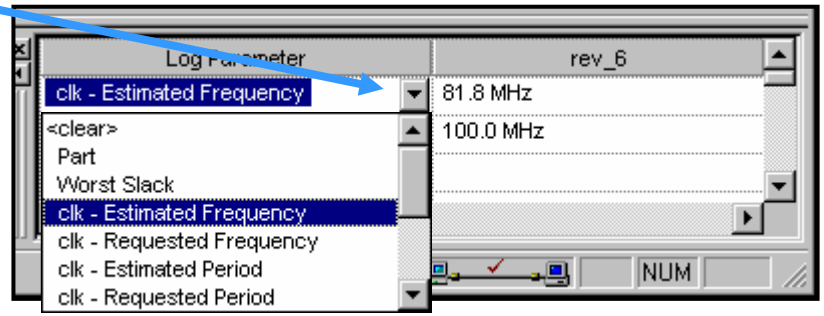
- Requested Clock Frequency
- Estimated Clock Frequency
- I/O Count

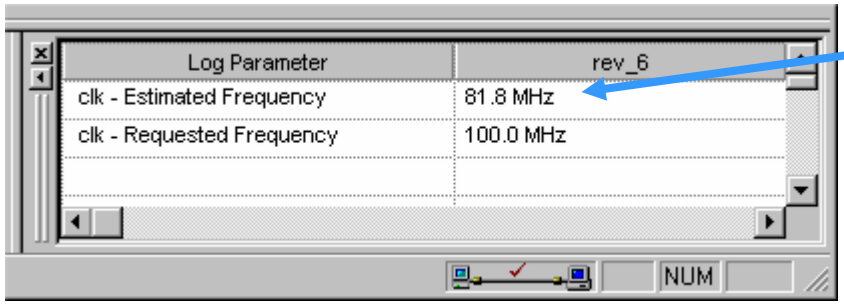


Copyright © 2004 Altera Corporation

Log Watch Window – Setting Parameters

Click Log Parameter box and then Use Drop-down Menu to Select Parameter

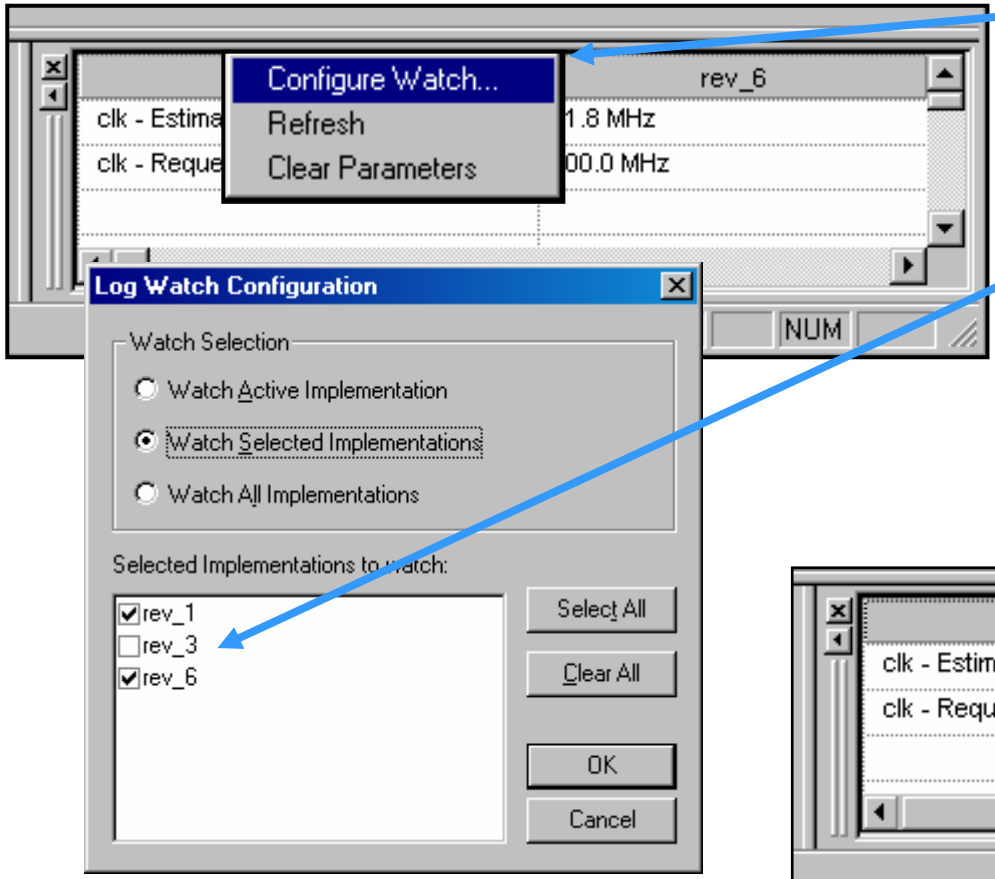




Values Updated after Synthesis

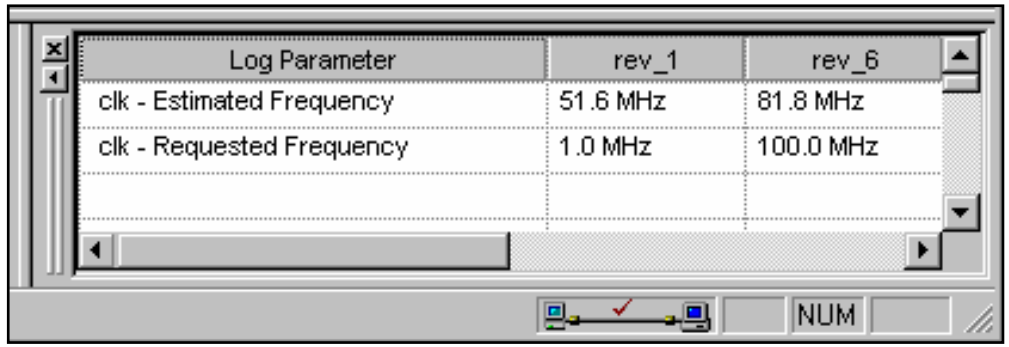
Copyright © 2004 Altera Corporation

Log Watch – Multiple Implementations



Right-Click on Column and Select Configure Watch

Choose Implementations to View and Click OK



Copyright © 2004 Altera Corporation

HDL Analyst

- ⑤ Graphical Representation of HDL Design
  - RTL View
    - ③User Sees Exactly What Is Written
    - ③High-Level, Technology Independent Schematic
      - Variable Width Adders
      - State Machines
      - RAM/ROM
  - Technology View
    - ③User Sees Exact Components Used to Implement Design
    - ③Low-Level, Technology Specific Schematic
      - LUTs
      - Flip-Flops
      - Muxes
    - ③Allows Critical Path Viewing & Timing

Copyright © 2004 Altera Corporation

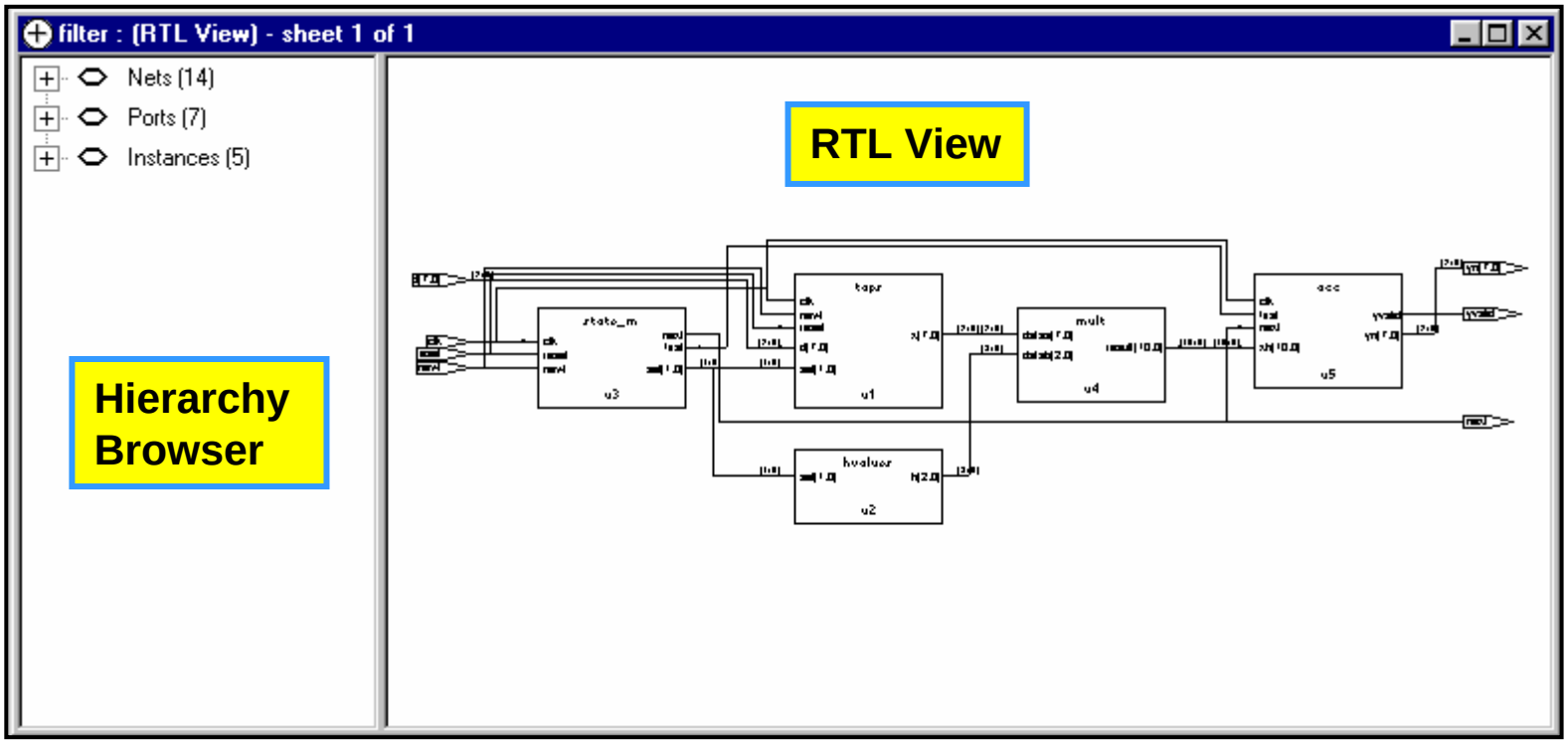
HDL Analyst Advanced Features

- Hierarchical and Flattened Views
- Advanced Filtering
- Crossprobing from Place and Route timing files
- FSM Viewer

Copyright © 2004 Altera Corporation

RTL View

Opens RTL View



Copyright © 2004 Altera Corporation

RTL View Features

- Hierarchy Browser
  - Breaks Down Design Using Symbols & Logic Names
- Push/Pop Control
  - Allows User to Enter & Exit Sub-Levels of Hierarchy
- Flattened Hierarchy
  - Displays Entire Design As Generic Logic Symbols
- Filter Schematic
  - Allows User to Isolate RTL View on Selected Component(s)
- Cross-Probing

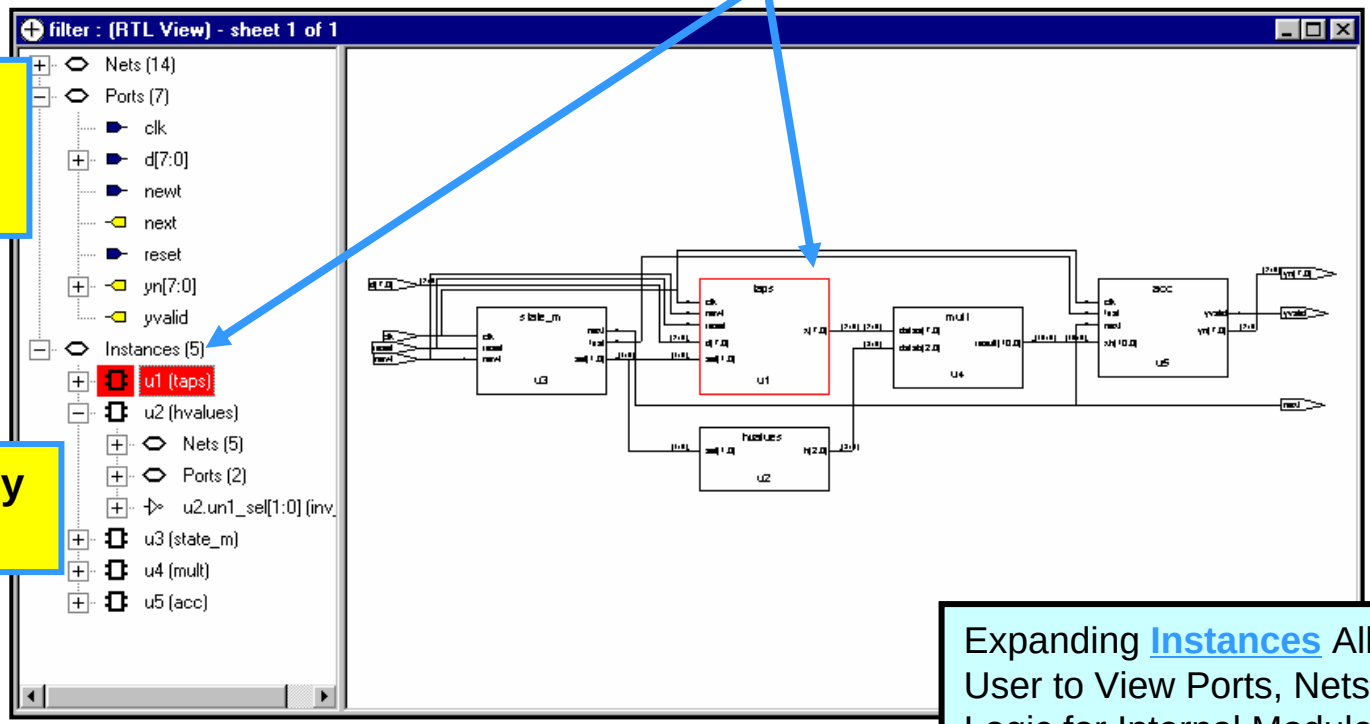
Copyright © 2004 Altera Corporation

RTL View - Hierarchy Browser

Highlighting a Net, Port, or Instance in the Hierarchy Browser Highlights/Views that Construct in the RTL View Window

Top-Level Port & Net List

Hierarchy Tree



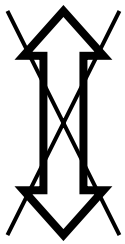
Expanding Instances Allows User to View Ports, Nets, & Logic for Internal Modules

Copyright © 2004 Altera Corporation

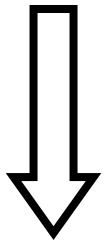
RTL View – Push/Pop

Push/Pop Button

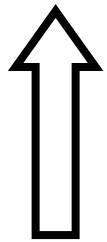
Mouse Pointers Below Indicate which Operation Will Be Performed by a **Left-Mouse Click** in the RTL View Window



No Hierarchical Boundary to Navigate



"Pushes" (Descends) Down into Selected Hierarchical Block



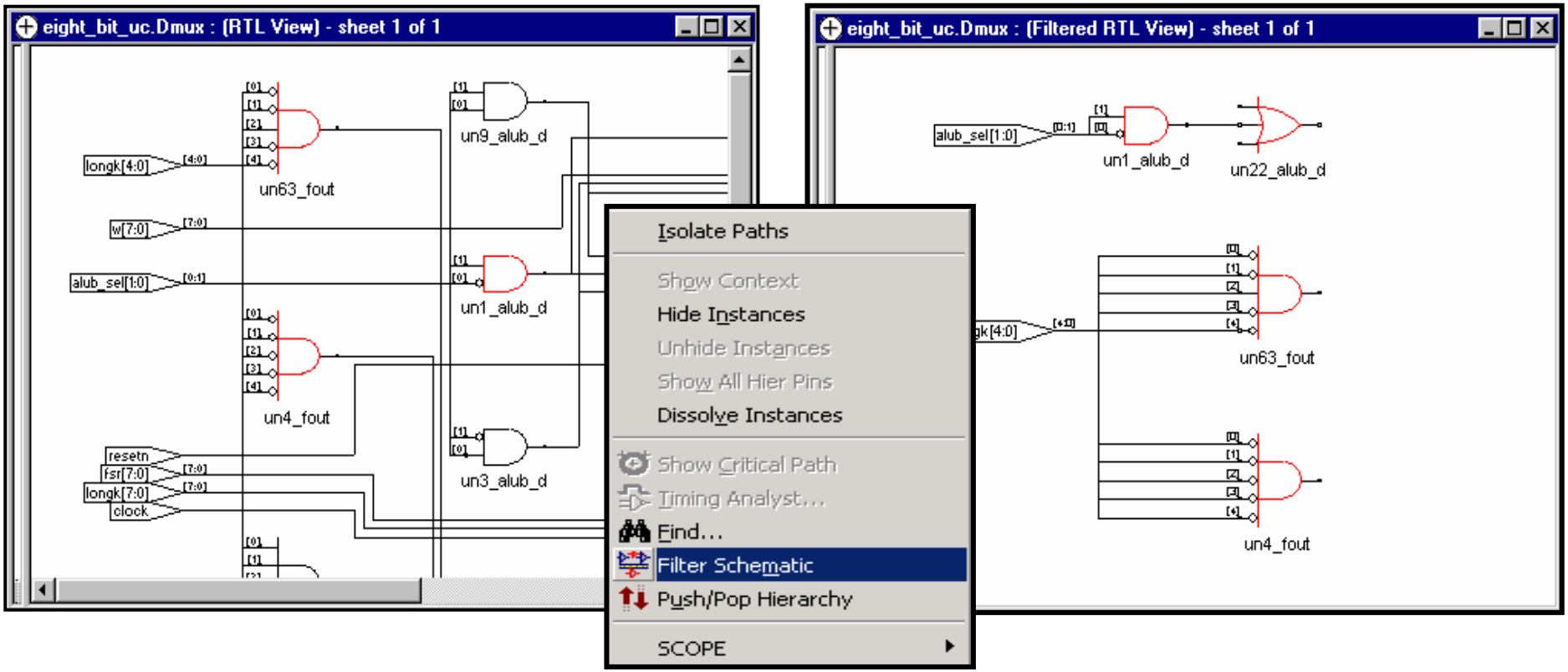
"Pops" (Ascends) Out of Current Hierarchical Block

Copyright © 2004 Altera Corporation

Filtered RTL View

Un-Filtered
User Can Select Components

Filtered
Only Selected Components Shown

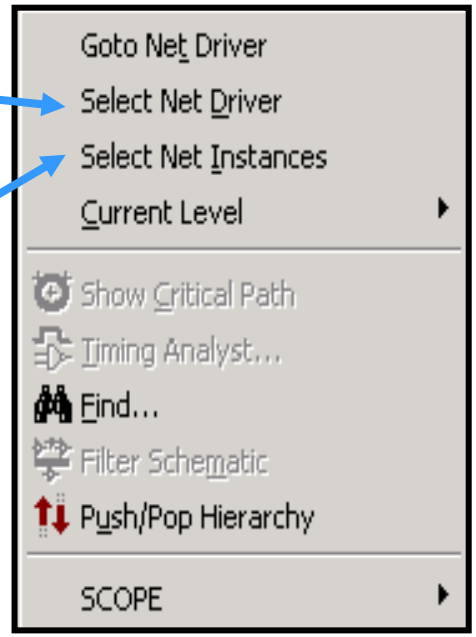


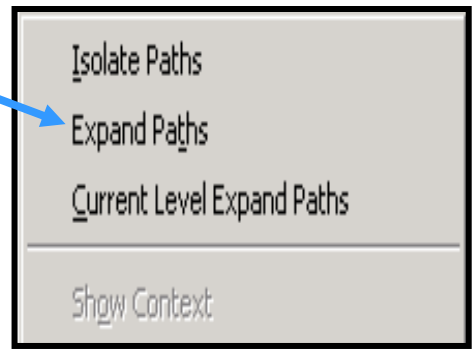
Right-Click for Pop-Up Window

Copyright © 2004 Altera Corpor

Other Filtered View Controls

- Select Net Driver
  - Adds to the Filtered View the Driver of the Selected Net (Fan-In)
- Select Net Instances
  - Adds to the Filtered View Everything Connected to Selected Net (Fan-Out)
- Expand Paths
  - Shows All Instances/Nets in the Path Between Two Selected Components

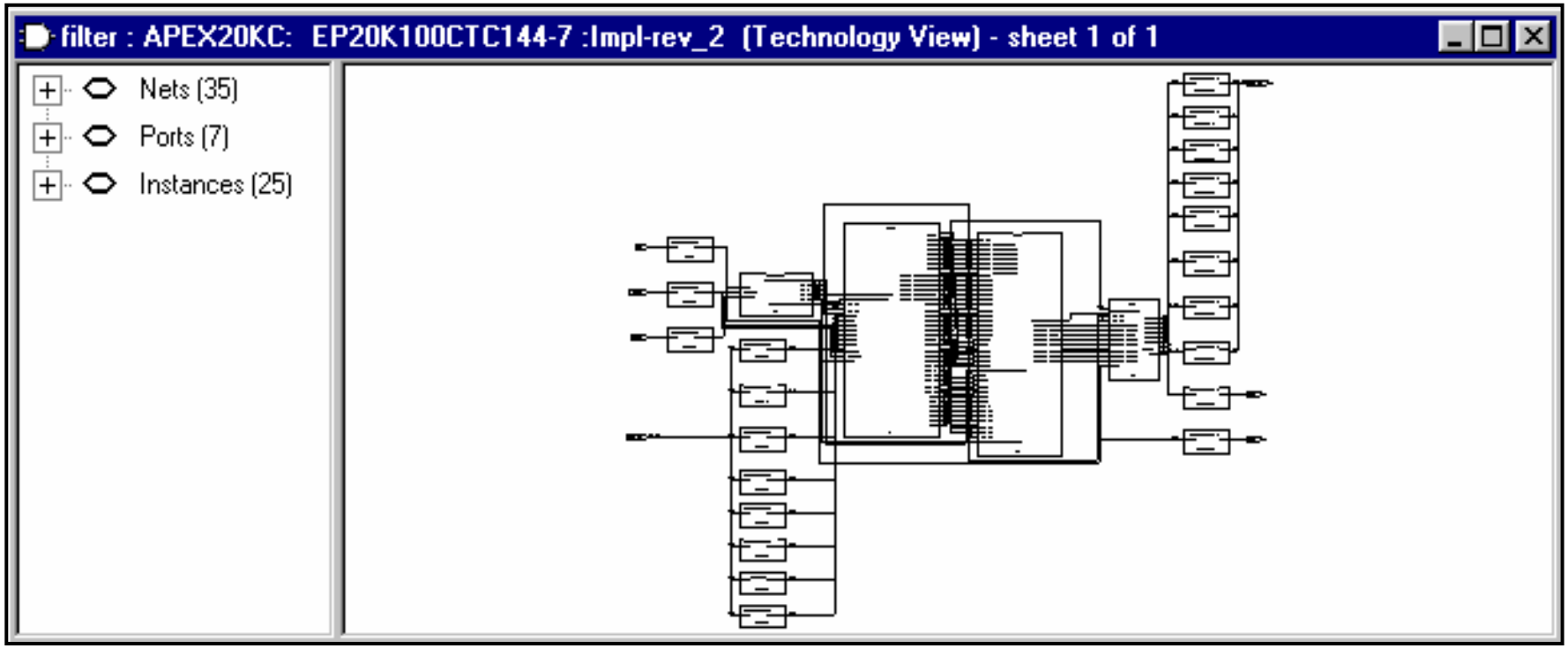




Copyright © 2004 Altera Corporation

Technology View

Opens Technology View



Copyright © 2004 Altera Corporation

Technology View Features

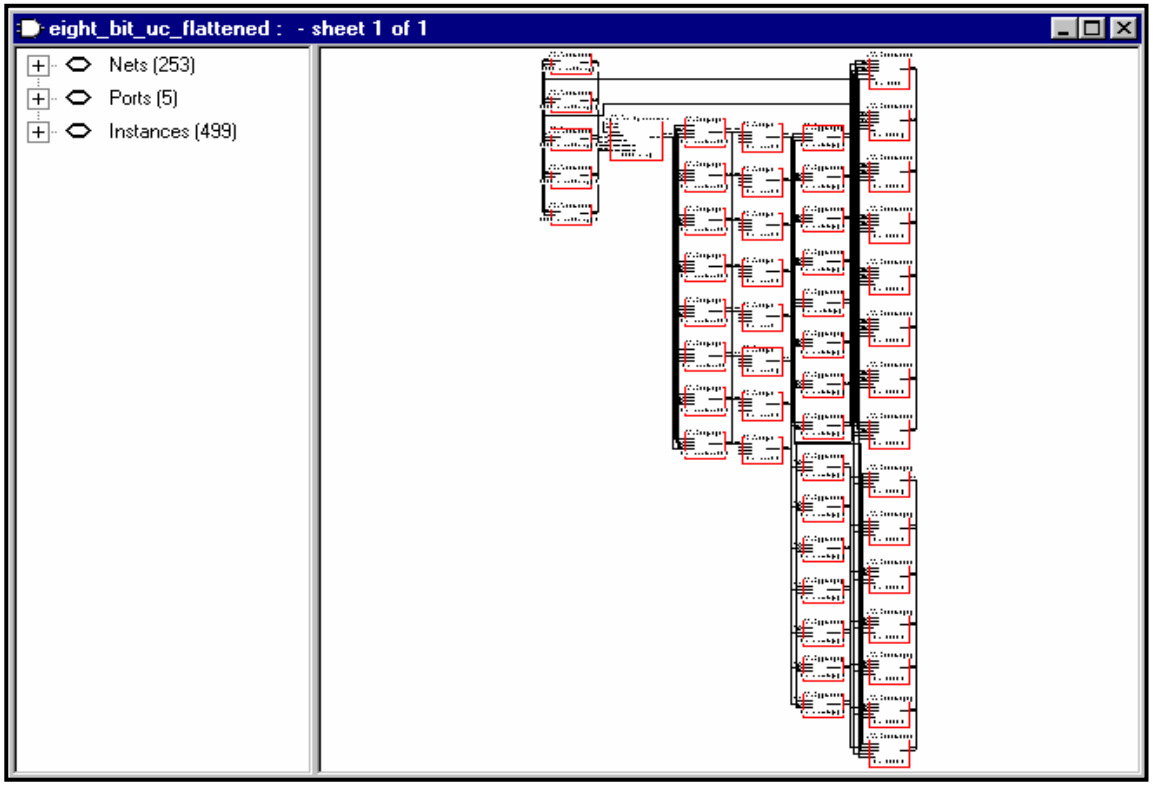
- Hierarchy Browser
- Push/Pop Control
- Flattened Hierarchy
- Flattened to Gates View
  - Displays Design Down to Boolean Logic
- Filter Schematic
- Critical Path
  - Highlights Instances & Nets in Critical Paths
  - Displays Timing
- Cross-Probing

Copyright © 2004 Altera Corporation

Technology View – Critical Path

View Critical Path

Filter Critical Path

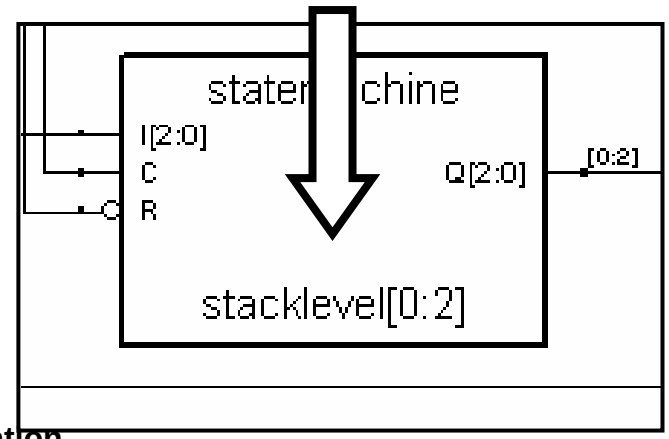


Copyright © 2004 Altera Corporation

FSM Viewer

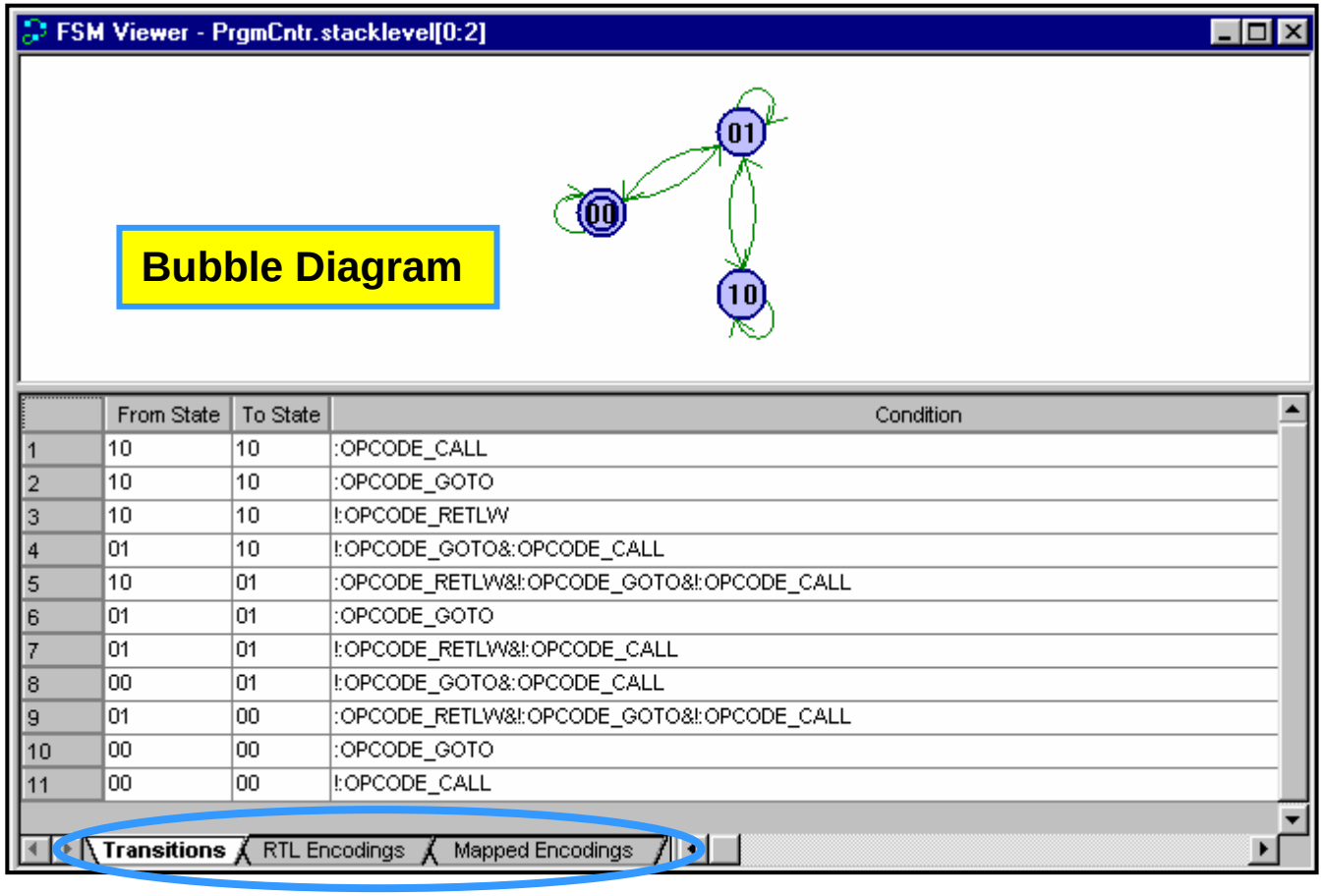
⑤ FSM Viewer

- Displays
  - ③Transition Bubble Diagram
  - ③Transition Table
  - ③FSM Encoding
- Open FSM from the RTL View
  - ③Push into FSM block
  - ③Right-Click on Extracted FSM and Select View FSM



Copyright © 2004 Altera Corporation

FSM Viewer Window



	From State	To State	Condition
1	10	10	:OPCODE_CALL
2	10	10	:OPCODE_GOTO
3	10	10	!:OPCODE_RETLW
4	01	10	!:OPCODE_GOTO&:OPCODE_CALL
5	10	01	:OPCODE_RETLW&!:OPCODE_GOTO&!:OPCODE_CALL
6	01	01	:OPCODE_GOTO
7	01	01	!:OPCODE_RETLW&!:OPCODE_CALL
8	00	01	!:OPCODE_GOTO&:OPCODE_CALL
9	01	00	:OPCODE_RETLW&!:OPCODE_GOTO&!:OPCODE_CALL
10	00	00	:OPCODE_GOTO
11	00	00	!:OPCODE_CALL

Mapping Information Tabs

Copyright © 2004 Altera Corporation

Mapping Information Tabs

- Transitions (Previous Slide)
  - Simple Equations/Conditions for each State Transition
- RTL Encoding
  - Shows State Machine as One-Hot Encoding
- Mapped Encoding
  - Shows State Machine as Implemented in Technology

State	Register
00	stacklevel[2]
01	stacklevel[1]
10	stacklevel[0]

RTL Encodings | Mapp

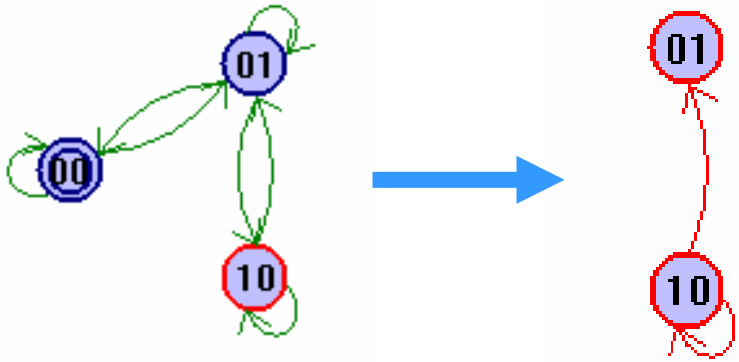
State	stacklevel[1]	stacklevel[0]
00	0	0
01	0	1
10	1	0

RTL Encodings | Mapped Encodings

Copyright © 2004 Altera Corporation

FSM Viewer Features

- Use Filter to Isolate States
  - Selected
  - By Input Transitions
  - By Output Transitions
- State Properties
  - Shows State Name, RTL Encoding, & Mapped Encoding
  - Right-Click on State in Bubble Diagram
- FSM Info File
  - Text Description of FSM
  - Right-Click on FSM in RTL View



State "01" Filtered By Output Transitions

Copyright © 2004 Altera Corporation

FSM Info File

```
State Machine: work.Prgm_Cntr(rtl)-stacklevel[0:2]
No selected encoding - Synplify will choose
Number of states: 3
Number of inputs: 3
Inputs:
   0: OPCODE_RETLW
   1: OPCODE_GOTO
   2: OPCODE_CALL
Clock: CLK
Reset: RESET to state S0
Transitions: (input, start state, destination state)
-01 S0 S1
-1- S0 S0
--0 S0 S0
-01 S1 S2
-1- S1 S1
0-0 S1 S1
100 S1 S0
--1 S2 S2
-1- S2 S2
0-- S2 S2
100 S2 S1
```

Copyright © 2004 Altera Corporation

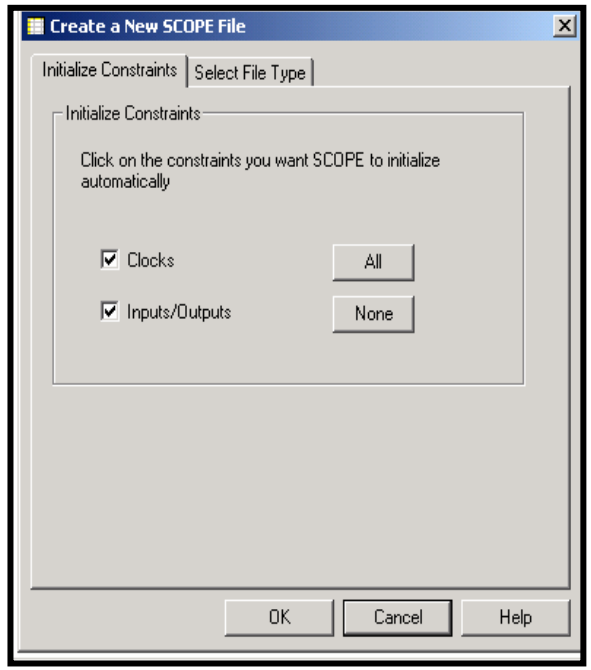
SCOPE

- Graphical, spread-sheet format to enter and manage timing constraints and attributes
- Has the capability of initializing clocks, inputs and outputs
- Has auto-fill capability in order to help in filling the constraints and attributes
- Filters out objects based on the constraint or attribute applied
- Filters attributes and constraints based on the object selected

Note: *Design should be compiled at least once before using SCOPE*

Copyright © 2004 Altera Corporation

SCOPE



New Constraints File Toolbar Button Opens the Intialize New Constraint File Dialog Box

Choosing Initialize Tells Synplify Pro to Initialize the Constraint File (SDC) with All Port & Clock Names

Copyright © 2004 Altera Corporation

SCOPE Editor

NOT IN PROJECT!! u_control_1.sdc (constraint File) *

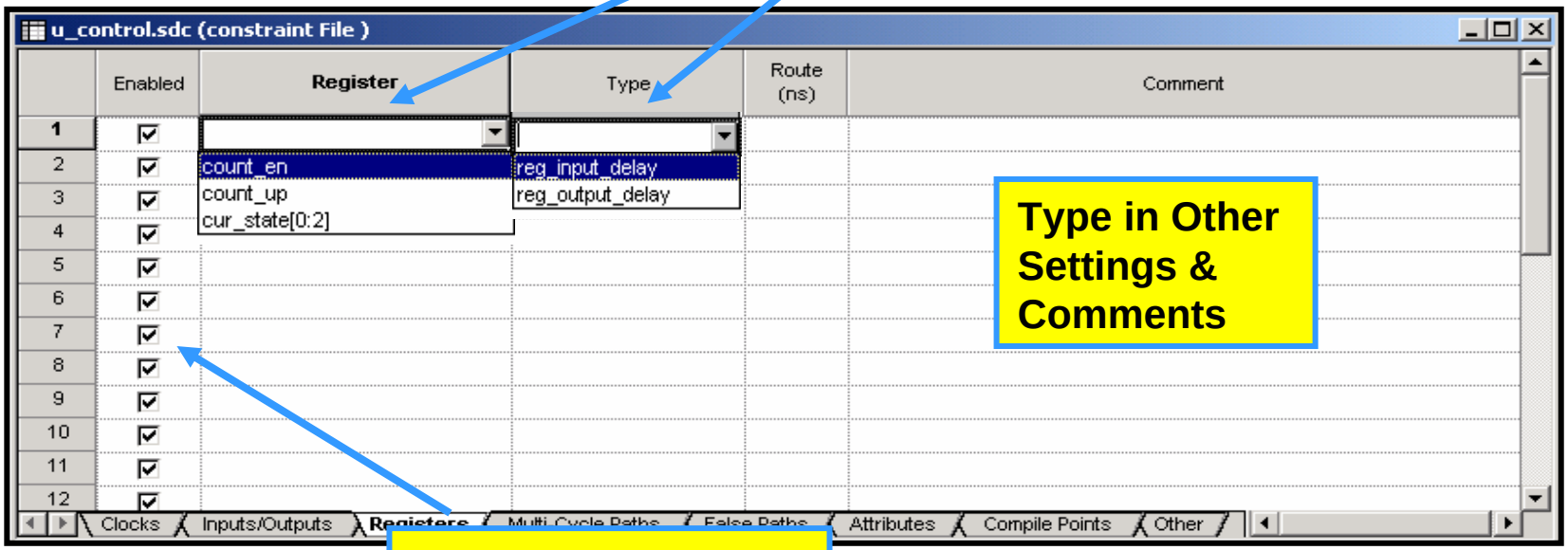
	Enabled	Clock	Frequency (MHz)	Period (ns)	Clock Group	Rise At (ns)	Fall At (ns)	Duty Cycle (%)	Route (ns)	Virtual Clock
1	☐	clk								☐
2	☑									☐
3	☑									☐
4	☑									☐
5	☑									☐
6	☑									☐
7	☑									☐
8	☑									☐
9	☑									☐
10	☑									☐
11	☑									☐
12	☑									☐

Clocks | Inputs/Outputs | Registers | Multi-Cycle Paths | False Paths | Attributes | Compile Points | Other

Copyright © 2004 Altera Corporation

Using SCOPE

Use Drop-Down Menus to Select Object Name and Allowed Values



Type in Other Settings & Comments

Each Constraint Can Be Individually Enabled or Disabled

Copyright © 2004 Altera Corporation

Using SCOPE (cont.)

- Synplify Pro tool allows dragging and dropping of objects from HDL Analyst Module to SCOPE editor
- SCOPE editor also uses specific abbreviation in order to distinguish same names for different objects
  - v: module or view
  - n: net
  - p : port
  - i: instance
  - b: bit slice

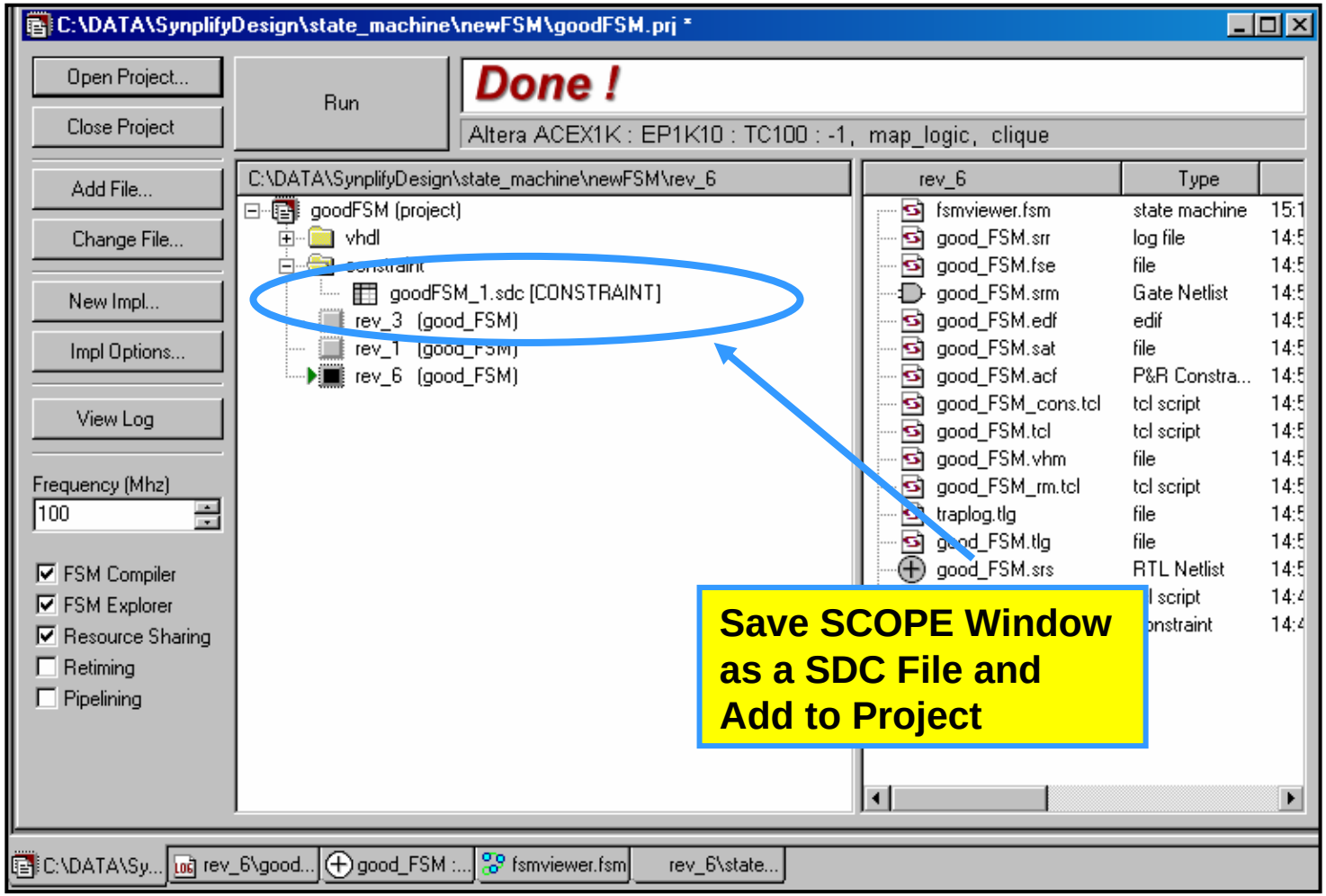
Copyright © 2004 Altera Corporation

SCOPE Tabs

- Clocks
- Inputs/Outputs
- Registers
- Multi-Cycle Paths
- False Paths
- Attributes
- Compile Points
- Other

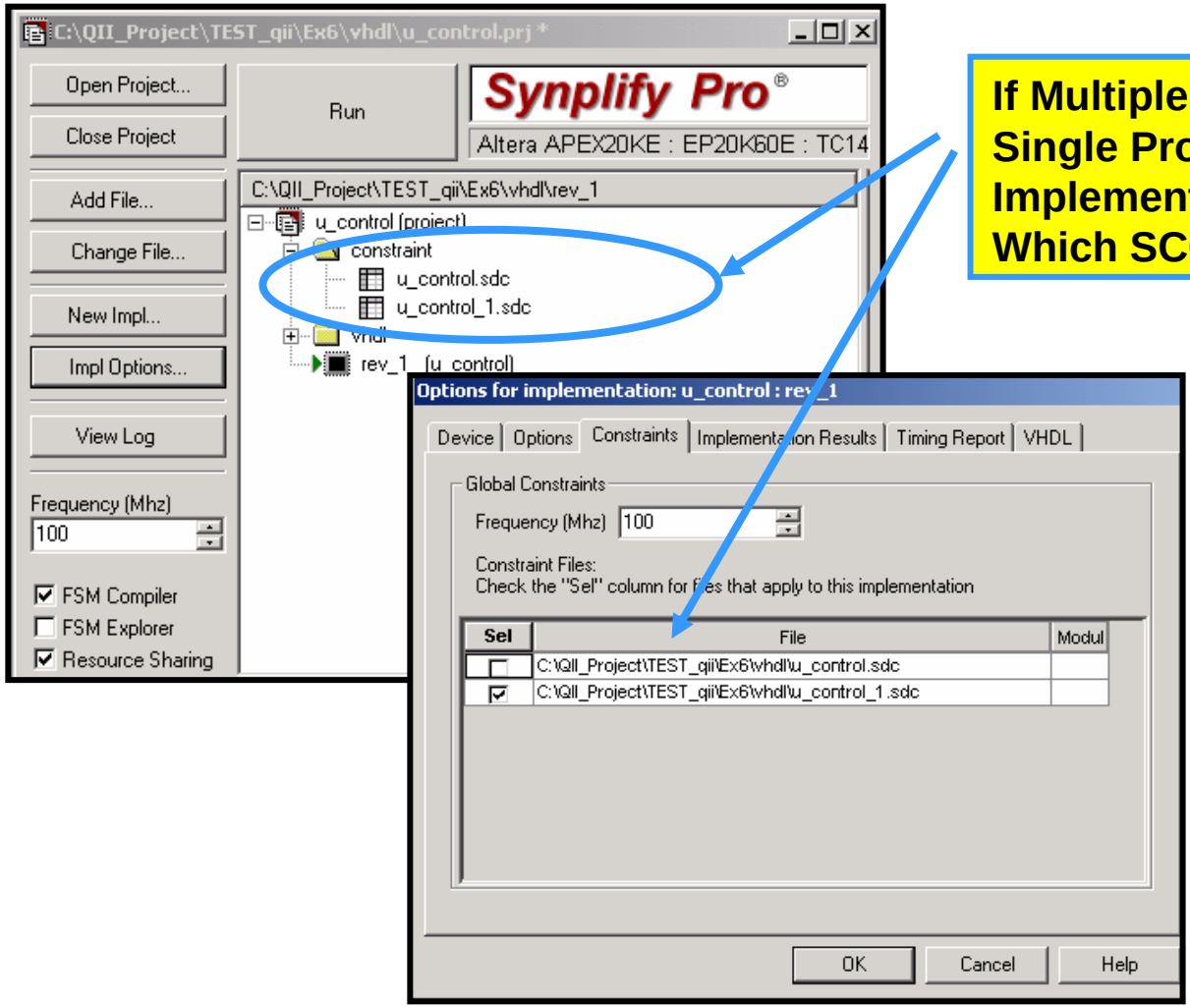
Copyright © 2004 Altera Corporation

Add SCOPE File to Project



Copyright © 2004 Altera Corporation

Add SCOPE File to Implementation



If Multiple SCOPE Files in Single Project, Each Implementation Selects Which SCOPE Files to Apply

Copyright © 2004 Altera Corporation

Timing Constraints

⑤Importance of Timing Constraints

⑤SCOPE Timing Assignments

Copyright © 2004 Altera Corporation

Importance of Timing Constraints

- The Synplify Pro synthesis engine is based on timing driven synthesis
  - Giving accurate timing constraints leads the tool to perform better
  - The Synplify Pro tool does extensive optimization, replication and buffering in order to achieve specified timing goals
  - The Synplify Pro tool does intelligent packing of registers in the I/Os depending on the timing specified
- Forward annotation to Quartus II
  - The timing information is forward annotated to Quartus II hence it is important to specify detailed timing constraints

Copyright © 2004 Altera Corporation

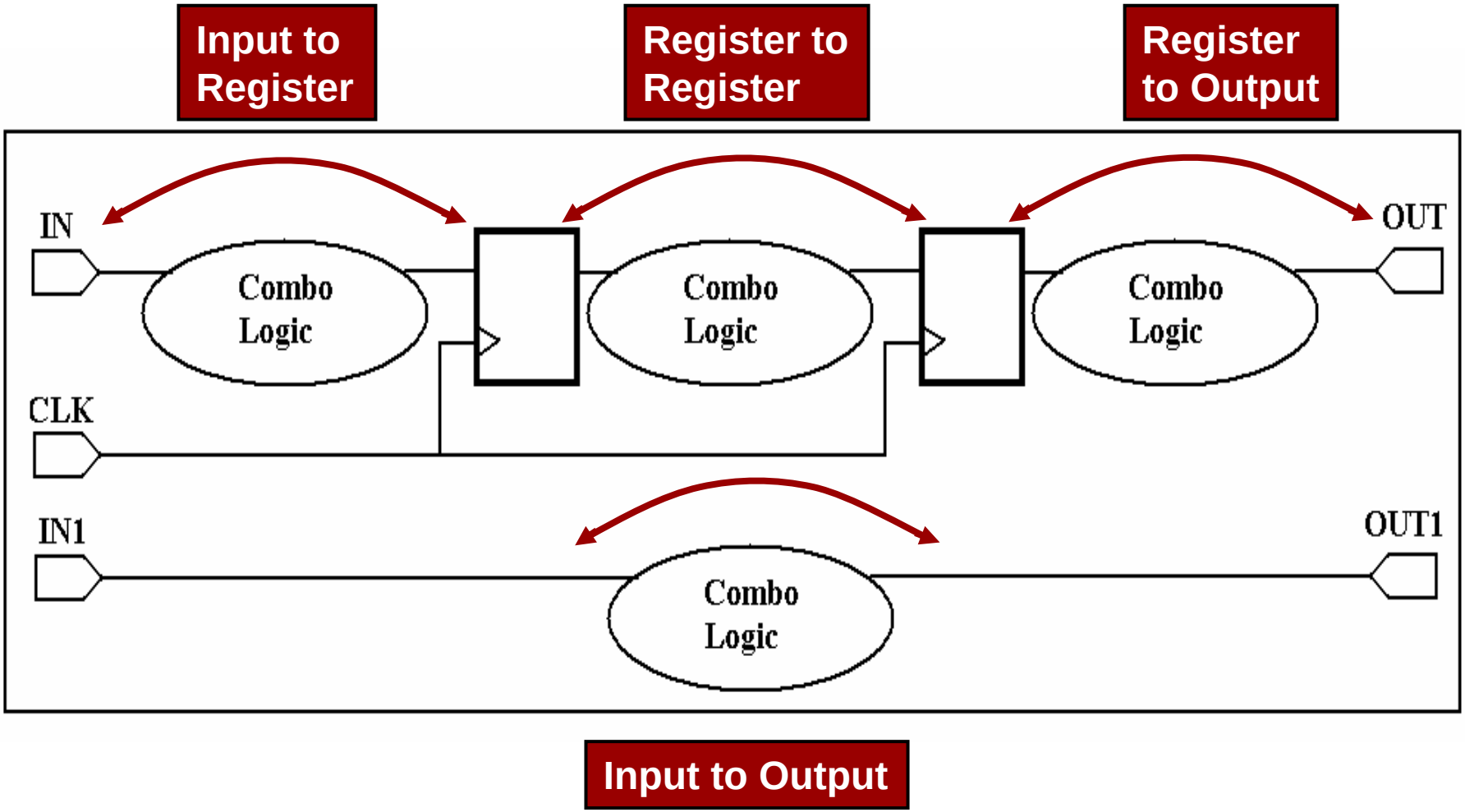
Importance of Timing Constraints (cont.)

⑤Use Realistic Timing Constraints

- Paths Ranked According to Applied Timing Constraints
  - ③Synplify Pro Works on the Most Critical
- Unnecessary Paths Optimized
- Area Sacrificed to Meet Performance

Copyright © 2004 Altera Corporation

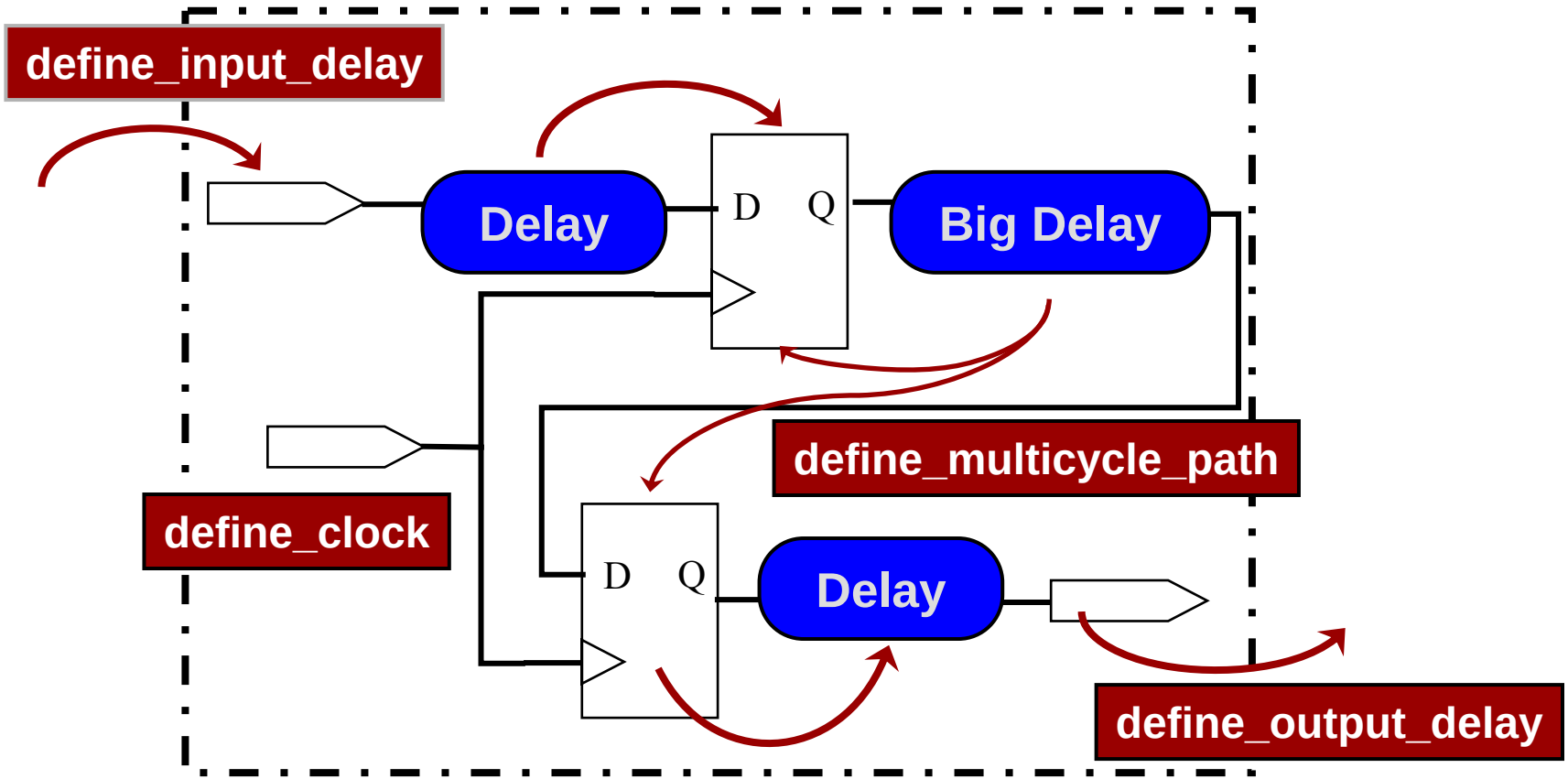
Types of Timing Paths



Copyright © 2004 Altera Corporation

Timing Constraints

⑤ Essential for achieving the performance goal

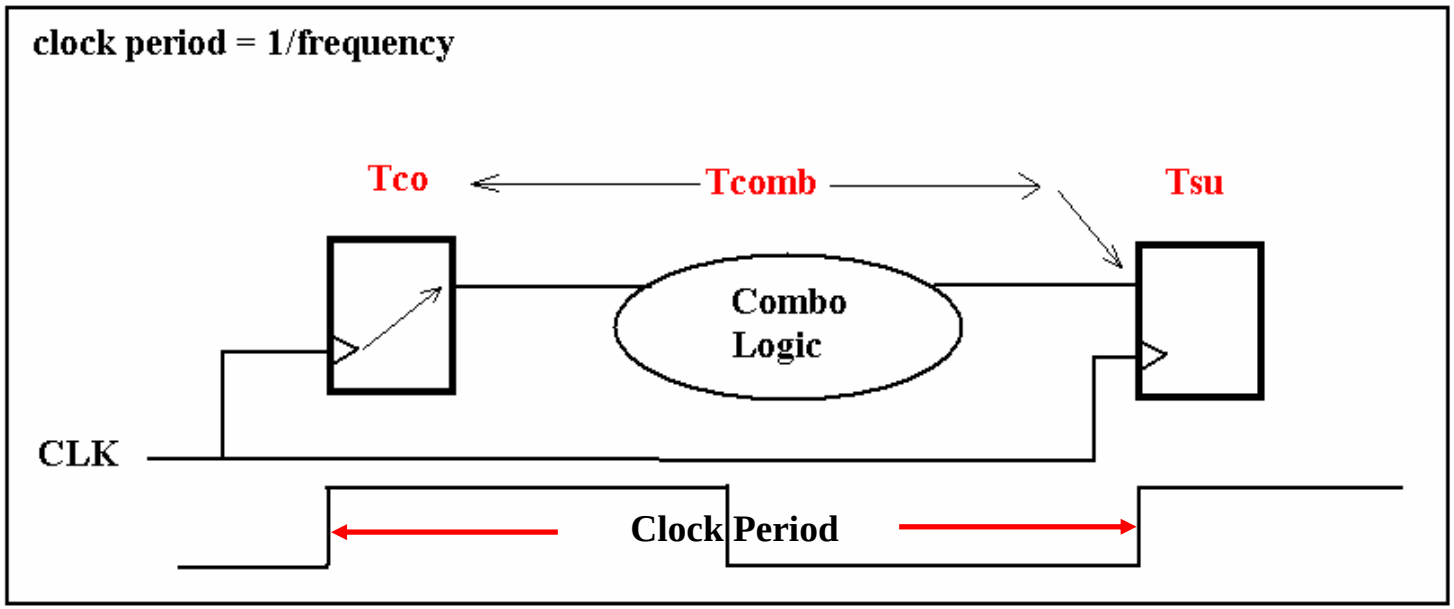


Copyright © 2004 Altera Corporation

Clock Frequency

- For Data Transfer to Occur Correctly,

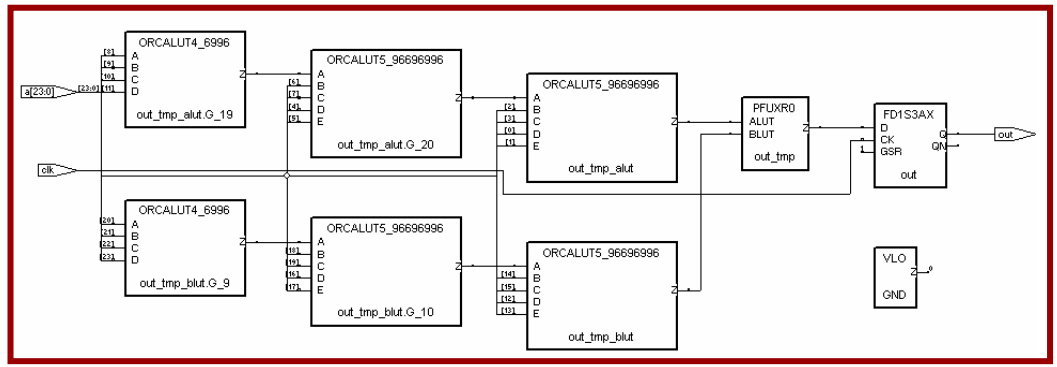
  Tco + Tcomb + Tsu < Clock Period
- Clock Frequency Is the Most Important Setting
  - Guides Combinatorial Logic Synthesis
  - Guides Reduction of Logic Levels between Registers



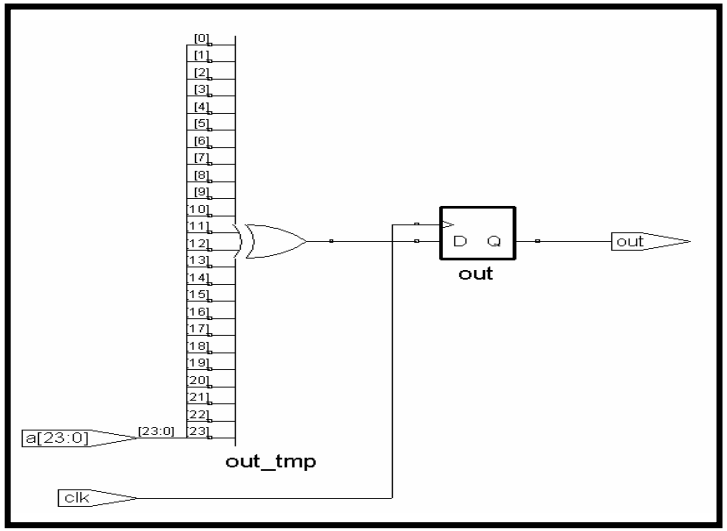
Copyright © 2004 Altera Corporation

Effects of Timing Constraints

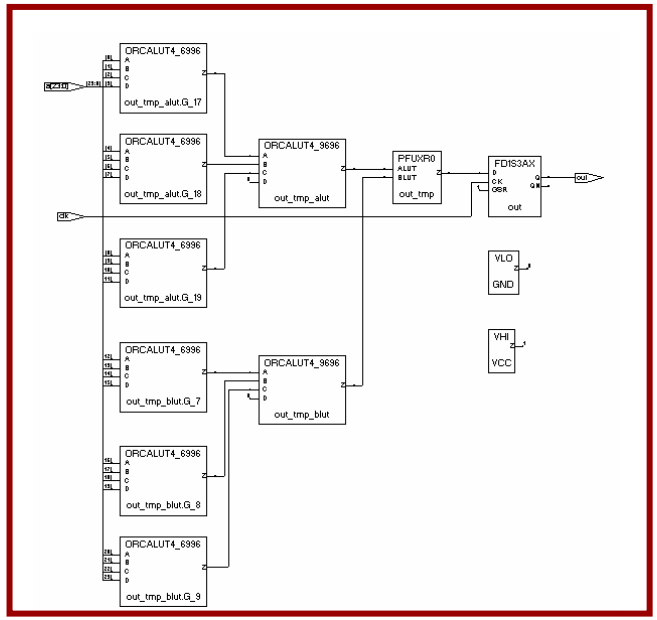
1 MHz Clock Setting



Original Logic



100 MHz Clock Setting



Copyright © 2004 Altera Corporation

Clocks in Synplify Pro

- Undefined Clocks Will Default to the Global Frequency
- Clocks Are Inferred by Synthesizing HDL & Tracing Back the Clock Pin of Flip-Flops
- All Clocks (Defined & Inferred) Appear in the “Performance Summary” Section of the Log File

Copyright © 2004 Altera Corporation

SCOPE Editor – Clock Tab

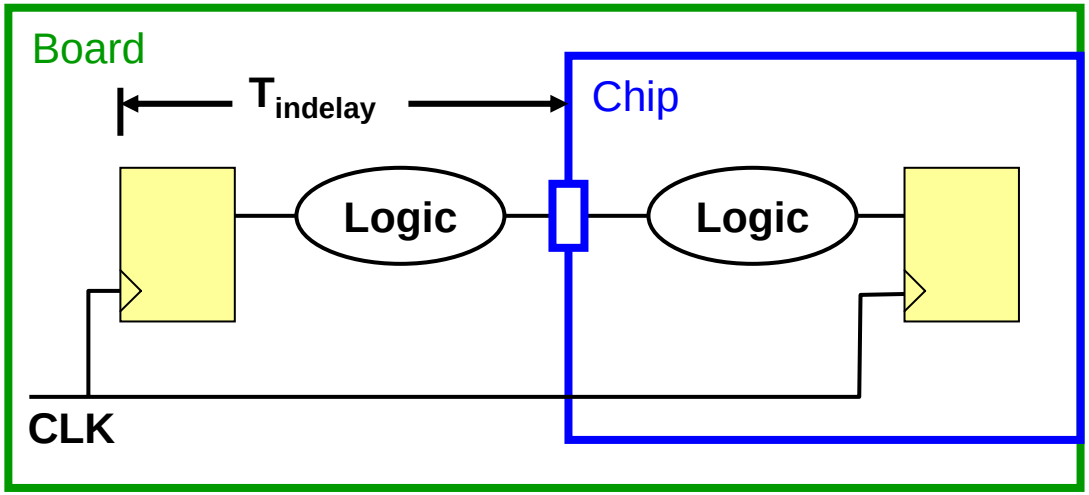
u_control.sdc (constraint File) *

	Enabled	Clock	Frequency (MHz)	Period (ns)	Clock Group	Rise At (ns)	Fall At (ns)	Duty Cycle (%)	Route (ns)	Virtual Clock
1	☐	clk								☐
2	☑									☐
3	☑									☐
4	☑									☐
5	☑									☐
6	☑									☐
7	☑									☐

- ⑤ Clock Ports in Design Automatically Filled In
- ⑤ Specify Frequency Value, Units, & Duty Cycle
- ⑤ Internally Generated Clocks
  - Registers & Instance Outputs
    - ③Apply Constraint to Register/Instance
  - Combinatorial Logic Output
    - ③Must Use *syn_keep* & *keepbuf* attributes

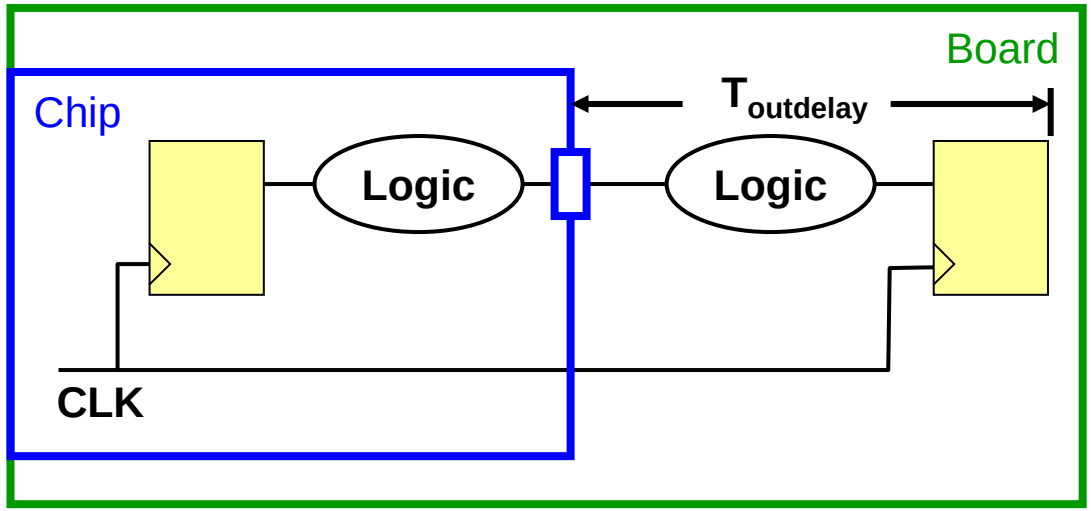
Copyright © 2004 Altera Corporation

Input & Output Delay



Input Delay Refers to the Arrival Time from an External Register to the Chip Interface (T_{CO} + T_{Logic})

Output Delay Refers to the Time from the Chip Interface to an External Register Plus that Register's Setup Time (T_{Logic} + T_{SU})



Note: Default Delays Equal Zero

Copyright © 2004 Altera Corporation

SCOPE Editor – Inputs/Outputs Tab

- I/O Ports Automatically Filled In
- Global or Individual Port Settings
- *input_delay*

  Constrains Timing for Input & Bi-Directional Ports
- output_delay

  Constrains Timing for Output & Bi-Directional Ports

u_control.sdc (constraint File) *

	Enabled	Port	Type	Clock Edge	Value (ns)	Route (ns)	Comment
1	☐	<input default>	input_delay		5.00		
2	☐	<output default>	output_delay		4.00		
3	☑	inst[4:0]	input_delay				
4	☑	opA[7:0]	input_delay				
5	☑	opB[7:0]	input_delay				
6	☑	result_high_out[15:0]	output_delay				
7	☑	result_out[7:0]	input_delay				

Copyright © 2004 Altera Corporation

SCOPE Editor – Multi-Cycle Paths

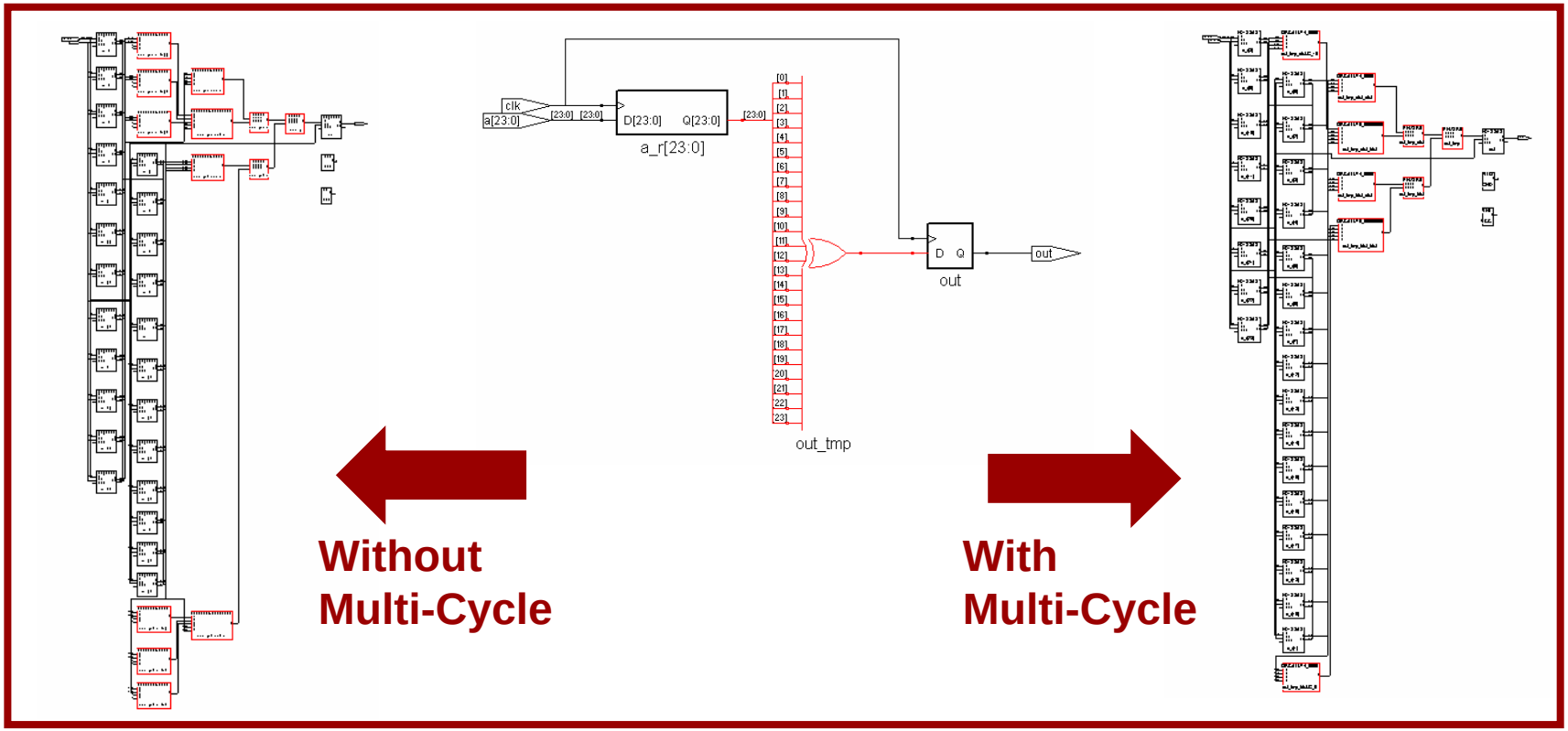
- Paths Requiring Multiple Clock Cycles for Data Transfer
- Types
  - *from*
    - Constrains All Paths Coming from the Selected Register
  - *to*
    - Constrains All Paths Leading to the Selected Register
  - *through*
    - Constrains All Paths Passing through the Selected Net

u_control.sdc (constraint File) *

	Enabled	From	To	Through	Cycles	Comment
1	☑		alu_inst.result_high_out[15:0]		2	
2	☑					
3	☑					
4	☑					
5	☑					
6	☑					

Copyright © 2004 Altera Corporation

Effect of Multi-Cycle Path Constraint



⑤ Without Multi-Cycle Constraint

- More Logic Is Used
- Longer Mapping Times
- Timing Report Is Inaccurate

Copyright © 2004 Altera Corporation

SCOPE Editor – False Paths

- Specifies Paths with Low or No Priority During Optimization
- Path Not Reported During Timing Reporting
- Similar to Multi-Cycle Path Tab
  - No Value Necessary

Copyright © 2004 Altera Corporation

Purely Combinatorial Paths

- Pure Combinatorial Paths Represent Unregistered Paths between Input & Output Ports
- The Global Frequency Specified Is Used to Constrain These Paths
- Make Sure the Global Frequency Specified is Set for Purely Combinatorial Paths if Performance Is an Issue

Copyright © 2004 Altera Corporation

Please go to Exercise 2

Copyright © 2004 Altera Corporation

Exercise 2 Review

- Using HDL Analyst
  - RTL View
  - Technology View
- Using SCOPE
- Making Timing Settings

Copyright © 2004 Altera Corporation

Design & Synthesis Summary

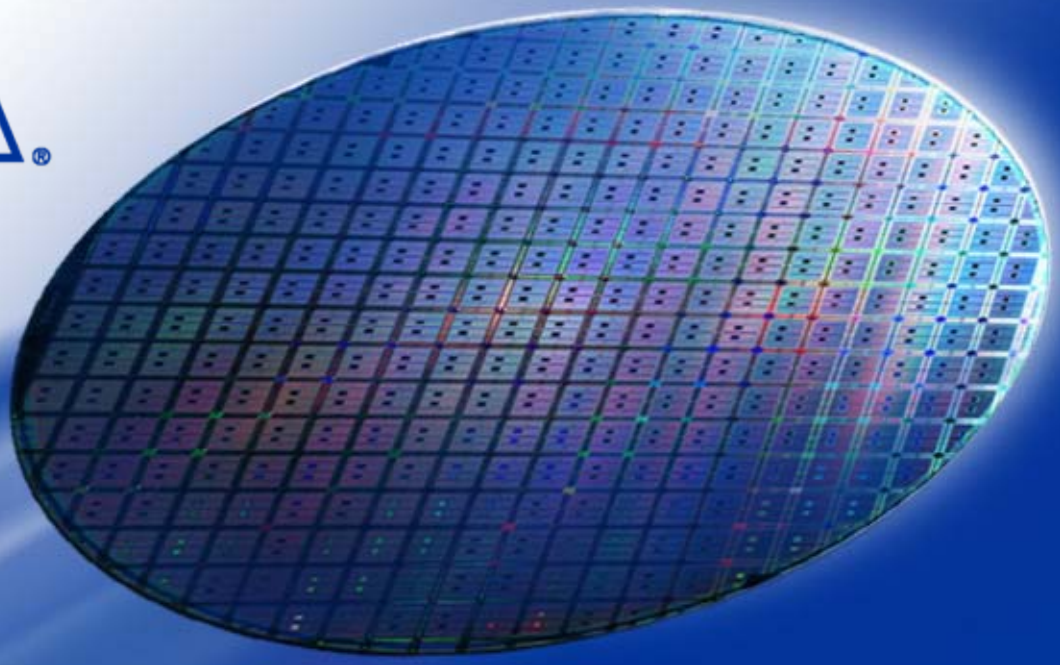
- 5 Easy Steps to Synthesis with Synplify Pro
  - Make Project
  - Add Files
  - Set Options (Device & Clock Frequency)
  - Run
  - View Results
- Add Timing Constraints to Improve Results

Copyright © 2004 Altera Corporation

Synthesis Complete!

Next…Place & Route

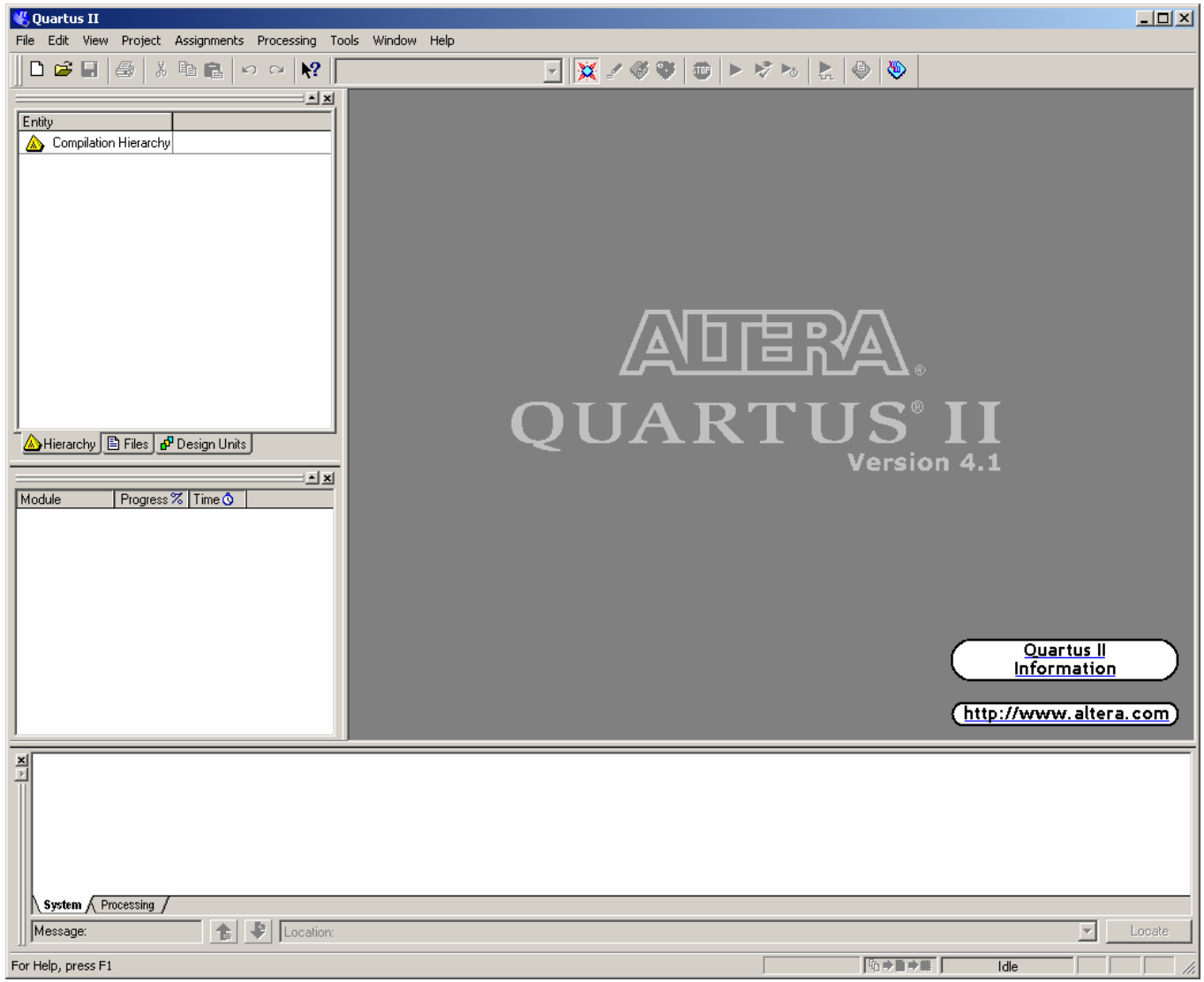
Copyright © 2004 Altera Corporation



Quartus II Design Environment

Copyright © 2004 Altera Corporation

Quartus II Operating Environment



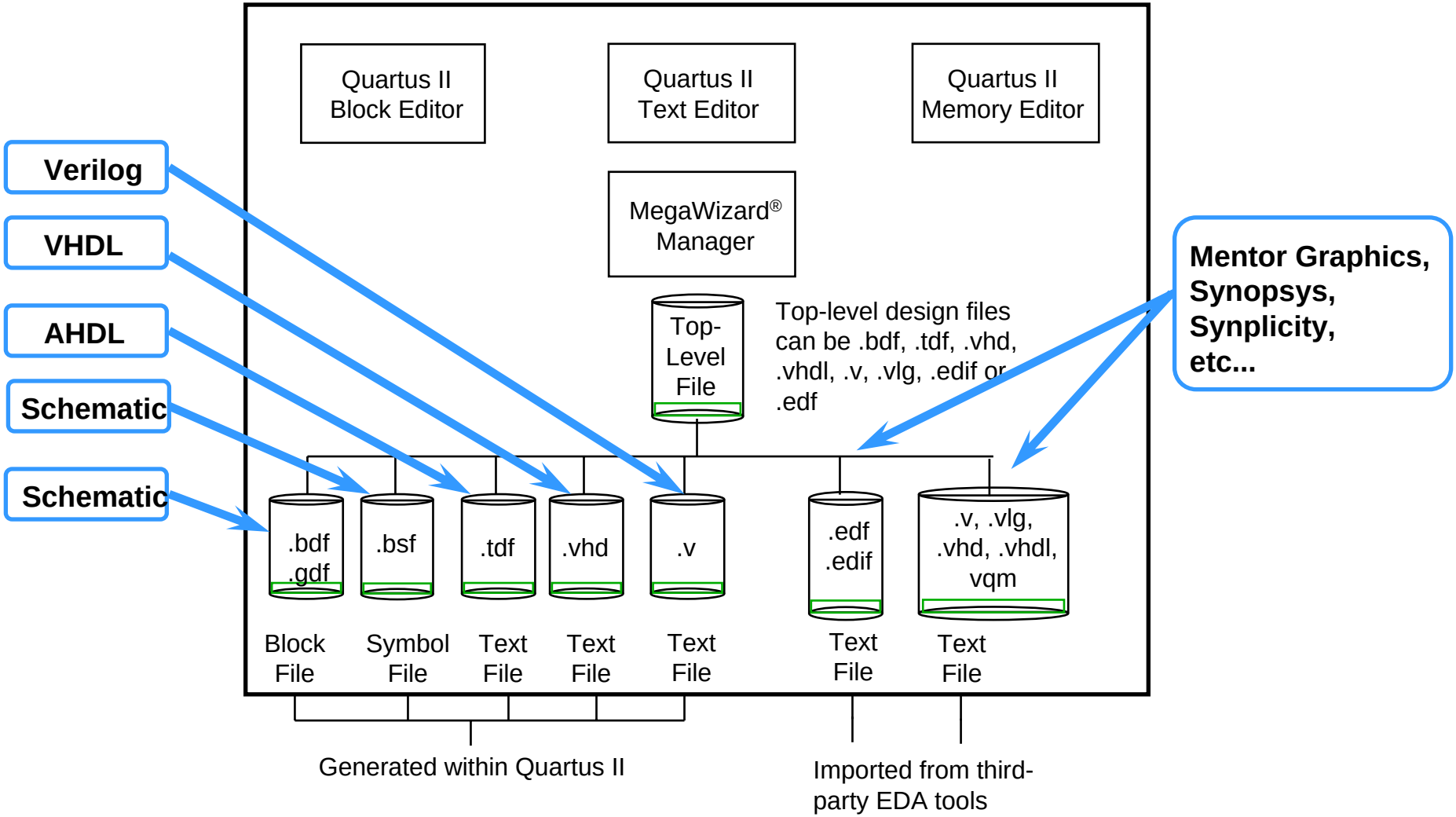
Copyright © 2004 Altera Corporation

Design Entry Methods

- ⑤Quartus II
  - Text Editor
    - ③AHDL, VHDL, Verilog
  - Memory Editor
    - ③HEX, MIF
  - Schematic Design Entry
- ⑤3rd-Party EDA Tools
  - EDIF
  - HDL
  - VQM
- ⑤Mixing & Matching Design Files Allowed

Copyright © 2004 Altera Corporation

Design Entry Files



Copyright © 2004 Altera Corporation

Text Design Entry

- ⑤ Available Features
  - Line Numbering in the HDL Text Files
  - Preview of HDL Templates
  - Syntax Coloring
  - When Editing a Text File, an Asterisk (*) Appears Next to the Filename
    - ③ Asterisk Disappears after Saving the File
- ⑤ Enter Text Description
  - AHDL (.TDF)
  - VHDL (.VHD)
  - Verilog (.V)

Copyright © 2004 Altera Corporation

Verilog & VHDL

- **VHDL**- VHSIC Hardware Description Language
  - 1987 & 1993 IEEE 1074 Standards Supported
- **Verilog** - 1995 & 2001 IEEE 1364 Standard HDL

- Create in Quartus II or any Standard Text Editor
- Use Quartus II Integrated Synthesis to Synthesize
- View Supported Commands in On-Line Help

Learn more about HDL in Altera HDL Customer Training Classes

Copyright © 2004 Altera Corporation

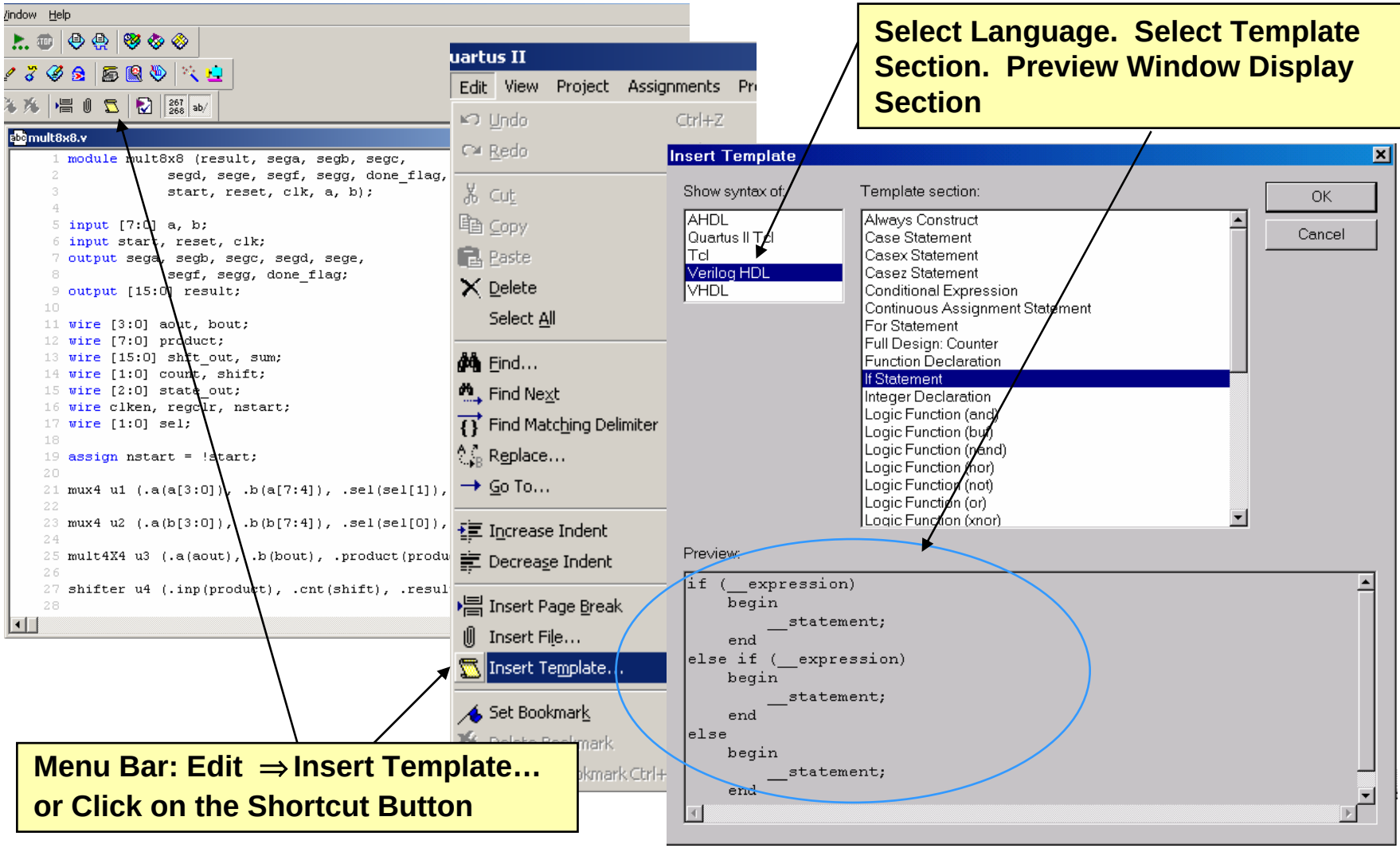
AHDL

- Altera Hardware Description Language
  - High-Level Hardware Behavior Description Language
  - Used in Altera Megafunctions
  - Uses Boolean Equations, Arithmetic Operators, Truth Tables, Conditional Statements, etc.

- Create in Quartus II or any Standard Text Editor

Copyright © 2004 Altera Corporation

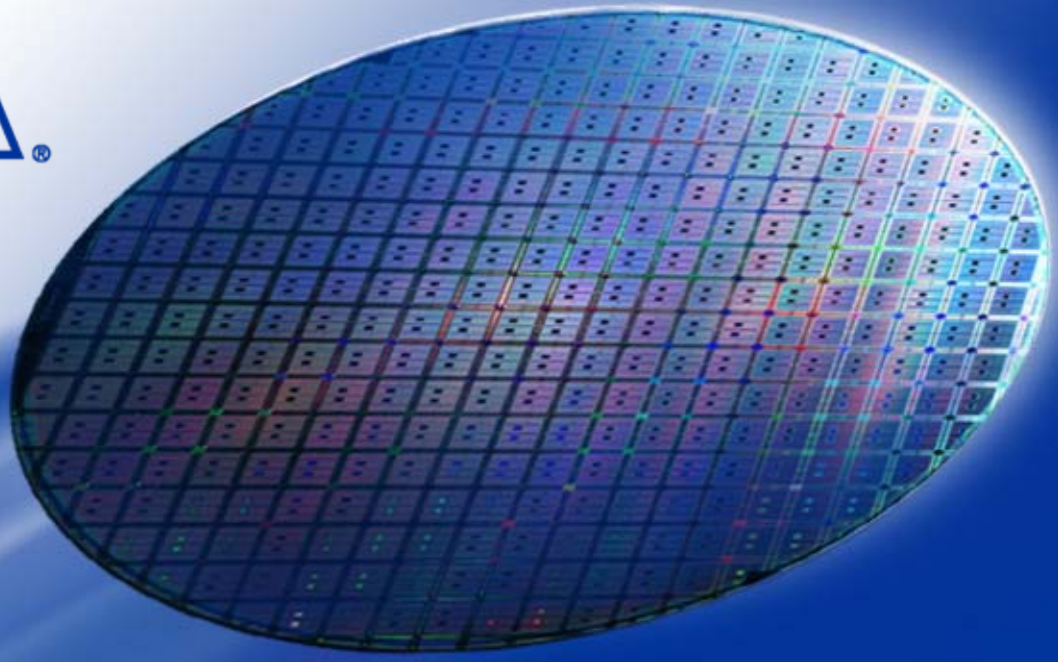
HDL Templates



Select Language. Select Template Section. Preview Window Display Section

Menu Bar: Edit ⇒ Insert Template… or Click on the Shortcut Button

Copyright © 2004 Altera Corporation



Quartus II Projects

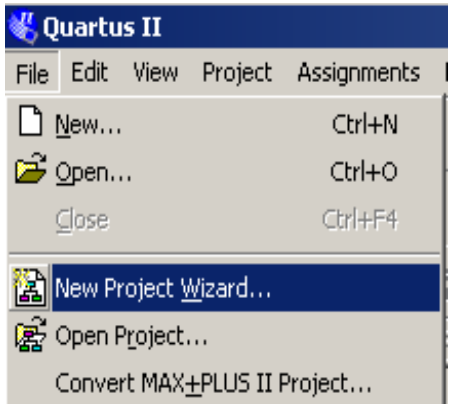
Copyright © 2004 Altera Corporation

Quartus II Projects

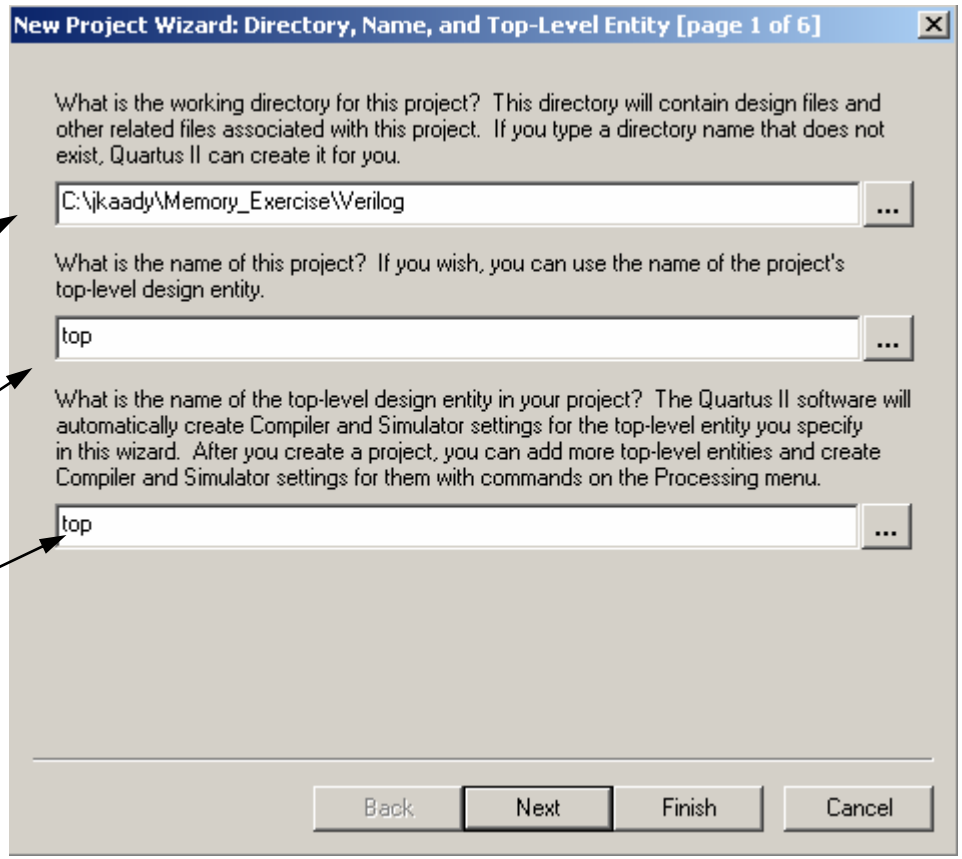
- Description
  - Collection of Related Design Files & Libraries
  - Must Have at Least One Designated Top-Level Entity
  - Target a Single Device or Can Be Partitioned into Multiple Devices
  - Store Settings in Quartus Settings File (.QSF)
- Create New Projects with New Project Wizard
  - Can Be Created Using Tcl Scripts

Copyright © 2004 Altera Corporation

New Project Wizard



Invoke New Project Wizard



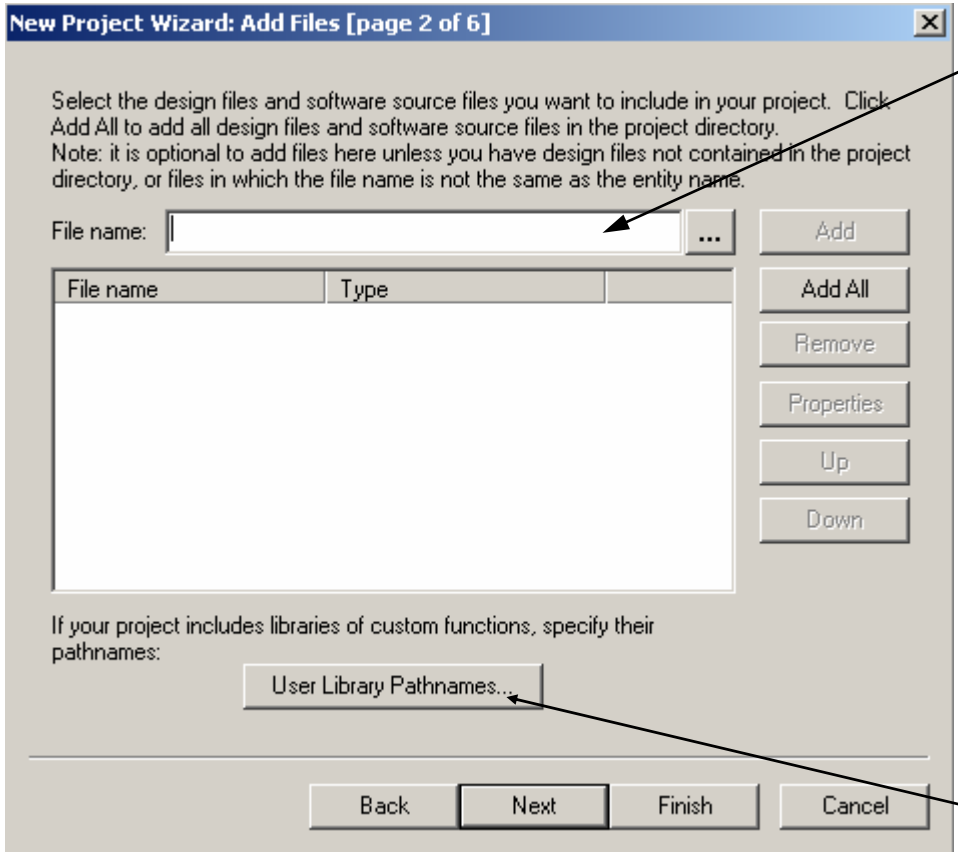
Select Working Directory

Name of Project Can Be Any Name; Recommend Using Top-Level File Name

Top-level Entity Does Not Need to Be the Same Name as Top-Level File Name

Copyright © 2004 Altera Corporation

Add Files



Add Design Files

- Graphic (.BDF, .GDF)
- AHDL
- VHDL
- Verilog
- EDIF

Notes:

- *Files in project directory do not need to be added*
- *Add top level file if filename & entity name are not the same*

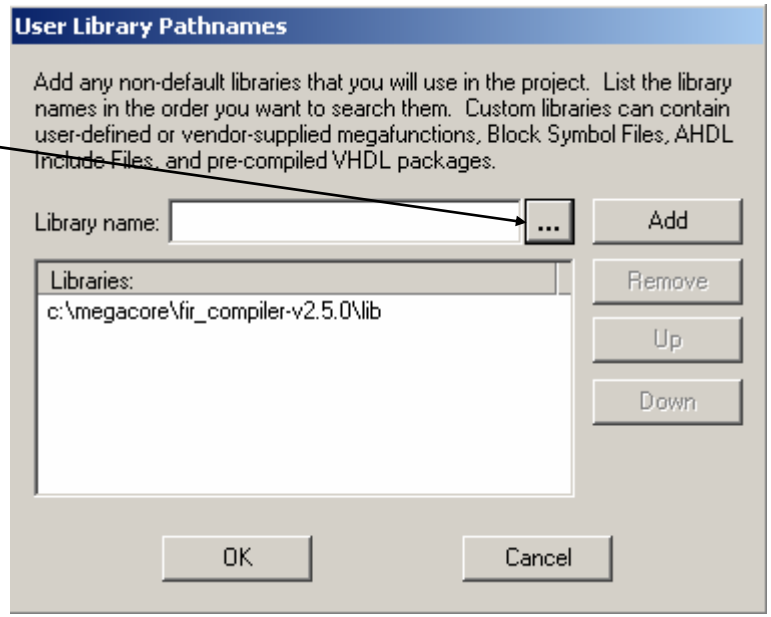
Add User Library Pathnames & Files

Copyright © 2004 Altera Corporation

User Libraries

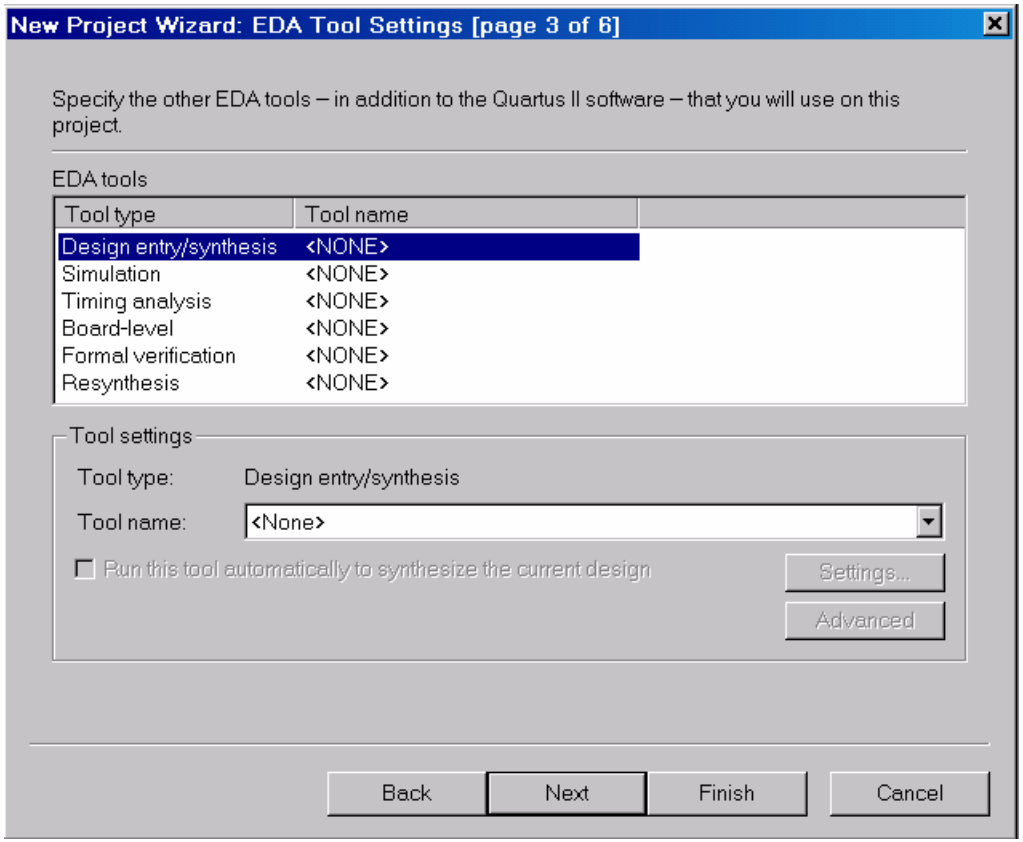
Add User Library Paths

- **User Libraries**
- **MegaCore®/AMPP Sm Libraries**
- **Pre-Compiled VHDL Packages**



Copyright © 2004 Altera Corporation

EDA Tool Settings

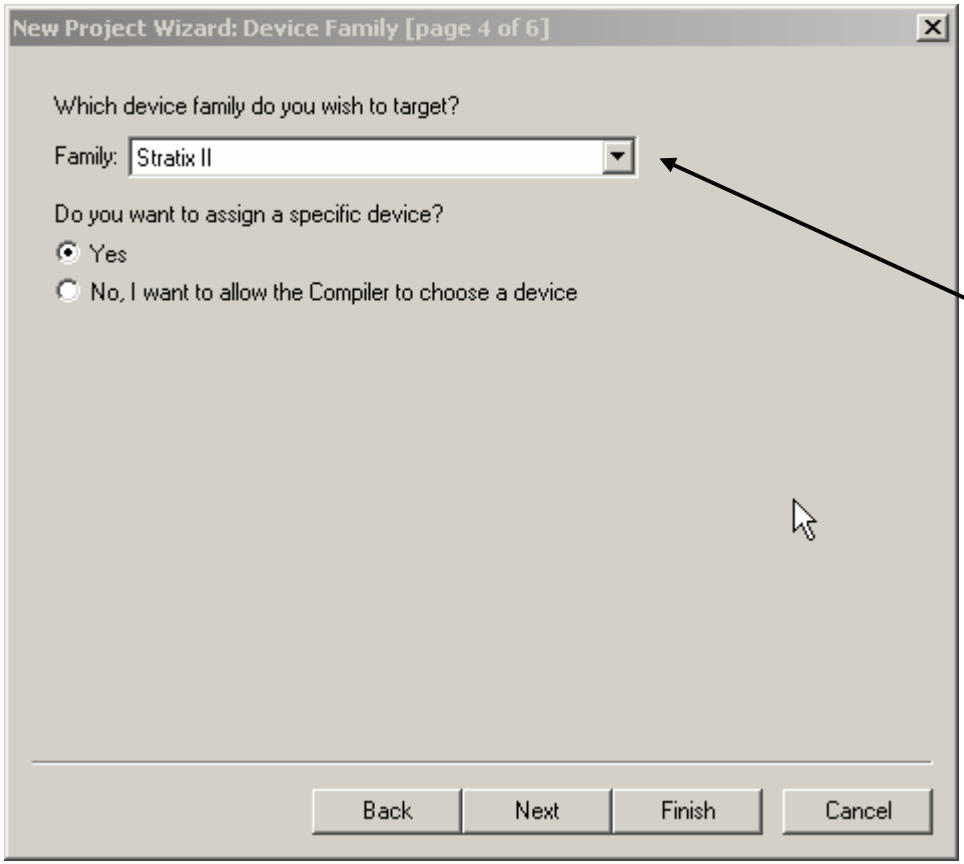


Choose EDA Tools

Add or Change Settings Later

Copyright © 2004 Altera Corporation

Device Selection

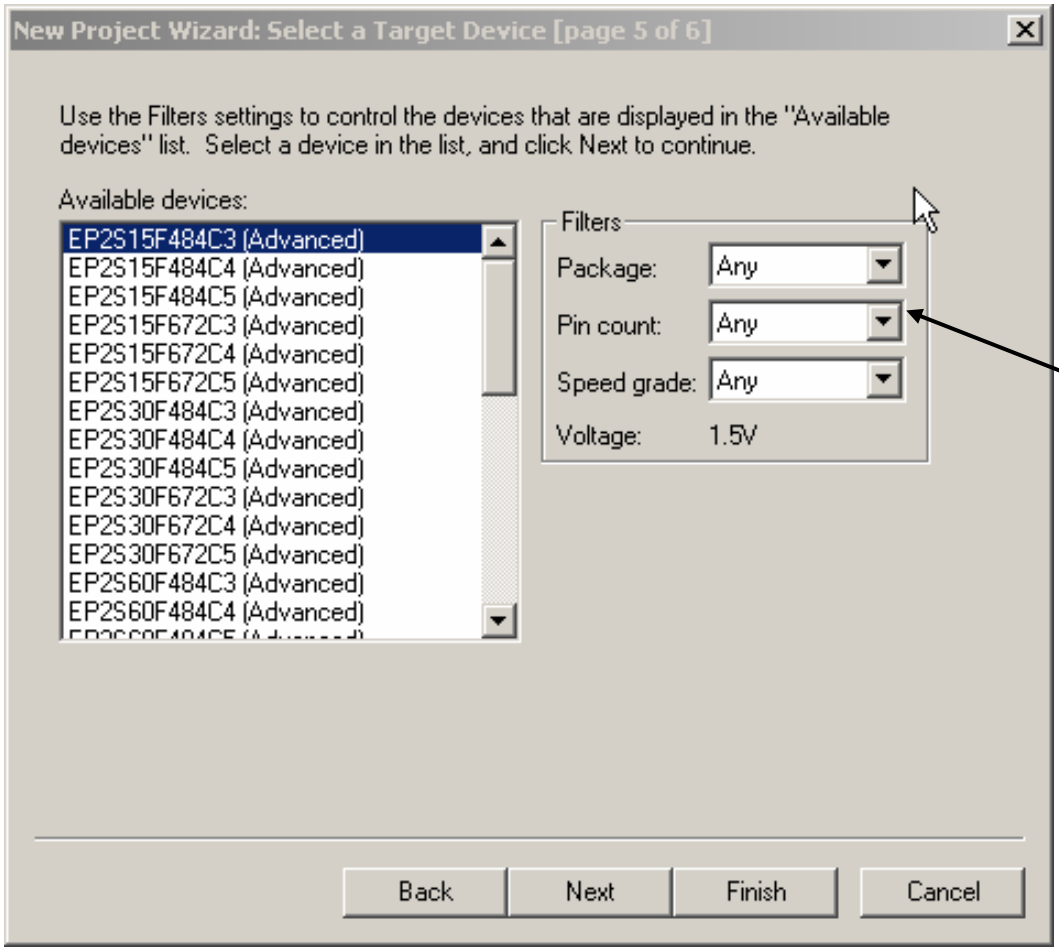


Choose the Device Family

Auto Device or Choose a Specific Device

Copyright © 2004 Altera Corporation

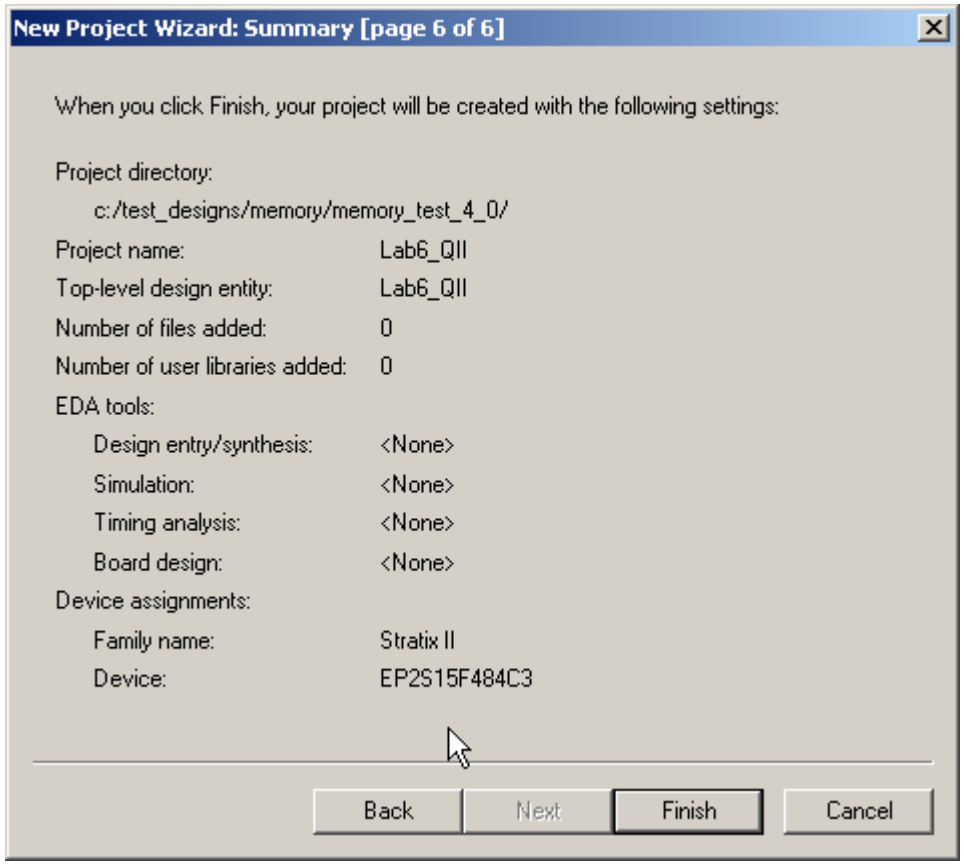
Device Selection



Use Filters for Auto Selection or Picking a Specific Device

Copyright © 2004 Altera Corporation

Done!

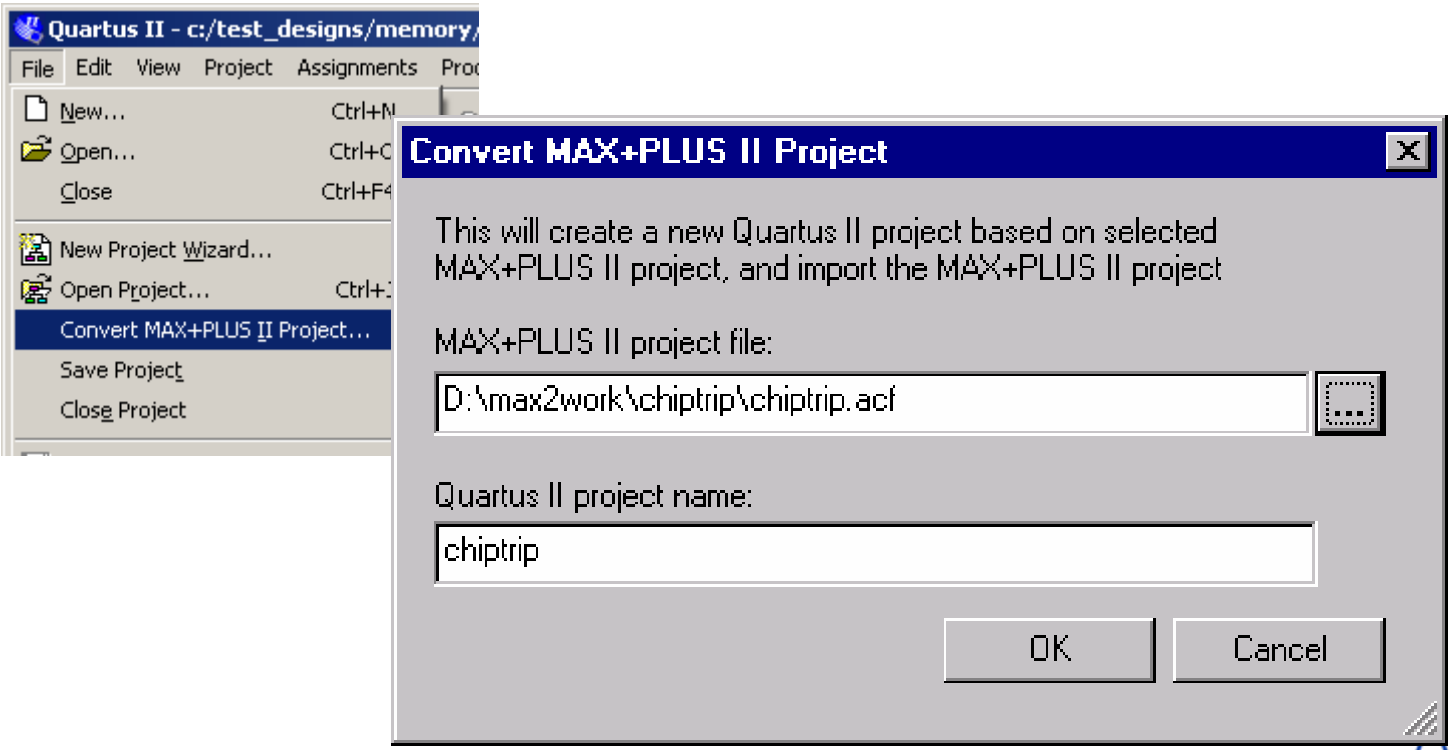


Review Results & Click on Finish

Copyright © 2004 Altera Corporation

MAX+PLUS II to Quartus II

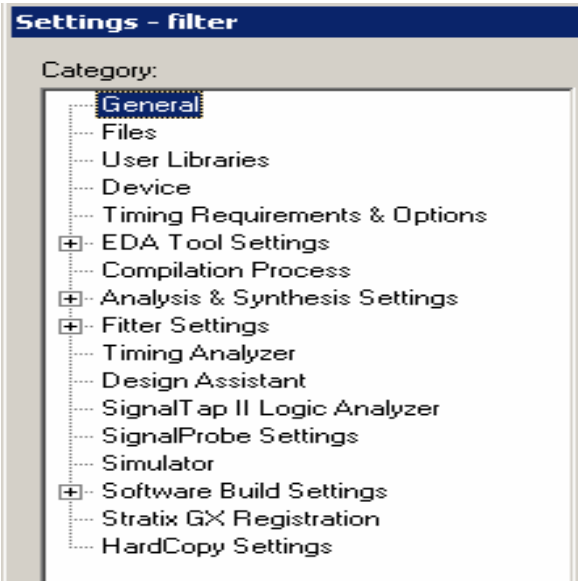
- Convert MAX+PLUS II Projects into Quartus II Projects
- Assignments Automatically Translated

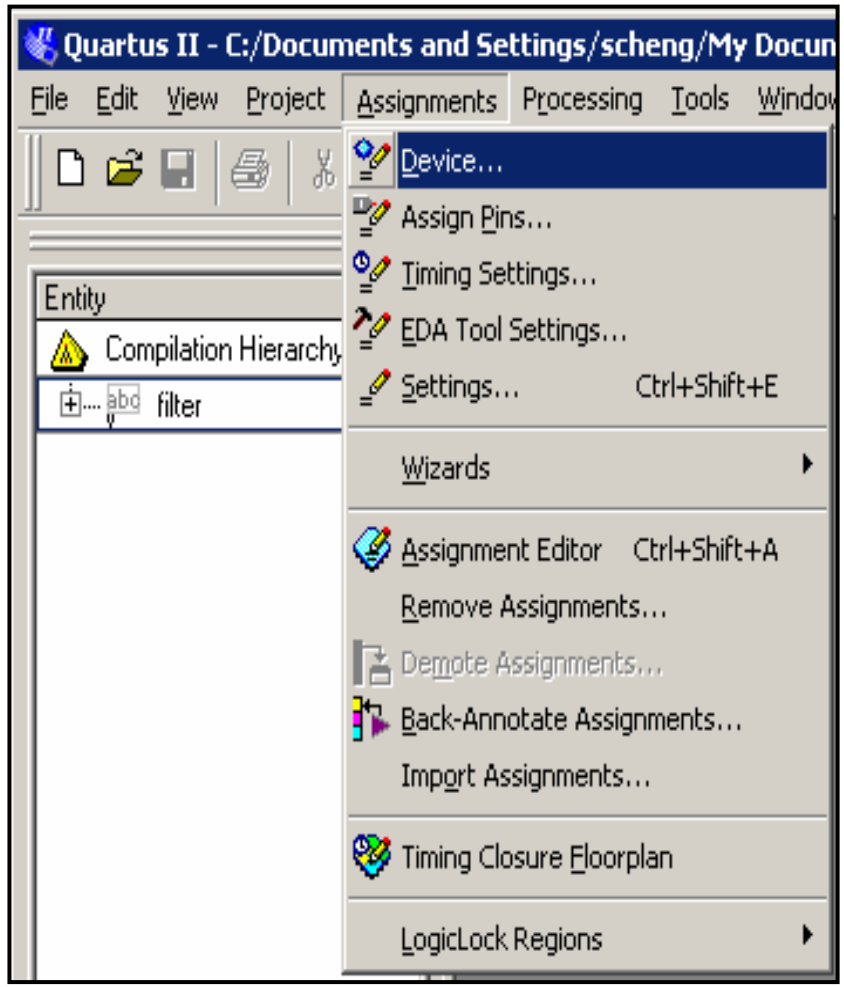


Copyright © 2004 Altera Corporation

Assignments Menu

- From Assignments Menu Access All Assignment & Settings Options
- All Settings Can Be Made from Assignments -> Settings

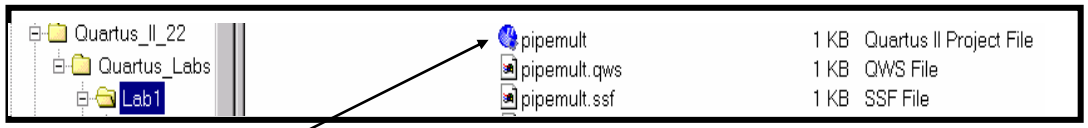




Copyright © 2004 Altera Corporation

Editing Project Settings

⑤ Existing project must first be opened to edit the settings



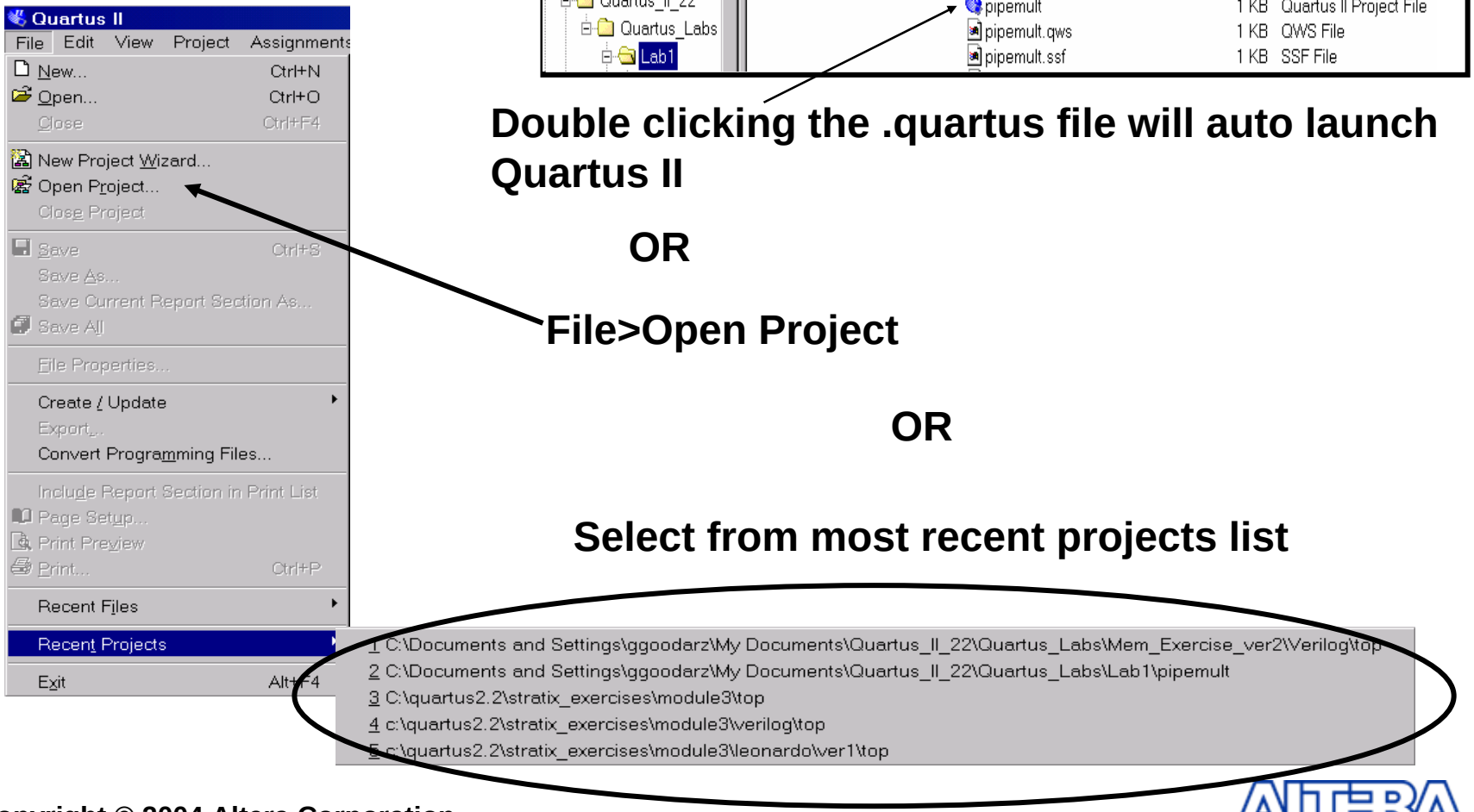
Double clicking the .quartus file will auto launch Quartus II

OR

File>Open Project

OR

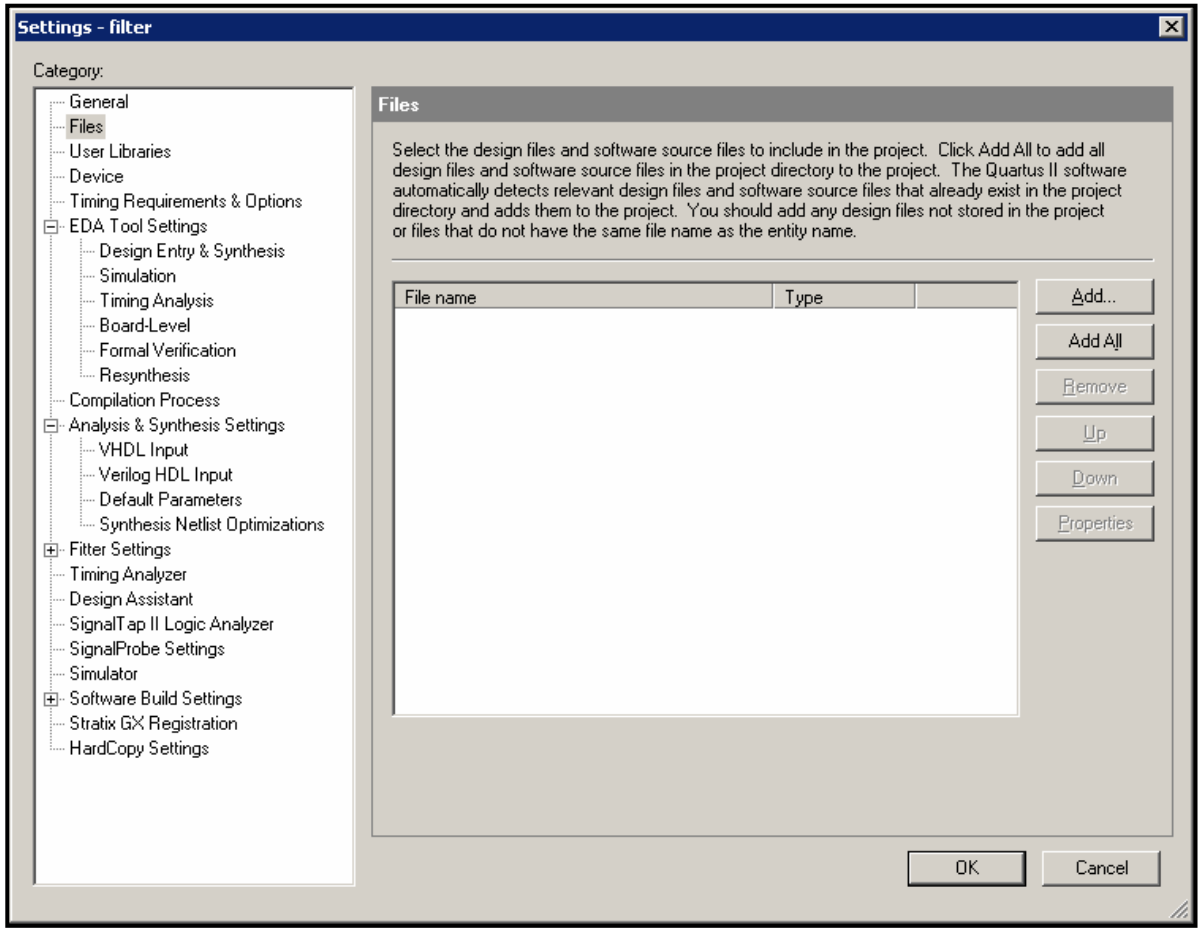
Select from most recent projects list



Copyright © 2004 Altera Corporation

Editing Project Settings

⑤ Assignments > Settings

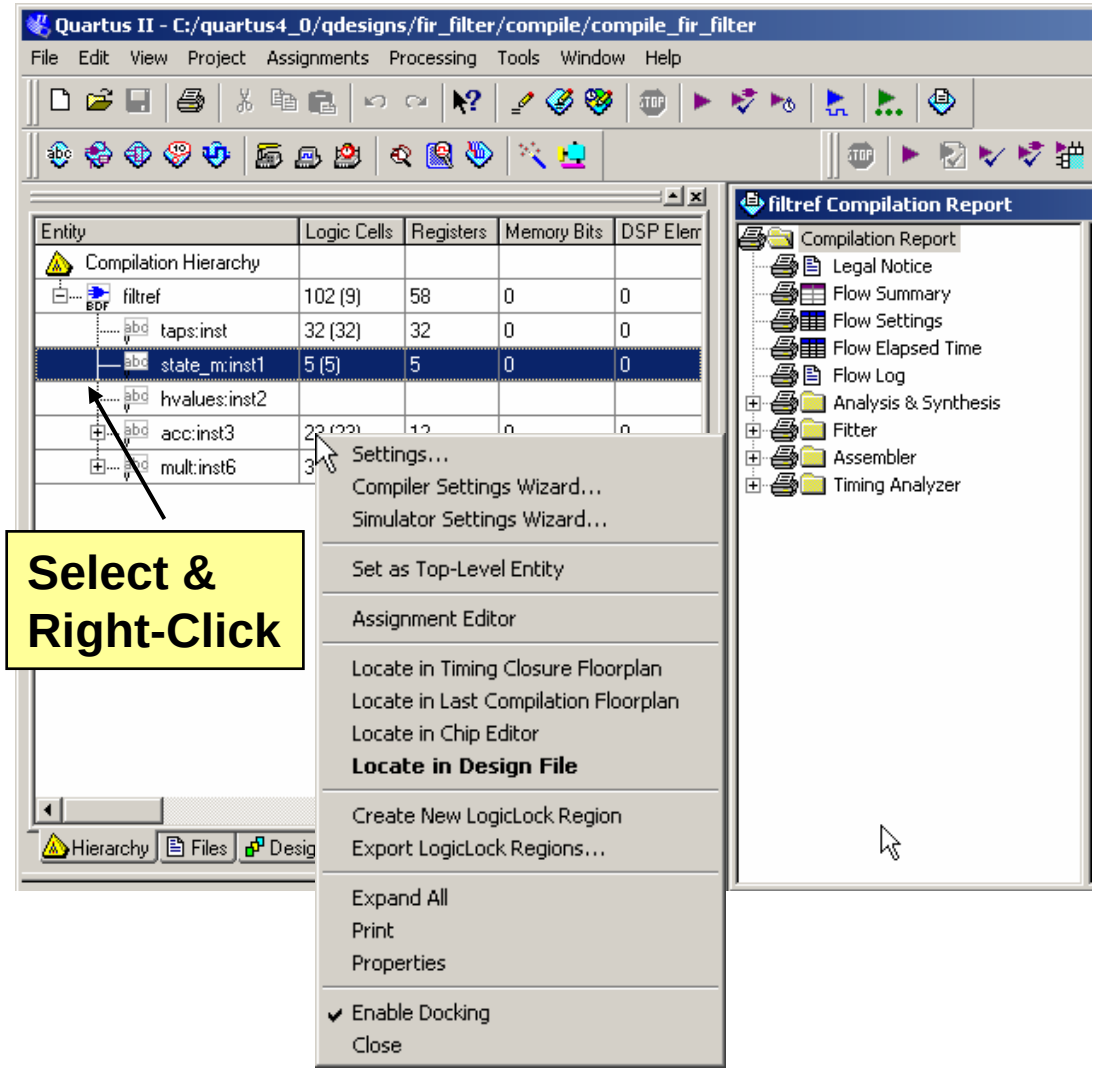


Change Settings:

- **General**
- **Files**
- **User Libraries**
- **Timing Requirement**
- **EDA Tool Settings**
- **Synthesis Settings**
- **Fitter Settings**
- **Timing Analyzer**
- **Design Assistant**
- **SignalTapII Analyzer**
- **SignalProbe Setting**
- **Simulator Settings**
- **Hardcopy Settings**

Copyright © 2004 Altera Corporation

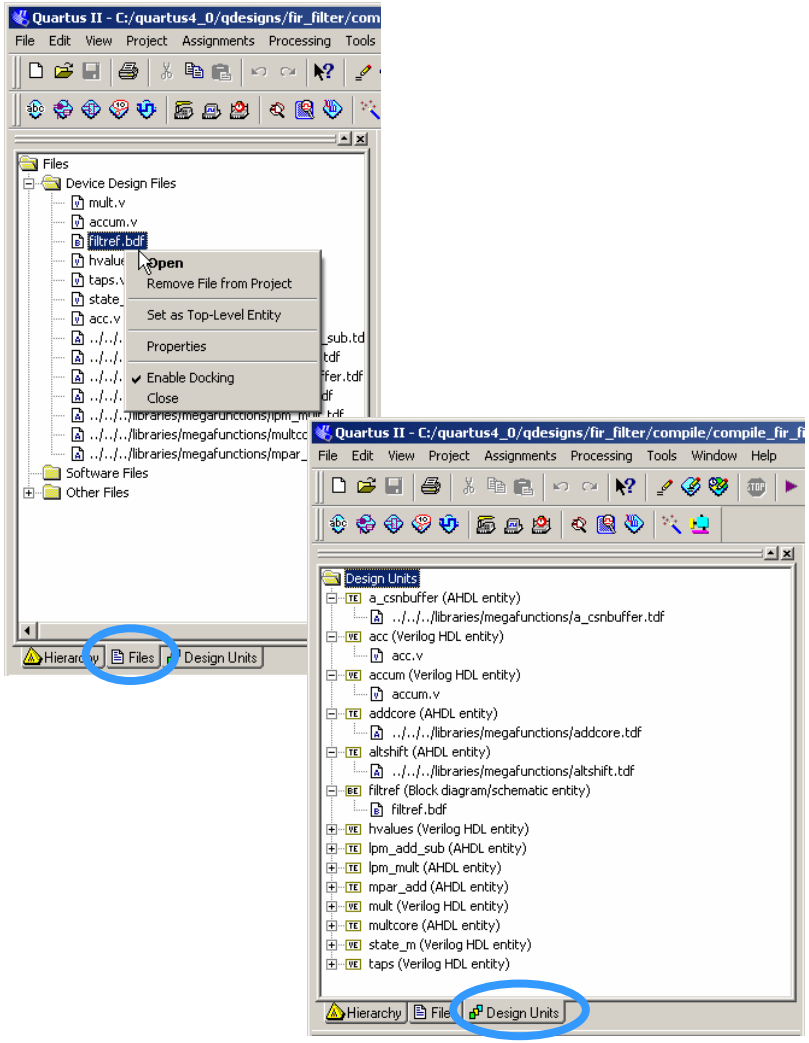
Project Navigator – Hierarchy Tab



- ⑤ Displays Project Hierarchy after Project Is Analyzed
- ⑤ Uses
  - Set Top-Level Entity
  - Make Assignments
  - Locate in Design Files, Floorplan or Chip Editor
  - View Resource Usage
  - Drag & Drop Hierarchical Blocks

Copyright © 2004 Altera Corporation

Files & Design Units Tabs



- ⑤ Files Tab
  - Shows Files Explicitly Added to Project
  - Uses
    - ③ Open Files
    - ③ Remove Files from Project
    - ③ Set New Top-Level Entity
- ⑤ Design Units Tab
  - Displays Design Unit & Type
    - ③ VHDL Entity
    - ③ VHDL Architecture
    - ③ Verilog Module
    - ③ AHDL Subdesign
    - ③ Block Diagram Filename
  - Displays File which Instantiates Design Unit

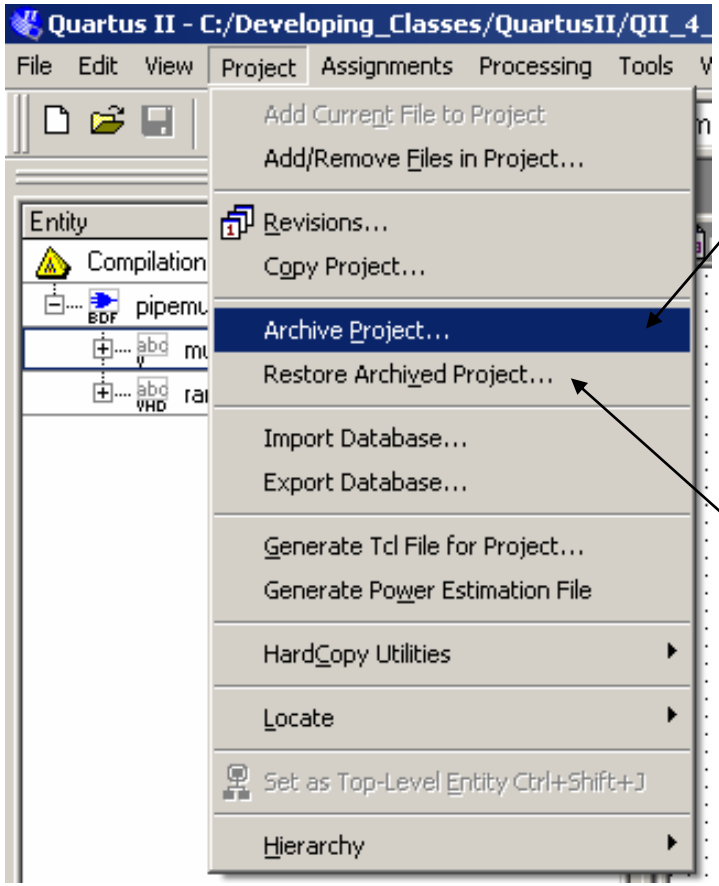
Copyright © 2004 Altera Corporation

Project Files

- ⑤ Quartus Project File (QPF)
  - Quartus II Version
  - Time Stamp
  - Active Revision
- ⑤ Quartus Default File (QDF)
  - Project Defaults
  - Name: assignment_defaults.qdf
  - Local or Bin Directory
    - ③ Local Read First

Copyright © 2004 Altera Corporation

Project Archive & Restore



Archive Project Creates:

1. **<project>.QAR**
   - Compressed Archive File
   - Options to Choose which Files Get Archived
2. **<project>.QARLOG**
   - Log of Archive Activity

Restore Archived Project:

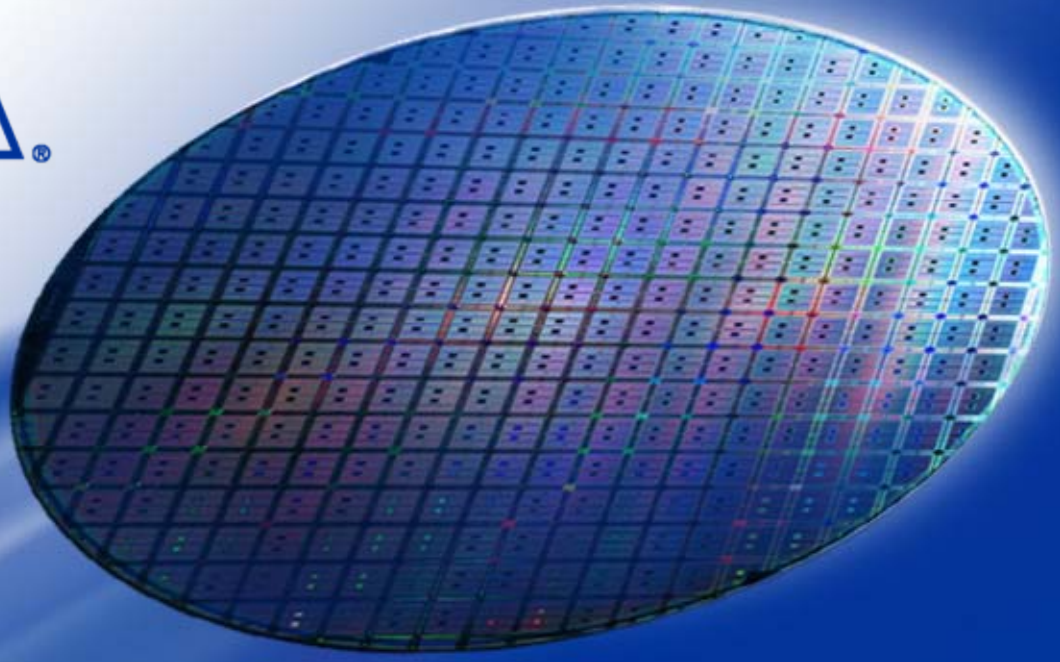
- Restores Complete Project & Library Function Files from **<project>.QAR** File

Copyright © 2004 Altera Corporation

Summary

- Use Project Wizard to create new projects
- Use Assignments Menu dialog boxes to
  - Edit existing project settings
  - Edit Third Party tools
- Use Project Navigator to study file and entity relationships within the project

Copyright © 2004 Altera Corporation



Quartus II Compilation

Copyright © 2004 Altera Corporation

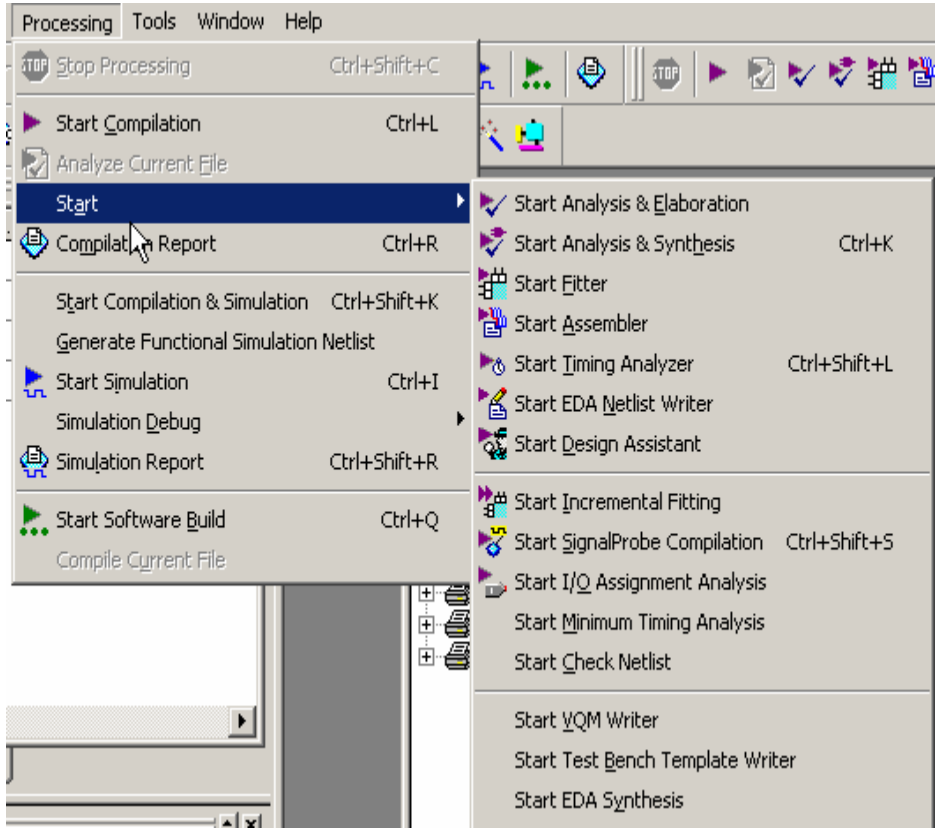
Quartus II Compilation

- ⑤Synthesis
- ⑤Fitting
- ⑤Generating Output
  - Timing Analysis Output Netlist
  - Simulation Output Netlists
  - Programming/Configuration Output Files

Copyright © 2004 Altera Corporation

Processing Options

- Start Compilation
  - Perform Full Compilation
- Start Analysis & Elaboration
  - Check Syntax & Build Database Only
- Start Analysis & Synthesis
  - Synthesize Code & Estimate Timing
- Start Fitter
- Start Assembler
- Start Timing Analysis
- Start I/O Assignment Analysis
- Start Design Assistant



Copyright © 2004 Altera Corporation

Status & Message Windows

Hit the Start Button

File Edit View Project Assignments Processing Tools Window Help

Entity	LC ...	LC ...	Memory Bits	DSP Ele
Compilation Hierarchy				
pipemult	62	77	512	0
mult:inst	41	77	0	0

Hierarchy | Files | Design Units

Module	Progress %	Time
Full Compilation	100 %	00:00:32
Analysis & Synthesis	100 %	00:00:06
Fitter	100 %	00:00:22
Assembler	100 %	00:00:01
Timing Analyzer	100 %	00:00:02

Compilation Report
- Legal Notice
- Flow Summary
- Flow Settings
- Flow Elapsed Time
- Flow Log
- Analysis & Synthesis
- Fitter
- Assembler
- Timing Analyzer

Flow Summary

Flow Status	Successful - Wed Dec 03 17:11:55 2003
Revision Name	pipemult
Top-level Entity Name	pipemult
Family	Stratix II
Total combinational functions	62
Total registers	77
Total pins	44 / 343 (12 %)
Total memory bits	512 / 419,328 (< 1 %)
DSP block 9-bit elements	0 / 96 (0 %)
Total PLLs	0 / 6 (0 %)
Total DLLs	0 / 2 (0 %)
Device	EP2S15F484C3
Total Logic Cells	80 / 12,480 (< 1 %)

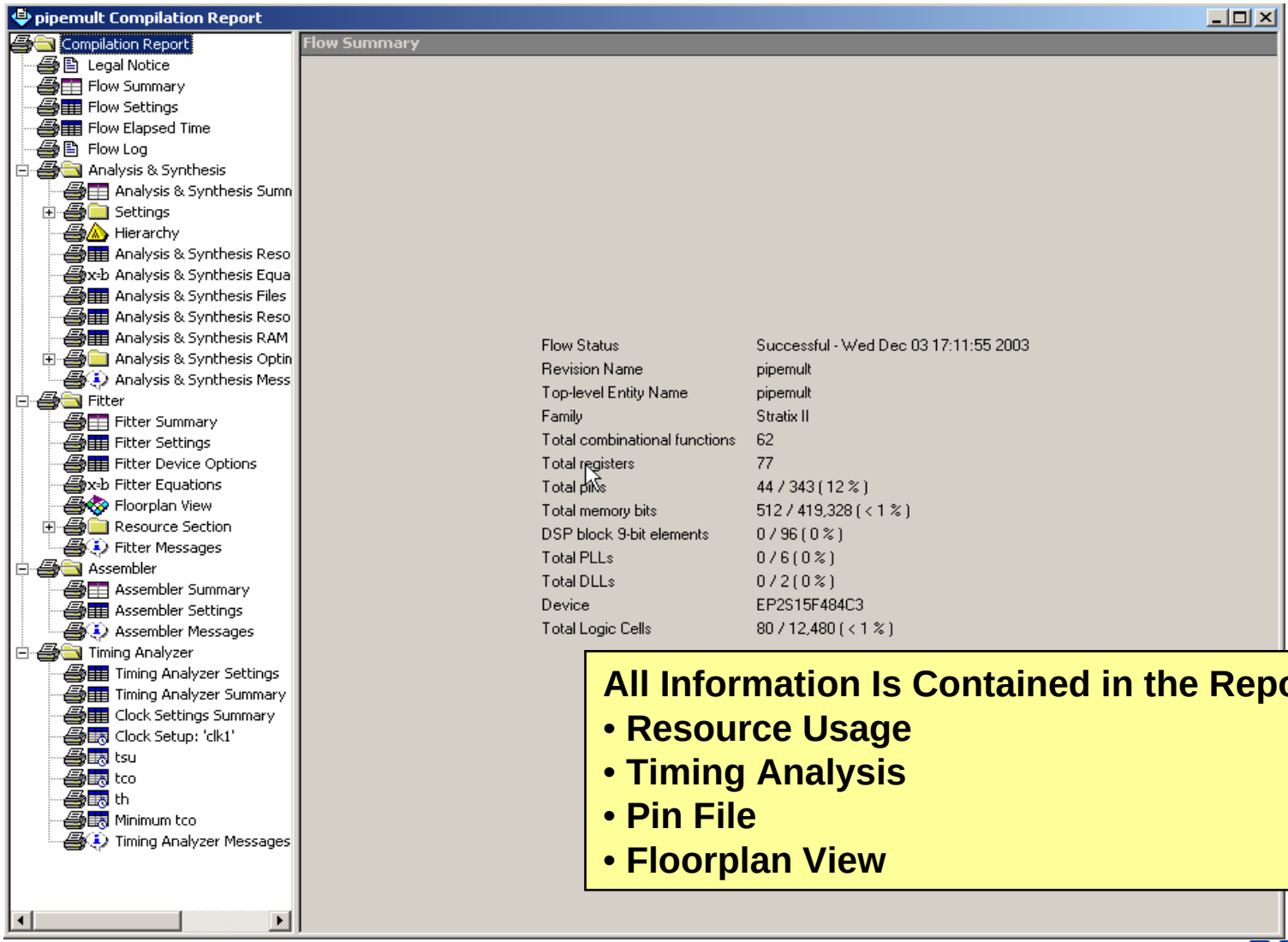
```
Info: ************************************************************
Info: Running Quartus II Analysis & Synthesis
Info: Command: quartus_map --import_settings_files=on --export_settings_files=off pipemult -c pipemult
Info: Found 1 design units and 1 entities in source file pipemult.bdf
```

Processing | System

- **Status Bars Scroll to Indicate Progress**
- **Message Window Displays Informational, Warning, & Error Messages**

Copyright © 2004 Altera Corporation

Compilation Report



All Information Is Contained in the Report

- **Resource Usage**
- **Timing Analysis**
- **Pin File**
- **Floorplan View**

Copyright © 2004 Altera Corporation

Resource Usage

pipemult Compilation Report

- Compilation Report
  - Legal Notice
  - Flow Summary
  - Flow Settings
  - Flow Elapsed Time
  - Flow Log
  - Analysis & Synthesis
    - Analysis & Synthesis Su
    - Settings
    - Hierarchy
    - Analysis & Synthesis R
    - Analysis & Synthesis Ec
    - Analysis & Synthesis Fi
    - Analysis & Synthesis R
    - Analysis & Synthesis R
    - Analysis & Synthesis O
    - Analysis & Synthesis M
  - Fitter
    - Fitter Summary
    - Fitter Settings
    - Fitter Device Options
    - Fitter Equations
    - Floorplan View
    - Resource Section
      - Fitter Resource Us
      - Input Pins
      - Output Pins
      - I/O Bank Usage
      - Output Pin Load Fc
      - Fitter Resource Uti
      - Delay Chain Summa
      - Pad To Core Delay
      - Control Signals
      - Global & Other Fas
      - Non-Global High Fa
      - Fitter RAM Summar
      - LAB and Routing Se
    - Fitter Messages
  - Assembler
    - Assembler Summary
    - Assembler Settings
    - Assembler Messages

Input Pins

	Name	Pin #	I/O Bank	X coordinate	Y coordinate	Cell number	Combinational Fan-Out	Registered Fan
1	clk1	X0_Y10_N1	1	0	10	1	1	0
2	dataa[0]	X40_Y8_N2	6	40	8	2	8	0
3	dataa[1]	X40_Y8_N0	6	40	8	0	8	0
4	dataa[2]	X40_Y9_N2	6	40	9	2	8	0
5	dataa[3]	X40_Y9_N1	6	40	9	1	8	0
6	dataa[4]	X40_Y7_N3	6	40	7	3	8	0
7	dataa[5]	X40_Y10_N0	6	40	10	0	8	0
8	dataa[6]	X40_Y10_N3	6	40	10	3	8	0
9	dataa[7]	X40_Y7_N0	6	40	7	0	8	0
10	datab[0]	X40_Y10_N2	6	40	10	2	8	0
11	datab[1]	X40_Y10_N1	6	40	10	1	8	0
12	datab[2]	X40_Y9_N3	6	40	9	3	8	0
13	datab[3]	X40_Y9_N0	6	40	9	0	8	0
14	datab[4]	X40_Y7_N1	6	40	7	1	8	0
15	datab[5]	X40_Y7_N2	6	40	7	2	8	0
16	datab[6]	X40_Y8_N1	6	40	8	1	8	0
17	datab[7]	X40_Y8_N3	6	40	8	3	8	0
18	rdaddress[0]	X40_Y24_N1	5	40	24	1	1	0
19	rdaddress[1]	X35_Y27_N2	4	35	27	2	1	0
20	rdaddress[2]	X40_Y24_N0	5	40	24	0	1	0
21	rdaddress[3]	X40_Y25_N2	5	40	25	2	1	0
22	rdaddress[4]	X37_Y27_N3	4	37	27	3	1	0
23	wraddress[0]	X40_Y24_N2	5	40	24	2	1	0
24	wraddress[1]	X37_Y27_N2	4	37	27	2	1	0
25	wraddress[2]	X37_Y27_N0	4	37	27	0	1	0
26	wraddress[3]	X35_Y27_N1	4	35	27	1	1	0
27	wraddress[4]	X35_Y27_N0	4	35	27	0	1	0
28	wren	X40_Y24_N3	5	40	24	3	1	0

Several Sections Detail the Resource Usage

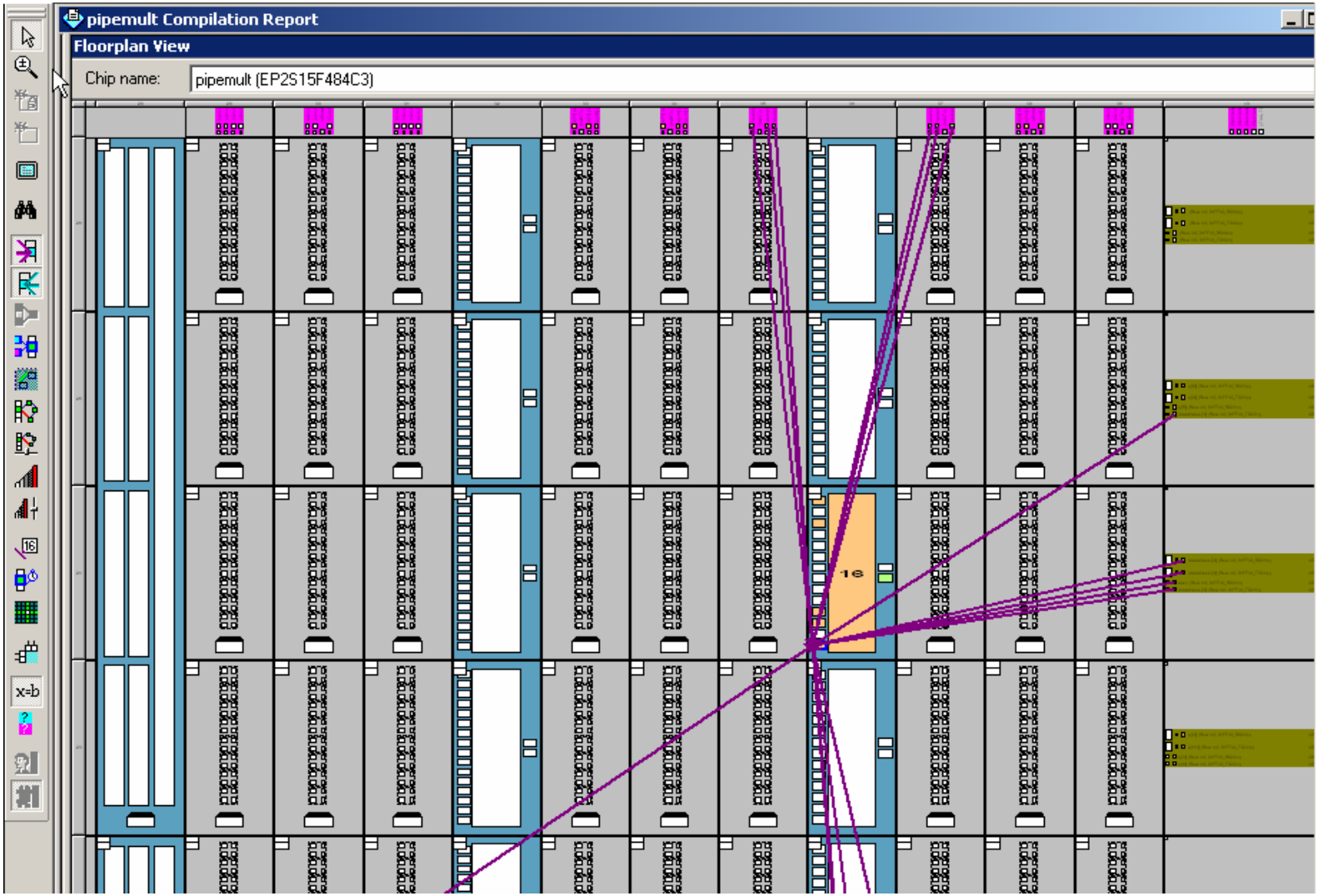
Copyright © 2004 Altera Corporation

Floorplan

- Available Views
  - Last Compilation
  - Chip Editor
  - Timing Closure
- Uses
  - Make Assignments
  - View Placement
  - View Connections

Copyright © 2004 Altera Corporation

Floorplan View



Copyright © 2004 Altera Corporation

Controlling Synthesis & Fitting

⑤ Two Methods to Control Logic Synthesis & Fitting Operations

- Settings
  - ③ Project-Wide Switches
- Assignments (i.e. Logic Options; Constraints)
  - ③ Individual Entity/Node Controls

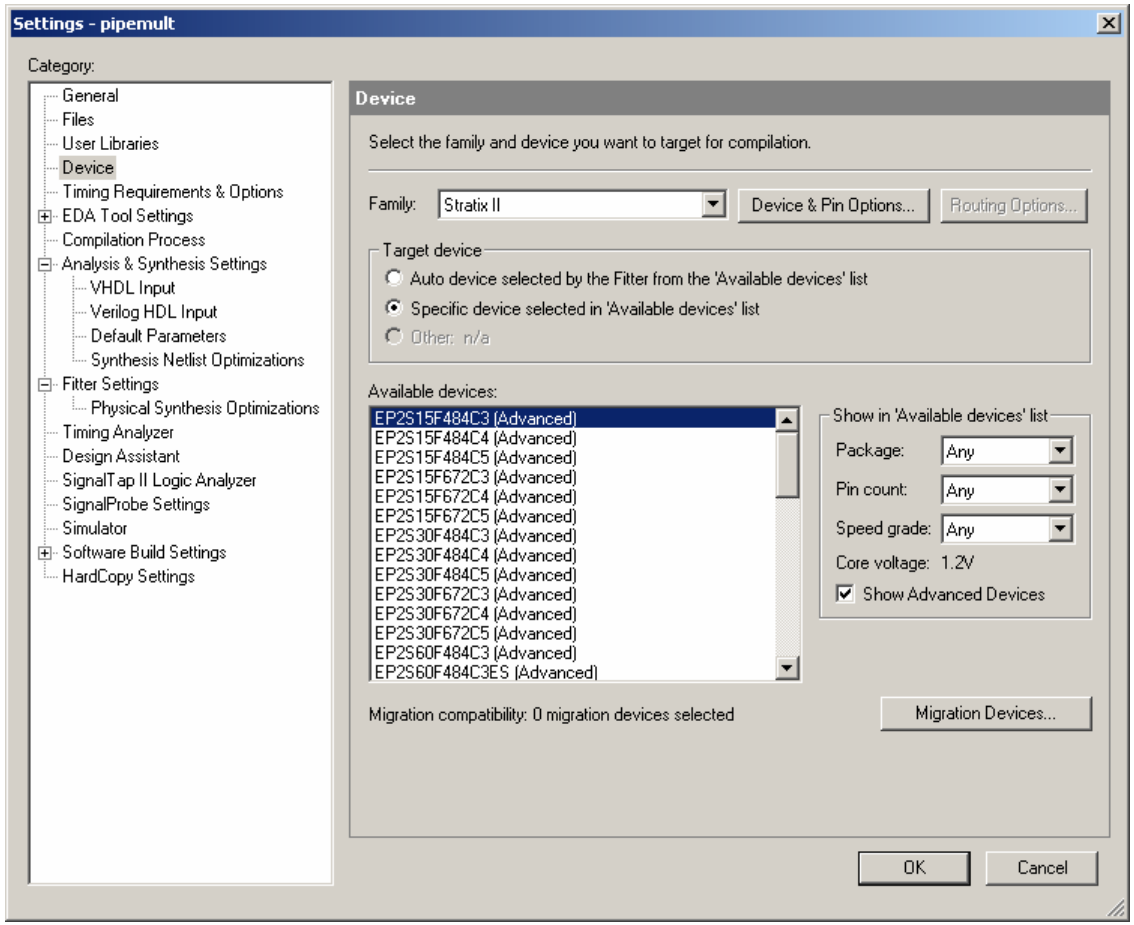
Copyright © 2004 Altera Corporation

Settings

- Examples
  - Device Selection
  - Synthesis Optimization
  - Fitter Optimization
  - Physical Synthesis
- Located in Settings Dialog Box (Assignments Menu)

Copyright © 2004 Altera Corporation

Settings Dialog Box

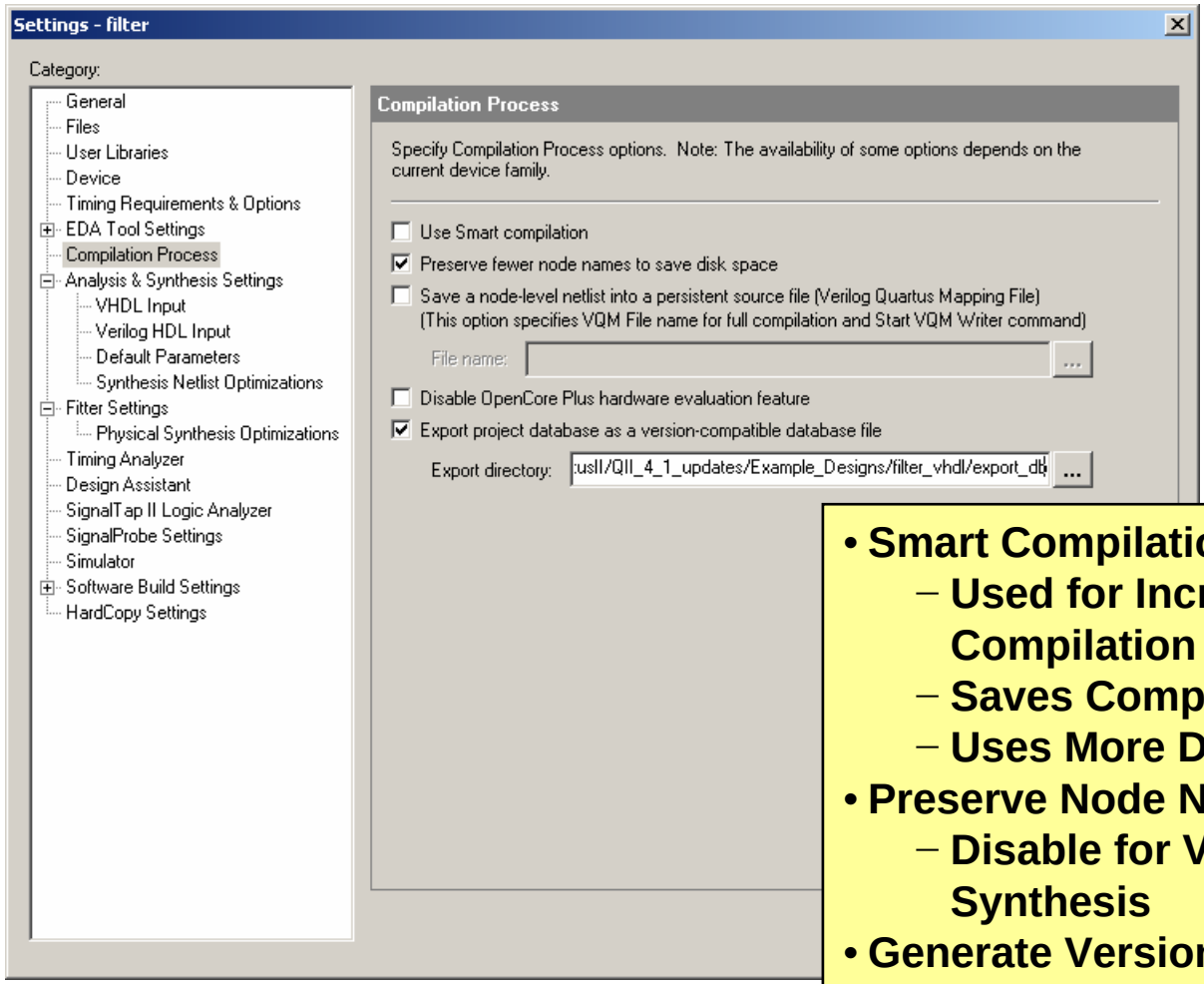


Change Settings

- **Top-Level Entity**
- **Target Device**
- **Add/Remove Files**
- **Libraries**
- **VHDL ‘87, ‘93?**
- **Verilog ‘95, ‘01?**
- **EDA Tool Settings**
- **Timing Settings**
- **Compiler Settings**
- **Simulator Settings**

Copyright © 2004 Altera Corporation

Compilation Process



- **Smart Compilation**
  - **Used for Incremental Re-Compilation**
  - **Saves Compiler Time**
  - **Uses More Disk Space**
- **Preserve Node Names**
  - **Disable for VHDL/Verilog Synthesis**
- **Generate Version-Compatible Database**

Copyright © 2004 Altera Corporation

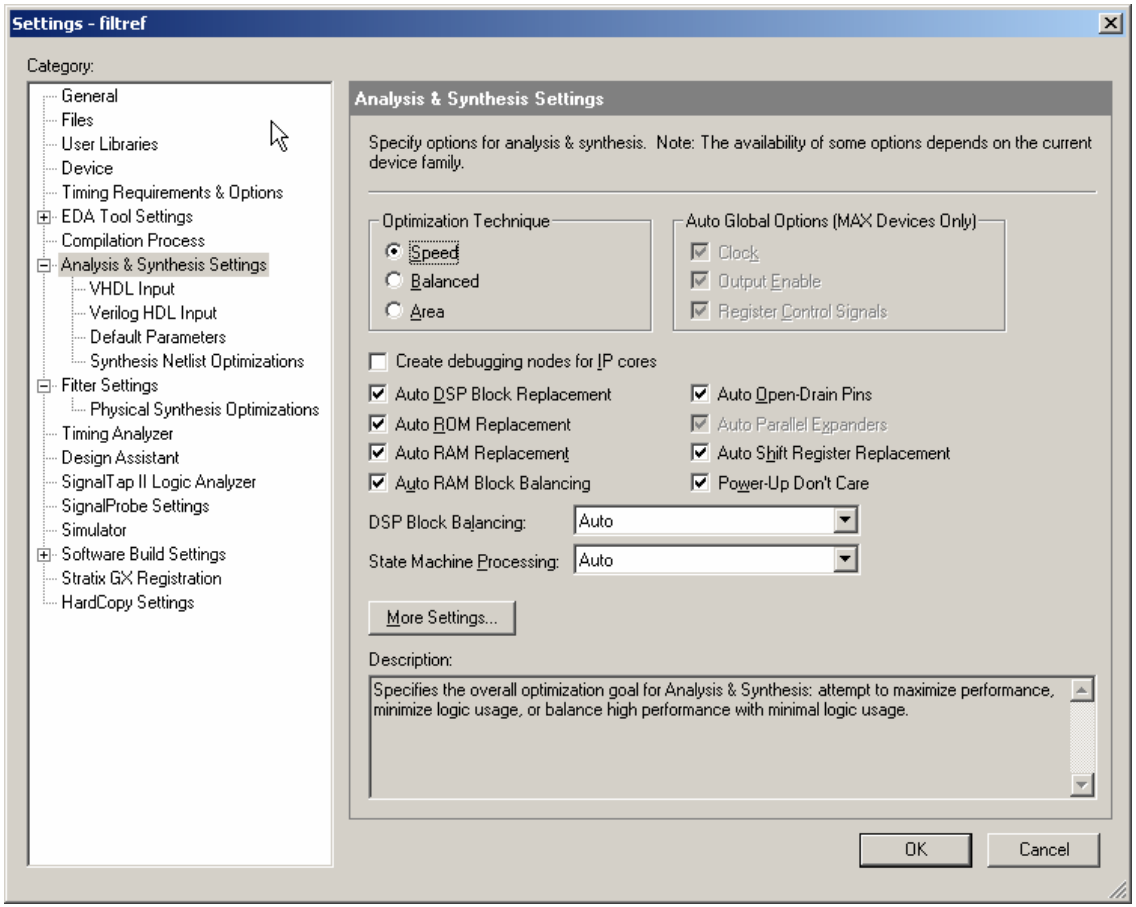
Version-Compatible Database

- Recommended
- Exports a Database that Can Be Opened Directly in another Version of Quartus II
  - Import Database into New Version
- Two Methods to Create
  - Settings Dialog Box
  - Project Menu

Analyze Previously Compiled Projects Using Updated Timing Models

Copyright © 2004 Altera Corporation

Synthesis Options



Global Optimization Goal

- Select Speed vs. Area or Balanced

Logic Replacement

- Replace Logic with Equivalent Megafunction

State Machine Processing

- Auto, One-Hot or Minimal Bit

Seed

- Controls Initial Placement Configuration
- Different Seeds Give Different Fmax

Copyright © 2004 Altera Corporation

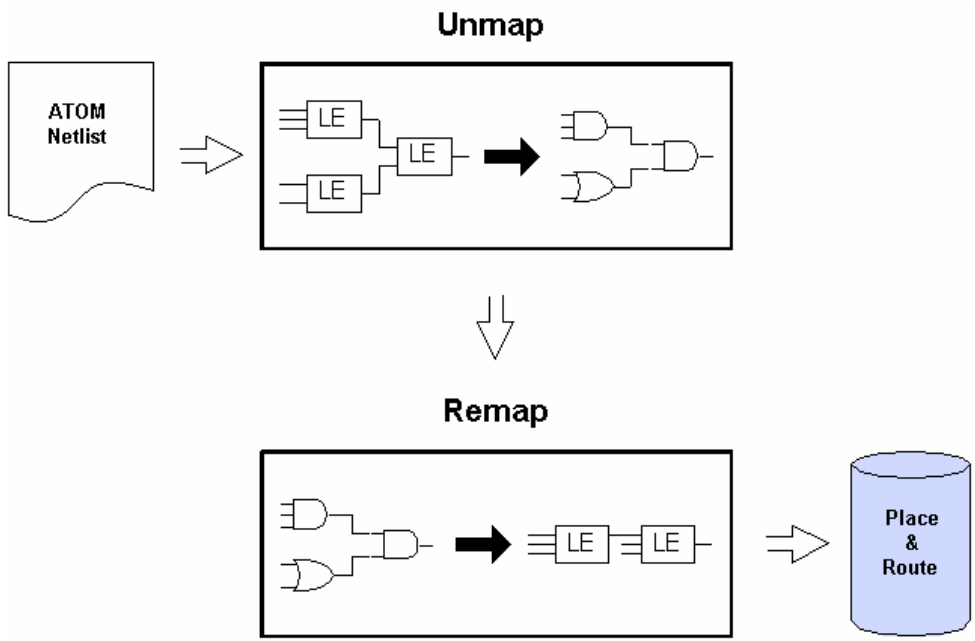
Synthesis Netlist Optimizations

- ⑤Further Optimize Netlists during Synthesis
- ⑤Types
  - WYSIWYG Primitive Resynthesis
  - Gate-Level Register Retiming

Copyright © 2004 Altera Corporation

WYSIWYG Primitive Resynthesis

- Unmaps 3rd-Party Atom Netlist Back to Gates & then Remaps to Altera Primitives
  - Unavailable when Using Integrated Synthesis
- Considerations
  - Node Names May Change
  - 3rd-Party Synthesis Attributes May Be Lost
    - Preserve/Keep
  - Some Registers May Be Synthesized Away

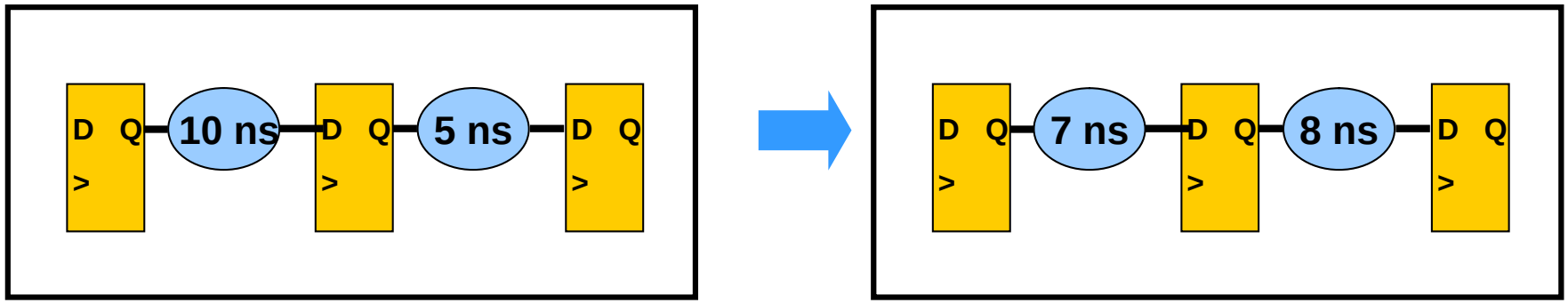


4-5 % Improvement

Copyright © 2004 Altera Corporation

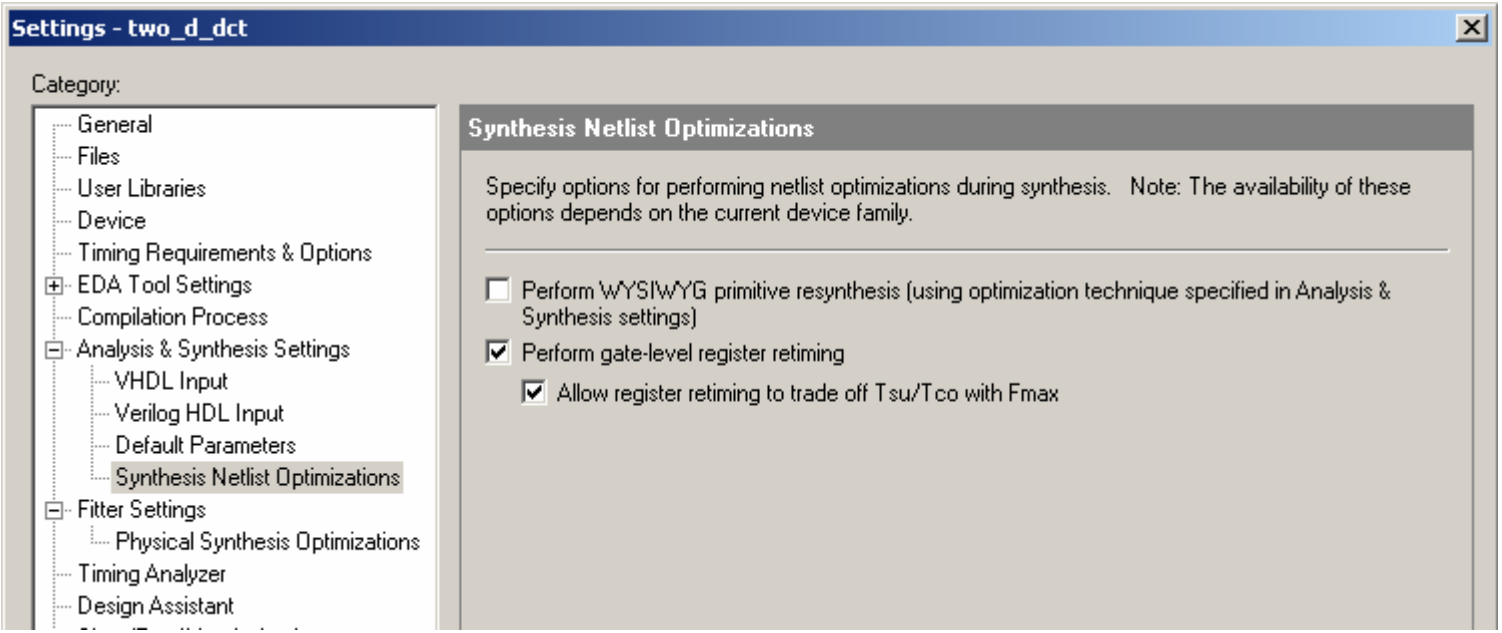
Gate-Level Register Retiming

- Moves Registers across Combinatorial Logic to Balance Timing
- Trades between Critical & Non-Critical Paths
- Makes Changes at Gate Level



Copyright © 2004 Altera Corporation

Enabling Synthesis Optimizations



Created/Modified Nodes Noted in Compilation Report

Copyright © 2004 Altera Corporation

Fitter Optimizations

Timing Driven Compilation

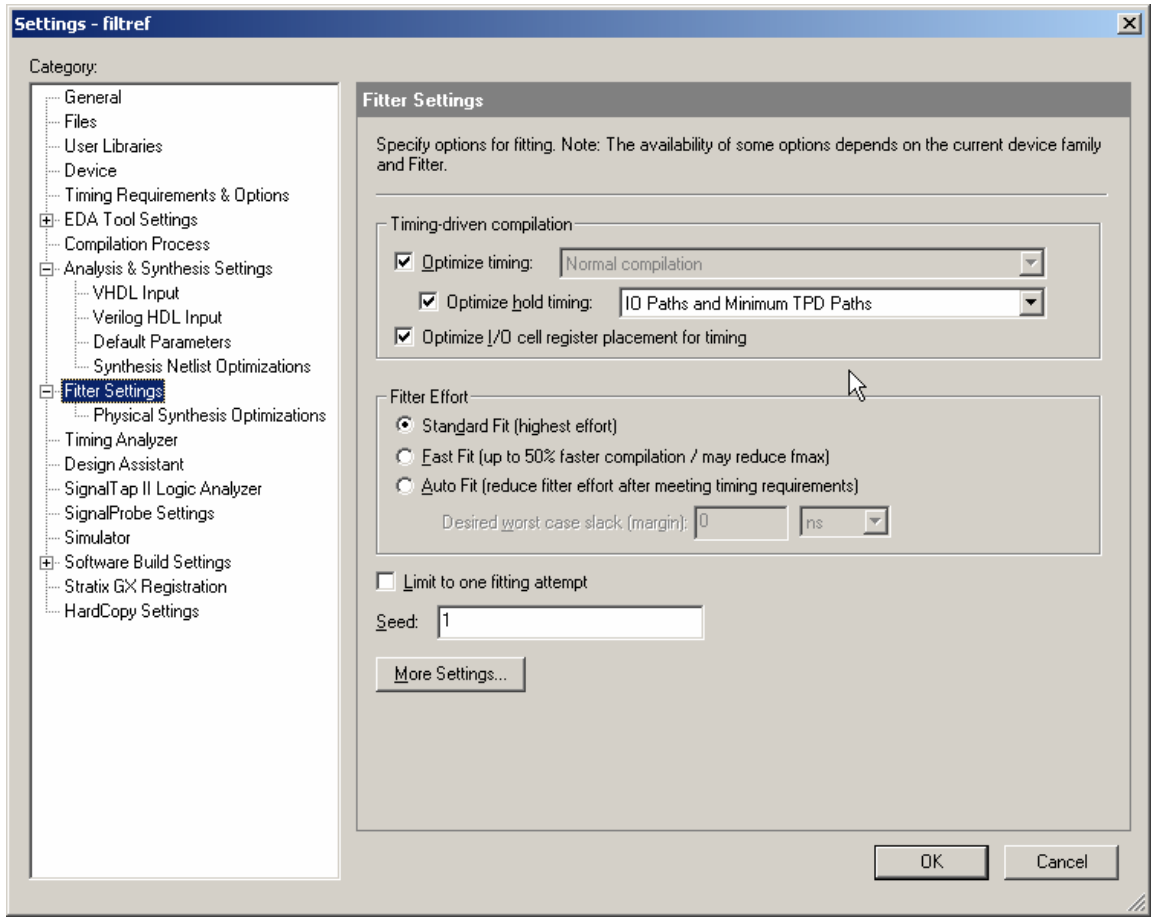
- Optimize Internal Timing
- Optimize I/O Register Placement

Compilation Speed/Fitter Effort

- Standard Fit (Highest Effort)
- Fast Fit (Faster Compile but Possibly Lesser Design Performance)
- Auto Fit (Compile Stops after Meeting Timing)
- One Fitting Attempt

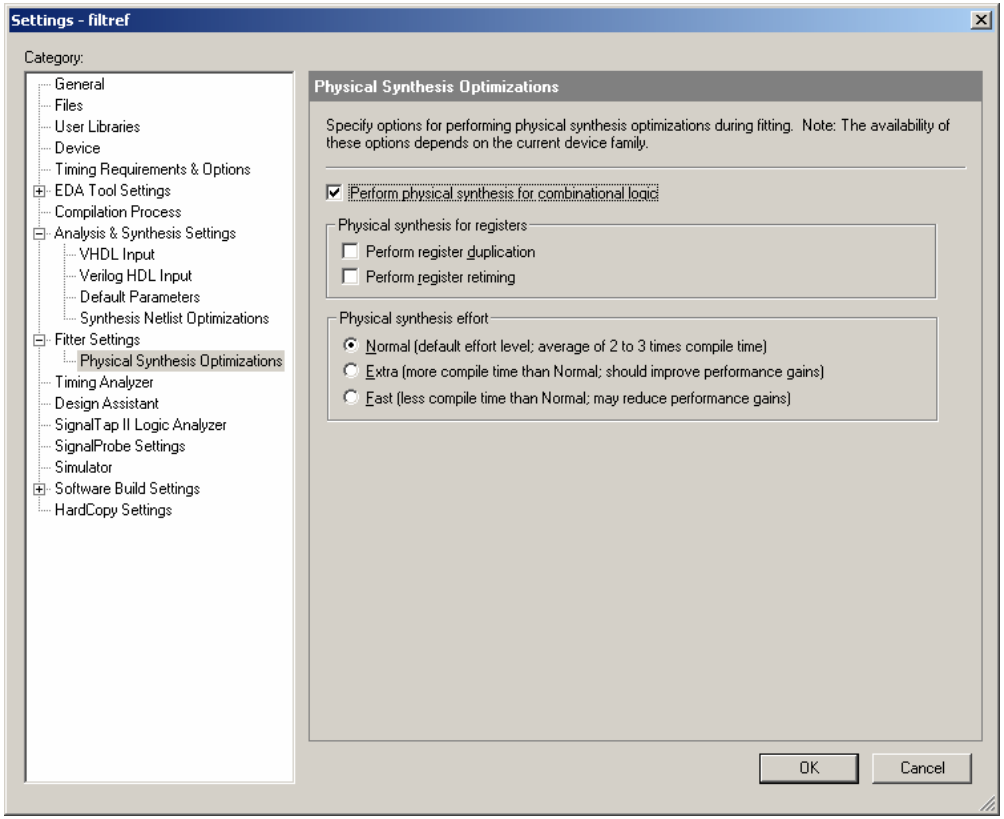
Seed

- Controls Initial Placement Configuration



Copyright © 2004 Altera Corporation

Physical Synthesis

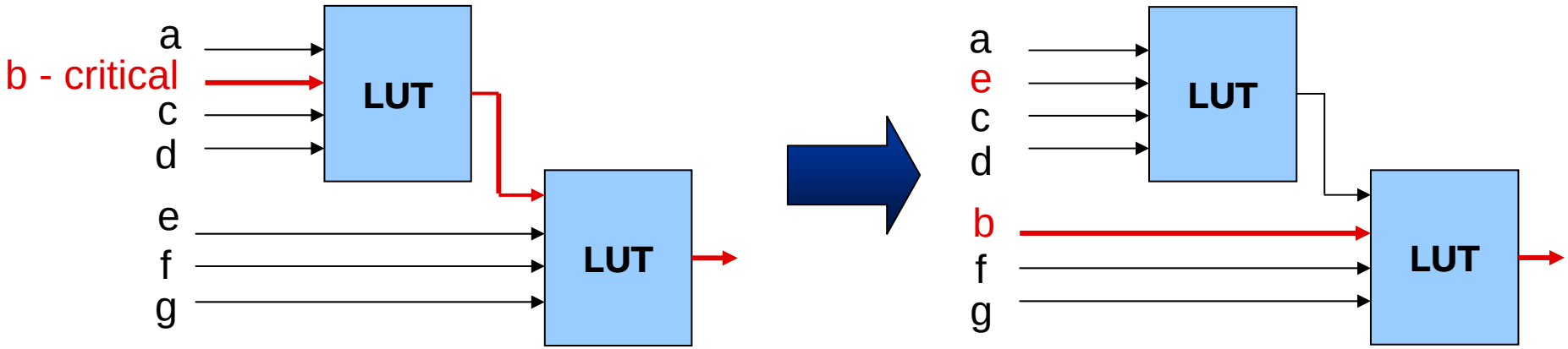


- ⑤ Re-Synthesis Based on Fitter Output
  - Makes Incremental Changes that Improve Results for a Given Placement
  - Compensates for Routing Delays from Fitter
- ⑤ Types
  - Combinational Logic
  - Registers
    - ③ Register Duplication
    - ③ Register Retiming
- ⑤ Effort
  - Trades Performance vs Compile Time
  - Normal, Extra or Fast

Copyright © 2004 Altera Corporation

Combinational Logic

- Swaps Look-Up Table (LUT) Ports within LEs to Reduce Critical Path LEs



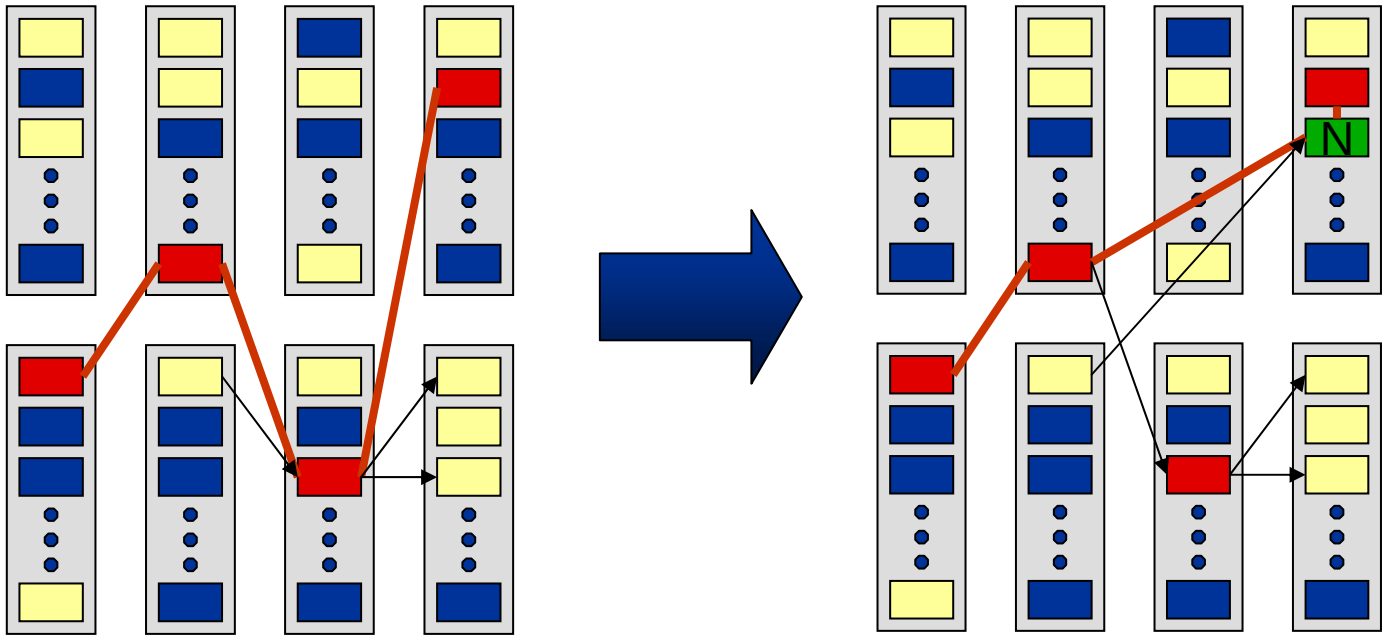
- Allows LUT Duplication to Enable Further Optimizations on the Critical Path

Copyright © 2004 Altera Corporation

Register Duplication

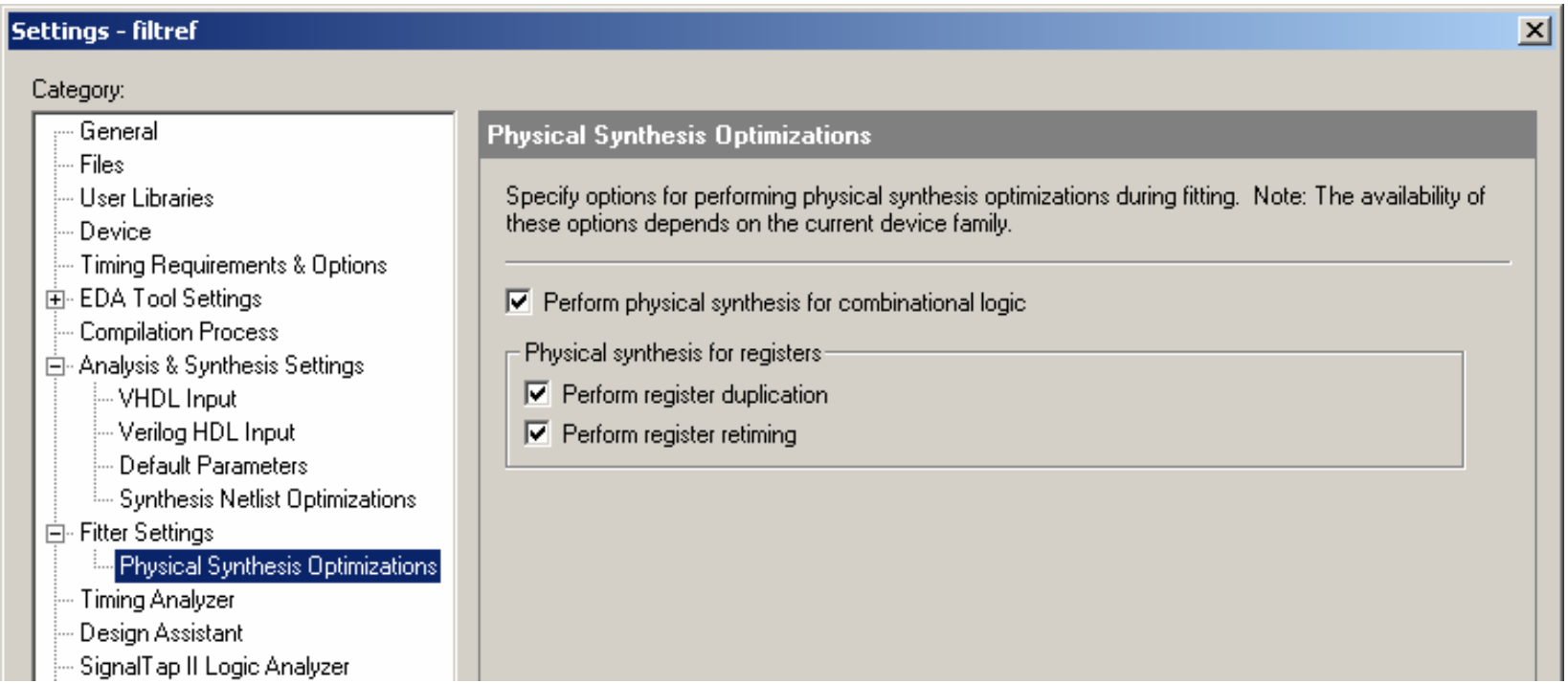
⑤High Fan-Out Register Is Duplicated & Placed to Reduce Delay

– Combinational Logic May Also Be Duplicated



Copyright © 2004 Altera Corporation

Enabling Physical Synthesis



Created/Modified Nodes Noted in Compilation Report

Copyright © 2004 Altera Corporation

Assignments

- Assignment Editor
- Example Assignments
- I/O Assignments & Analysis

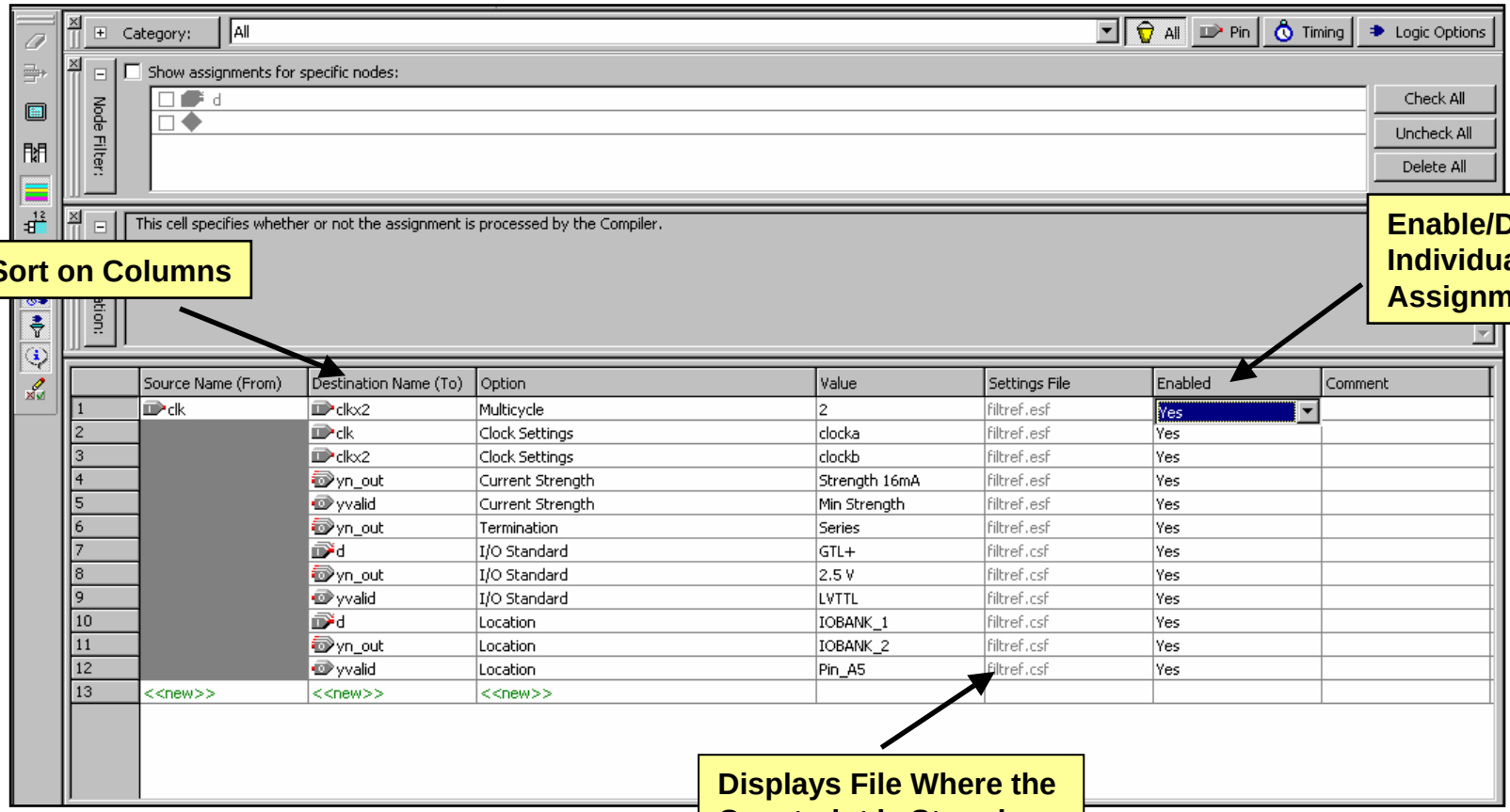
- Perform Analysis & Elaboration before Obtaining Hierarchy & Node Information

Copyright © 2004 Altera Corporation

Assignment Editor (AE)

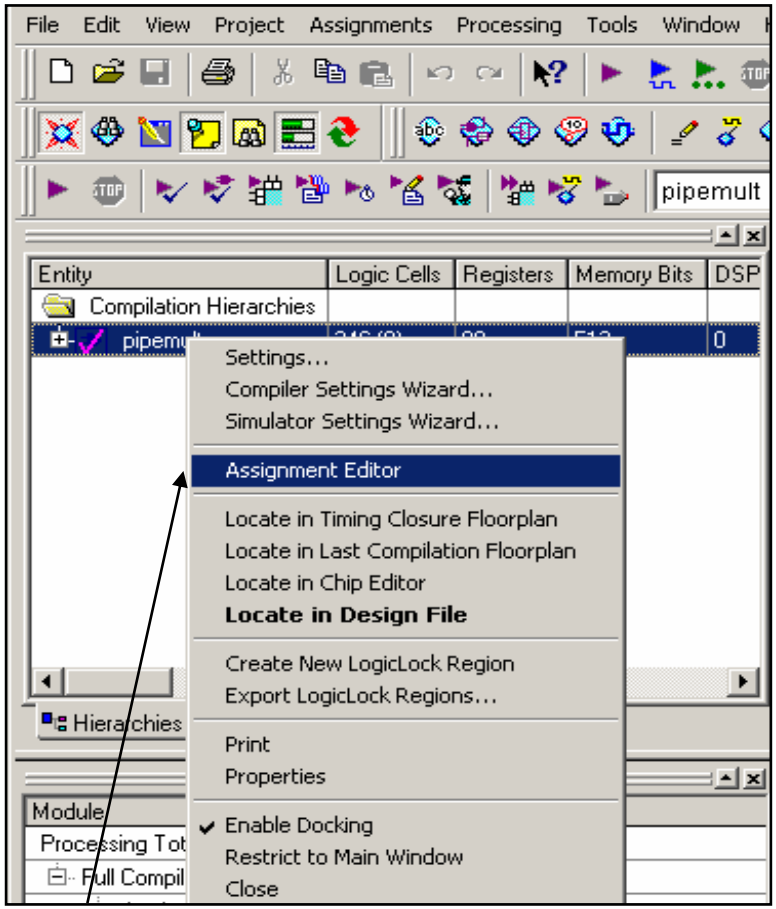
⑤ Provides Spreadsheet Assignment Entry & Display

– Can Copy & Paste from Clipboard

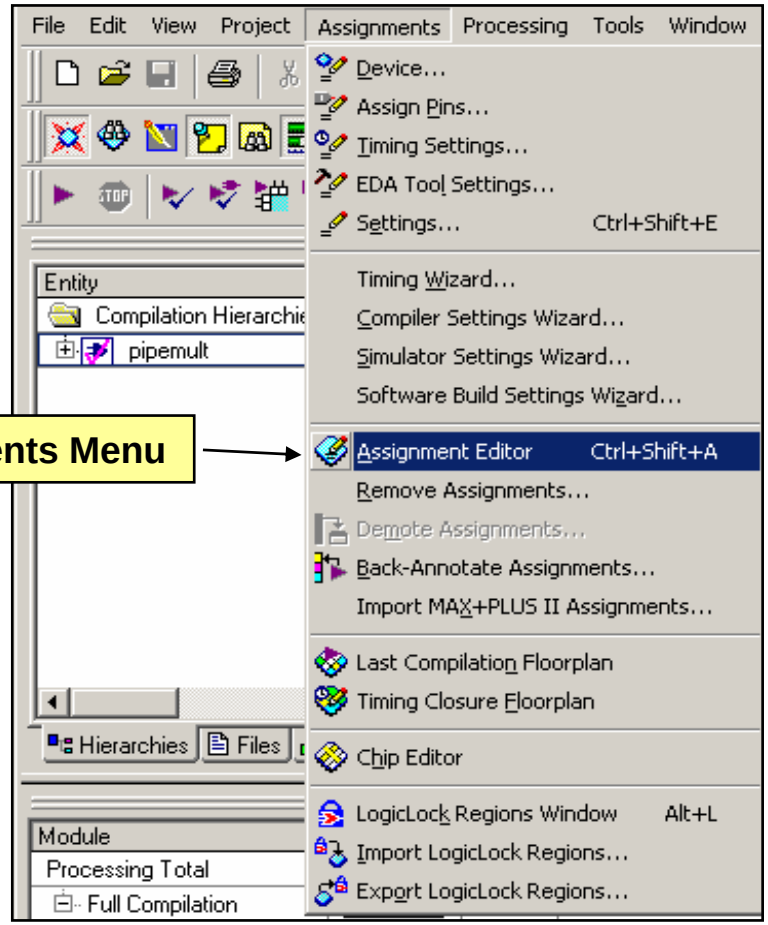


Copyright © 2004 Altera Corporation

Opening Assignment Editor



Assignments Menu

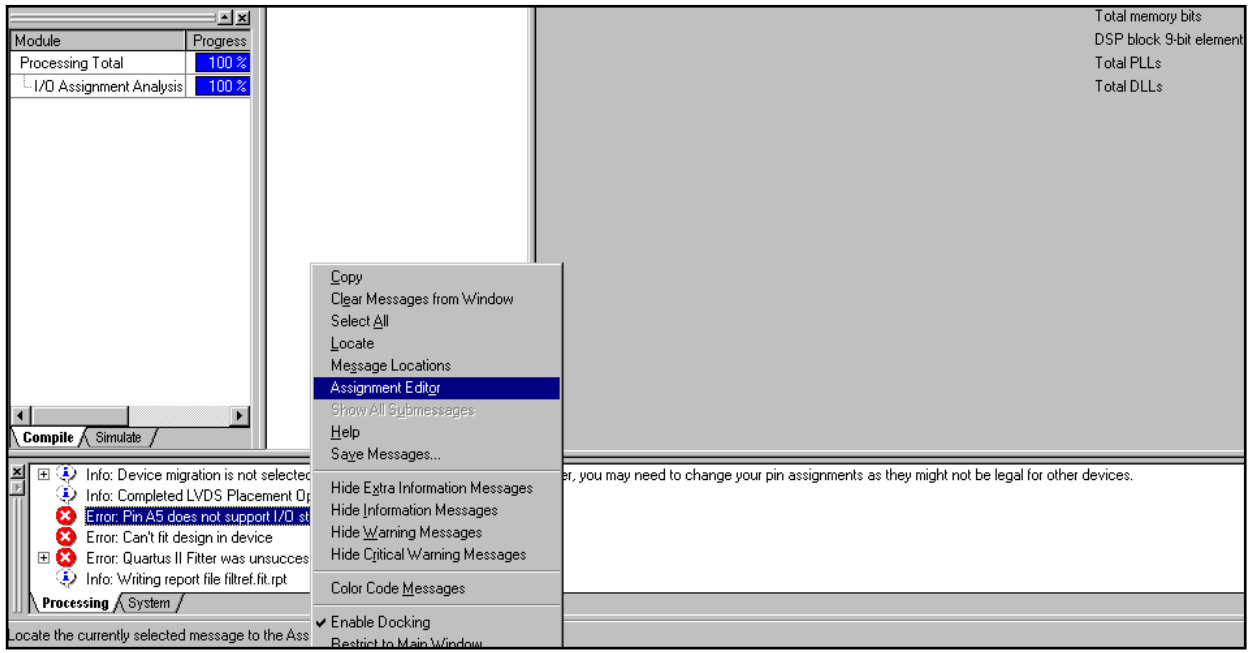


Invoke the Assignment Editor by Highlighting an Entity in the Hierarchy View & Right- Clicking

Copyright © 2004 Altera Corporation

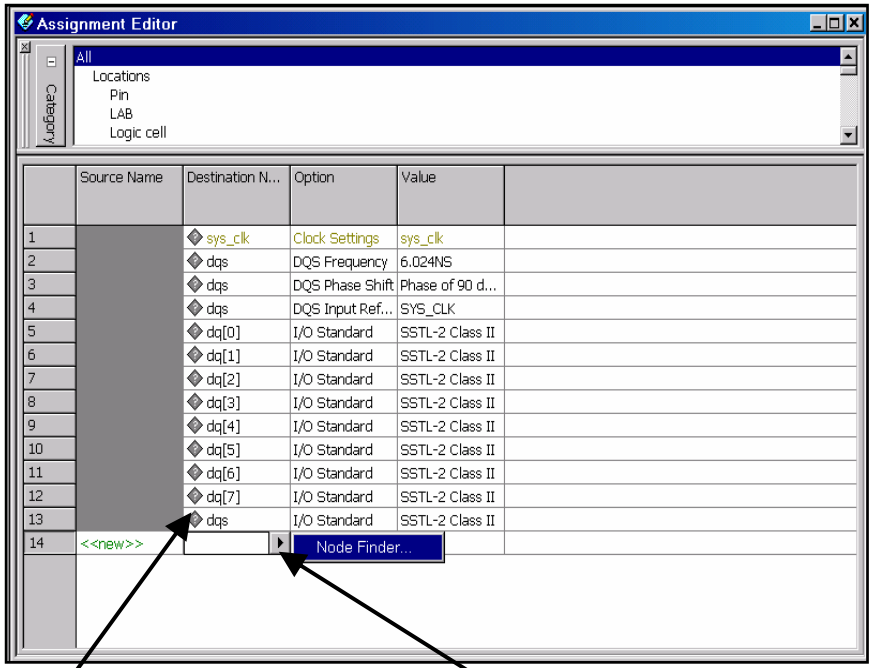
Opening Assignment Editor (cont.)

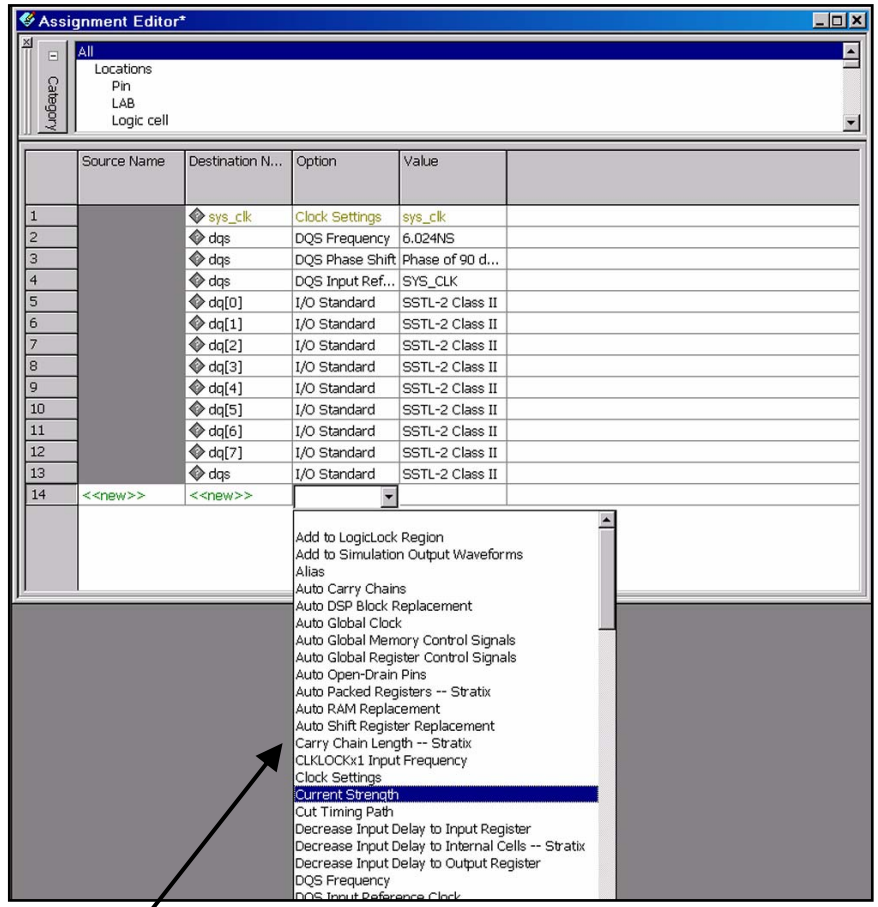
⑤ Locate to Assignment Editor from Timing Report, Messages, Etc



Copyright © 2004 Altera Corporation

Using Assignment Editor





Double-Click Cells to Edit or Add New Assignment

Launches Node Finder

Selecting an Option Determines which Fields Are Available for Editing

Copyright © 2004 Altera Corporation

Editing Multiple Assignments

⑤ Edit Multiple Constraints Simultaneously

– Select Multiple Pins & Use Edit Bar (Click Check Mark to Accept)

Category: Pin | All | Pin | Timing | Logic Options

Node Filter: Click the Node Filter button to view more options

Information: Specifies the I/O standard of a pin. Different device families support different I/O standards, and restrictions apply to placing pins with different I/O standards toge

Edit: 2.5 V

	Name	Location	I/O Bank	I/O Standard	tsu Requirement	Clock Settings	tco Requirement	Cu
1	d	IOBANK_6	6	1.8 V				
2	newt	IOBANK_8	8	2.5 V	5.2ns			
3	next	IOBANK_7	7	1.8 V			4ns	
4	reset	EDGE_BOTTOM		1.8 V				
5	yn_out[0]	IOBANK_1	1	2.5 V				
6	yn_out[1]	IOBANK_1	1	2.5 V				
7	yn_out[2]	IOBANK_1	1	2.5 V				
8	yn_out[3]	IOBANK_1	1	2.5 V				
9	yn_out[4]	IOBANK_1	1	2.5 V				
10	yn_out[5]	IOBANK_1	1	2.5 V				
11	yn_out[6]	IOBANK_1	1	2.5 V				
12	yn_out[7]	IOBANK_1	1	2.5 V				
13	yvalid	Pin_B18	3	2.5 V				
14	clk	Pin_A4	4	1.8 V		clocka		
15	clkx2	Pin_A5	4	1.8 V		clockb		
16	<<new>>	<<new>>						

Editing Multiple I/O Standards at Once

Copyright © 2004 Altera Corporation

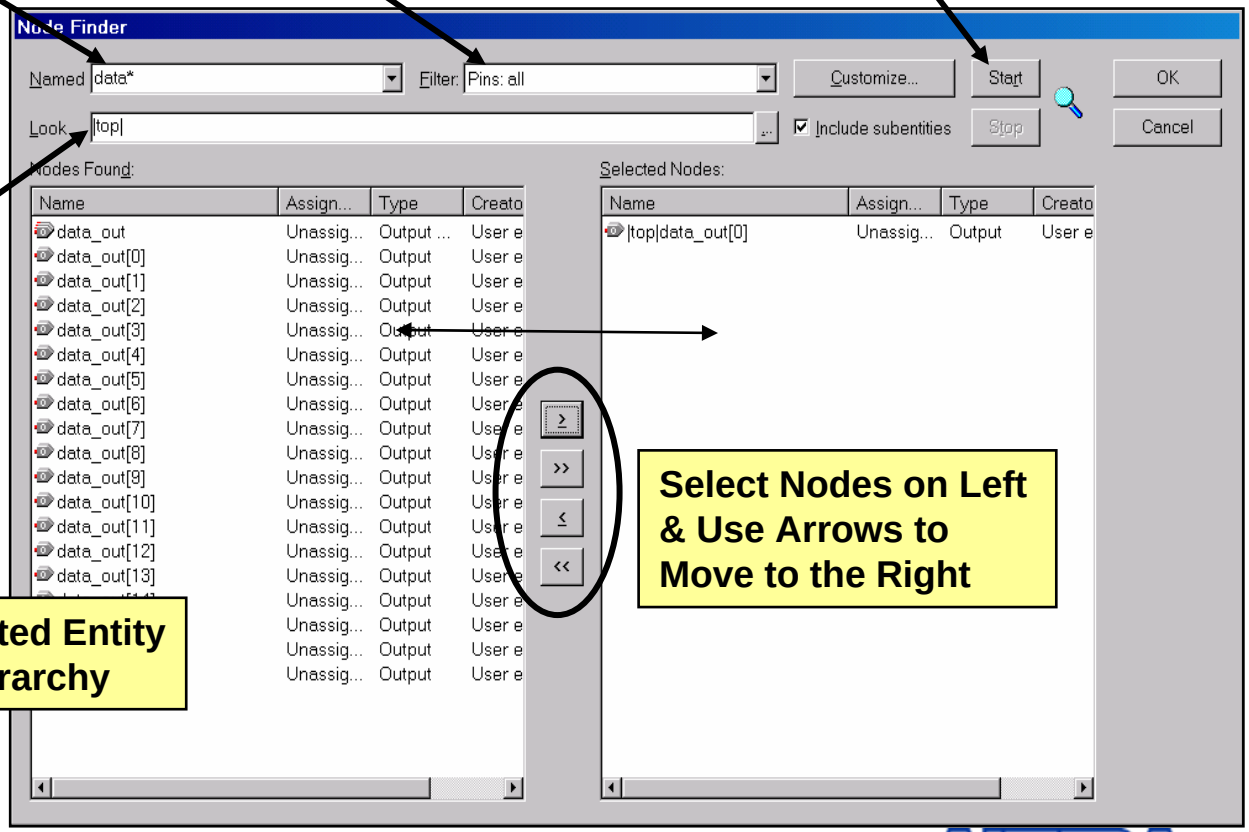
Node Finder

Search by Name Using Wildcards (? or *)

Use Filter to Select the Nodes to Be Displayed

Start Displays Nodes Meeting Search Criteria

Locate Nodes in a Certain Level of Hierarchy



Select Nodes on Left & Use Arrows to Move to the Right

List of Nodes in Selected Entity & Lower Levels of Hierarchy

Copyright © 2004 Altera Corporation

AE Dynamic Checking

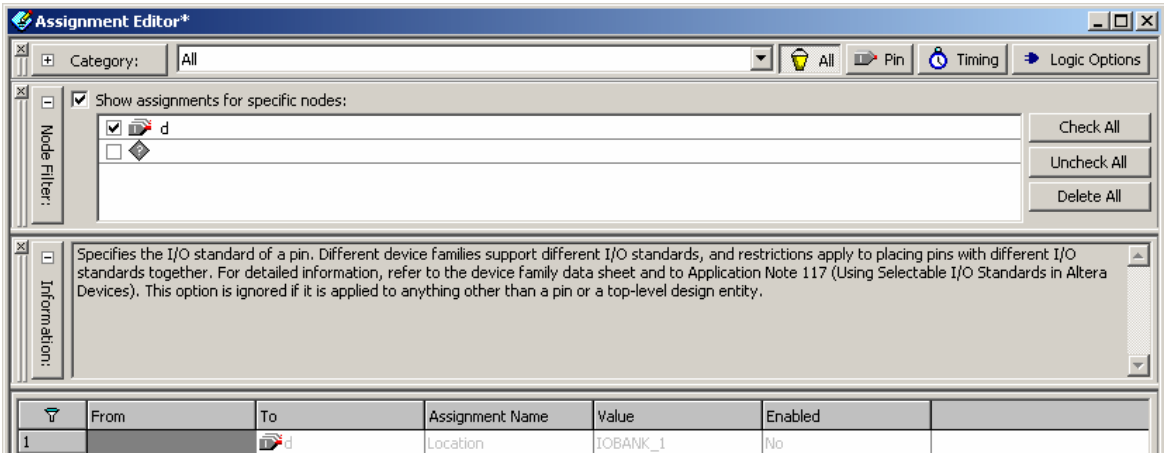
- Validity of Constraint Checked during Entry
- Color-Coded to Display Status
  - Grey – Disabled
  - Black – Applied
  - Yellow – Not Validated
  - Dark Red – Incomplete
  - Bright Red – Error/Illegal Value
  - Green – Enter New Assignment

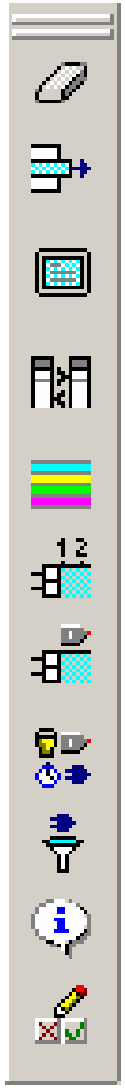
	From	To	Assignment Name	Value	Enabled
1		yn_out	Location	IOBANK_2	Yes
2		yvalid	Location	PIN_75	Yes
3		d	Location	IOBANK_1	No
4		clk	Clock Settings	clk	Yes
5		unknown_clock	Clock Settings	clk2	Yes
6		clkx2	Clock Settings	clk2	Yes
7	clk	clkx2	Multicycle	2	Yes
8		d	DQS Frequency	1MHz	Yes
9		yvalid		Minimum Current	Yes
10		d	I/O Standard	LVCMOS	Yes
11		yn_out	I/O Standard	LVCMOS	Yes
12		yvalid	I/O Standard	LVCMOS	Yes
13	<<new>>	<<new>>	<<new>>		

Copyright © 2004 Altera Corporation

Assignment Editor Features

- ⑤ Category Bar
  - Selects Category of Assignments to View
    - ③ Ex. Pin Assignments, Timing Assignments
  - Each Can Be Customized
- ⑤ Node Filter Bar
  - Filters Constraints Displayed Based on Node Name
- ⑤ Information Bar
  - Displays Description of Selected Cell or Assignment





Copyright © 2004 Altera Corporation

AE Tcl Commands

⑤ Equivalent Tcl Commands Are Displayed as Assignments Are Entered

- Manually Copy to Create Tcl Scripts
- Export Command (File Menu) Writes All Assignments to a Tcl File

Assignment Editor*

Category: All

Information: Assigns user-defined clock characteristics to a signal. To designate a particular signal as a clock,

	Source Name (From)	Destination Name (To)	Option	Value
1		hvalues:inst2	Add to LogicLock Reg...	kdjff
2		state_m:inst1	Add to LogicLock Reg...	kdjff
3		clkx2	Clock Settings	clk200
4		d[0]	Location	PIN_C1
5		d[1]	Location	Pin_W9
6		d[2]	Location	Pin_T4
7		d[3]	Location	Pin_AC9
8		d[5]	Location	Pin_AC16
9		d[6]	Location	Pin_AD15
10		d[7]	Location	Pin_Y16
11		yn_out[0]	Location	Pin_D20
12		yn_out[1]	Location	Pin_A21
13		yn_out[2]	Location	Pin_C21
14		yn_out[4]	Location	Pin_B20
15		yn_out	tco Requirement	5ns
16		clk	Clock Settings	clk100
17			Location	Pin_E3
18			Location	Pin_E4

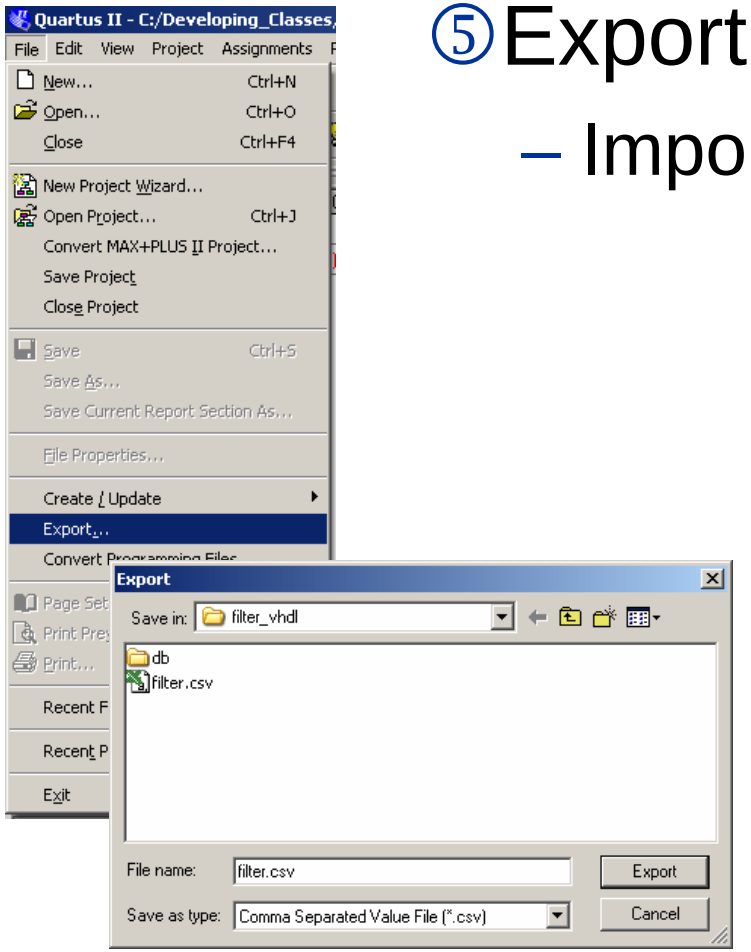
```
Info: set_location_assignment -to "d\[2\]" "Pin_T4"
Info: set_location_assignment -to "d\[1\]" "Pin_W9"
Info: set_location_assignment -to "d\[3\]" "Pin_AC9"
Info: set_location_assignment -to "d\[7\]" "Pin_Y16"
Info: set_location_assignment -to "d\[6\]" "Pin_AD15"
Info: set_location_assignment -to "d\[5\]" "Pin_AC16"
Info: set_location_assignment -to "yn_out\[0\]" "Pin_D20"
Info: set_location_assignment -to "yn_out\[1\]" "Pin_A21"
Info: set_location_assignment -to "yn_out\[2\]" "Pin_C21"
Info: set_location_assignment -to "yn_out\[4\]" "Pin_B20"
Info: set_location_assignment -to "d\[0\]" -remove
Info: set_location_assignment -to "yn_out" "Pin_C1"
Info: set_location_assignment -to "yn_out" -remove
Info: set_instance_assignment -to "yn_out" -name "TCO_REQUIREMENT" "5ns" -entity "filtref"
Info: set_instance_assignment -to "clk" -name "CLOCK_SETTINGS" -remove -entity "filtref"
Info: set_location_assignment -to "clk" "Pin_D2"
Info: set_location_assignment -to "clk" -remove
Info: set_instance_assignment -to "clk" -name "CLOCK_SETTINGS" "clk100" -entity "filtref"
```

Processing | System

Message Window

Copyright © 2004 Altera Corporation

Export CSV File Assignments (Excel)



⑤Export to CSV File (File Menu)

– Import Data into Excel

From	To	Assignment Name	Value
	~DATA0~	Location	PIN_L8
	d[7]	Location	PIN_J4
	d[6]	Location	PIN_H4
	d[5]	Location	PIN_E6
	d[4]	Location	PIN_F1
	d[3]	Location	PIN_H3
	d[2]	Location	PIN_J6
	d[1]	Location	PIN_G4
	d[0]	Location	PIN_F2
	yn[7]	Location	PIN_H2
	yn[6]	Location	PIN_L6
	yn[5]	Location	PIN_G2
	yn[4]	Location	PIN_J2
	yn[3]	Location	PIN_G1
	yn[2]	Location	PIN_J3
	yn[1]	Location	PIN_H1
	yn[0]	Location	PIN_K2
	clk	Location	PIN_L2
	reset	Location	PIN_L3
	newt	Location	PIN_K6
	yvalid	Location	PIN_E5
	nxt	Location	PIN_A6
	yn	Output Maximum Delay	4ns
clk	input data	Input Maximum Delay	7ns
mult_mega:u4|*	acc:u5|*	Cut Timing Path	On
	clk	Clock Settings	clk

Copyright © 2004 Altera Corporation

Example Assignments

- ⑤Optimization Technique
- ⑤PCI I/O
- ⑤Output Pin Load

Copyright © 2004 Altera Corporation

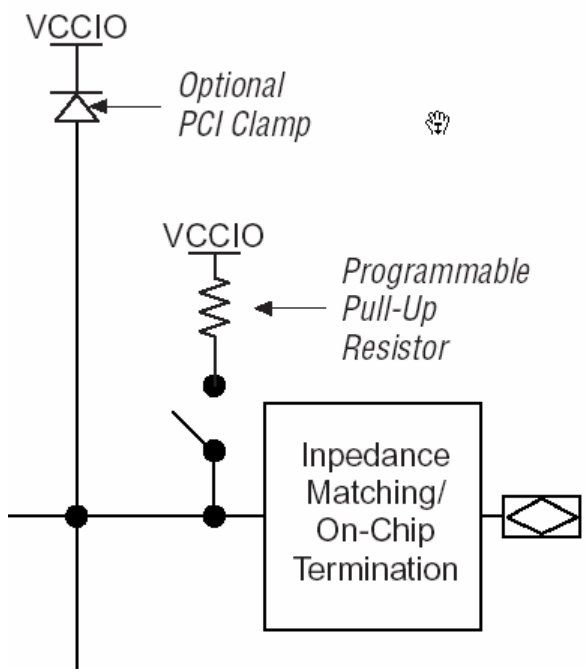
OPTIMIZATION TECHNIQUE

- Selects Synthesis Optimization Goal
  - Speed
  - Area
  - Balanced (Default)
- Applies Only to Hierarchical Entities
- Effects Synthesis & Logic Mapping
- Only Applies to Quartus II Integrated Synthesis

Copyright © 2004 Altera Corporation

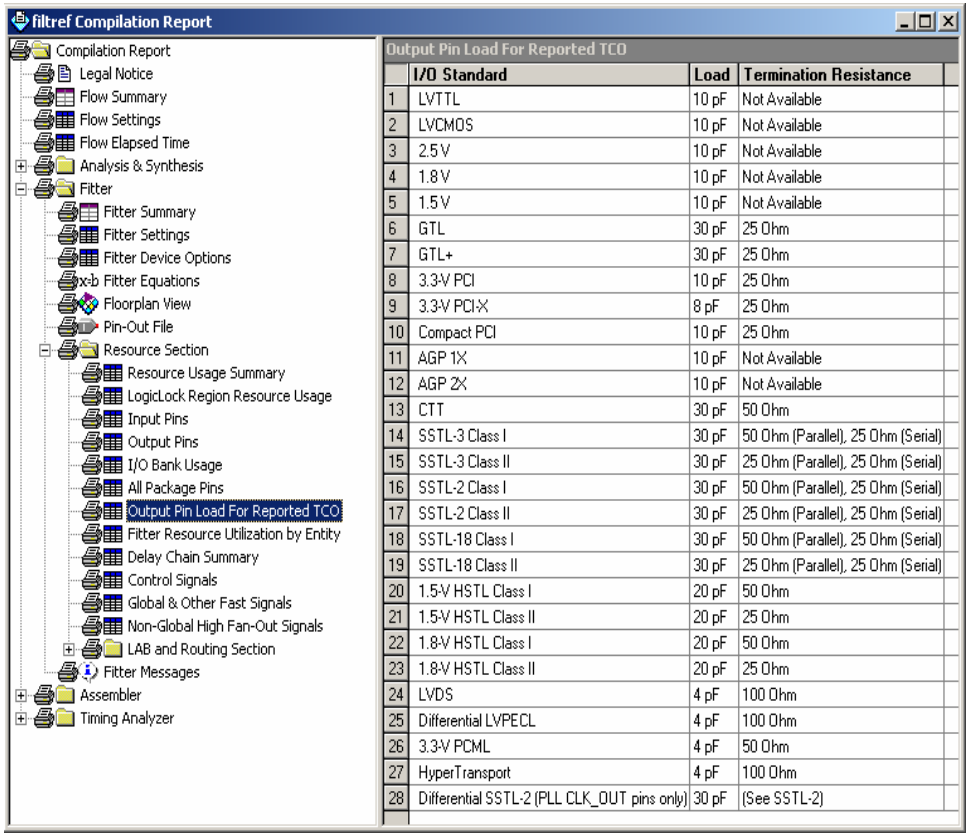
PCI I/O

- Turns on PCI Compatibility for Pins
  - Ignored If Applied to Anything other than a Pin or a Top-Level Design Entity
- Controls Clamping Diode Located in the I/O Elements



Copyright © 2004 Altera Corporation

Output Pin Load



Output Pin Load For Reported TCO

	I/O Standard	Load	Termination Resistance
1	LVTTL	10 pF	Not Available
2	LVCMOS	10 pF	Not Available
3	2.5 V	10 pF	Not Available
4	1.8 V	10 pF	Not Available
5	1.5 V	10 pF	Not Available
6	GTL	30 pF	25 Ohm
7	GTL+	30 pF	25 Ohm
8	3.3-V PCI	10 pF	25 Ohm
9	3.3-V PCI-X	8 pF	25 Ohm
10	Compact PCI	10 pF	25 Ohm
11	AGP 1X	10 pF	Not Available
12	AGP 2X	10 pF	Not Available
13	CTT	30 pF	50 Ohm
14	SSTL-3 Class I	30 pF	50 Ohm (Parallel), 25 Ohm (Serial)
15	SSTL-3 Class II	30 pF	25 Ohm (Parallel), 25 Ohm (Serial)
16	SSTL-2 Class I	30 pF	50 Ohm (Parallel), 25 Ohm (Serial)
17	SSTL-2 Class II	30 pF	25 Ohm (Parallel), 25 Ohm (Serial)
18	SSTL-18 Class I	30 pF	50 Ohm (Parallel), 25 Ohm (Serial)
19	SSTL-18 Class II	30 pF	25 Ohm (Parallel), 25 Ohm (Serial)
20	1.5-V HSTL Class I	20 pF	50 Ohm
21	1.5-V HSTL Class II	20 pF	25 Ohm
22	1.8-V HSTL Class I	20 pF	50 Ohm
23	1.8-V HSTL Class II	20 pF	25 Ohm
24	LVDS	4 pF	100 Ohm
25	Differential LVPECL	4 pF	100 Ohm
26	3.3-V PCML	4 pF	50 Ohm
27	HyperTransport	4 pF	100 Ohm
28	Differential SSTL-2 (PLL CLK_OUT pins only)	30 pF	(See SSTL-2)

- Specifies Output Pin Loading in picoFarads (pF)
  - Changes Default Loading Value of I/O Standard
  - Changes t_{co} of Output Pins
- Allows Designer to Accurately Model Board Conditions
- Must Be Applied to Output or Bidirectional Pins

Copyright © 2004 Altera Corporation

Available Logic Options (Assignments)

1) Go to Quartus II Help (Index)
2) Type in "Logic Options"
3) Click on "list of"

Supported Devices Shown for each Assignment

Quartus II Help Version 4.1

Back Forward Stop Refresh Home Print

Contents Index Search

Type in the keyword to find:

list of

logic options
- changing assignments
- list of
- making assignments
- overview topics
- resetting assignments
- specifying default

Logic Options (Assignment Editor)
logic primitives
Logic Section
- (see also Assert Statement)
- (see also Boolean control equations)
- (see also Boolean equations)
- (see also Case Statement)
- (see also Defaults Statement)
- (see also For Generate Statement)
- (see also If Generate Statement)
- (see also If Then Statement)
- (see also in-line logic function reference)
- (see also Truth Table Statement)
- general description
- indentation guidelines
- recommended position in Text Design File
- state transitions
- syntax for
- template for
- using in AHDL

logic synthesis
- (see also logic synthesis options)
- optimizing netlists
- overview topics
- performing on files
- stopping
- using Fitter information to optimize netlists
- viewing processing time

logic synthesis directives (see synthesis dire
logic synthesis options
- (see also logic synthesis)
- changing assignments
- list of
- making assignments
- resetting assignments
- specifying default

Display

Logic Options

The following table lists the logic options that are supported by the ACEX® 1K, APEX™ 20K, APEX 20KC, APEX 20KE, APEX II, ARM®-based Excalibur™, Cyclone™, Cyclone II, FLEX® 6000, FLEX 6000A, FLEX 10K®, FLEX 10KA, FLEX 10KE, HardCopy® Stratix®, MAX® II, MAX 3000, MAX 3000A, MAX 7000, MAX 7000AE, MAX 7000B, MAX 7000S, Mercury™, Stratix, Stratix II, and Stratix GX device families.

Logic Option	Device Family Support
Allow Any RAM Size For Recognition	ACEX 1K, APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, Cyclone, Cyclone II, FLEX 10K, FLEX 10KA, FLEX 10KE, Mercury, Stratix, Stratix GX, and Stratix II devices
Allow Any ROM Size For Recognition	ACEX 1K, APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, Cyclone, Cyclone II, FLEX 10K, FLEX 10KA, FLEX 10KE, Mercury, Stratix, Stratix GX, and Stratix II devices
Allow Any Shift Register Size For Recognition	APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, Cyclone, Cyclone II, Stratix, Stratix GX, and Stratix II devices
Allow XOR Gate Usage	APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, MAX 3000, MAX 7000A, MAX 7000AE, MAX 7000B, and MAX 7000S devices
Allows Synchronous Control Signal Usage in Normal Mode Logic Cells	Cyclone, Cyclone II, HardCopy Stratix, MAX II, Stratix, Stratix GX, and Stratix II devices
Auto Carry Chains	ACEX 1K, APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, Cyclone, Cyclone II, FLEX 10K, FLEX 10KA, FLEX 10KE, FLEX 6000, HardCopy Stratix, MAX II, Mercury, Stratix, Stratix GX, and Stratix II devices
Auto Cascade Chains	ACEX 1K, APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, FLEX 10K, FLEX 10KA, FLEX 10KE, and FLEX 6000 devices
Auto Clock Enable Replacement	ACEX 1K, APEX 20K, APEX 20KC, APEX 20KE, APEX II, ARM-based Excalibur, Cyclone, Cyclone II, FLEX 10K, FLEX 10KA, FLEX 10KE, Mercury, Stratix, Stratix GX, and Stratix II devices
Auto Delay Chains	Cyclone, Cyclone II, Stratix, Stratix GX, and Stratix II devices
Auto DSP Block	HardCopy Stratix, Stratix, Stratix GX, and Stratix II devices

Copyright © 2004 Altera Corporation

Quartus Settings File (QSF)

- Stores All Settings & Assignments
- Uses Tcl Syntax

pipemult.qsf*

```
29
30 # Project-Wide Assignments
31 # ========================
32 set_global_assignment -name ORIGINAL_QUARTUS_VERSION 4.0
33 set_global_assignment -name PROJECT_CREATION_TIME_DATE "11:30:40  JANUARY 19, 2004"
34 set_global_assignment -name LAST_QUARTUS_VERSION 4.0
35
36 # Pin & Location Assignments
37 # ==========================
38 set_location_assignment IOBANK_2 -to dataa
39 set_location_assignment IOBANK_2 -to datab
40
41 set_location_assignment PIN_L2 -to clk1
42 set_location_assignment PIN_F22 -to wren
43 set_location_assignment PIN_L8 -to ~DATA0~
44
45 # Timing Assignments
46 # ==================
47 set_global_assignment -name INCLUDE_EXTERNAL_PIN_DELAYS_IN_FMAX_CALCULATIONS OFF
48
49 # Analysis & Synthesis Assignments
50 # ================================
51 set_global_assignment -name FAMILY Stratix
52 set_global_assignment -name DEVICE_FILTER_SPEED_GRADE ANY
53 set_global_assignment -name TOP_LEVEL_ENTITY pipemult
54
55 # Fitter Assignments
56 # ==================
57 set_global_assignment -name PHYSICAL_SYNTHESIS_REGISTER_RETIMING ON
58 set_global_assignment -name DEVICE EP1S10F484C5
59 set_instance_assignment -name IO_STANDARD "2.5 V" -to dataa
60 set_instance_assignment -name IO_STANDARD "2.5 V" -to datab
61
62 # -----------------
```

Organized by Assignment Type

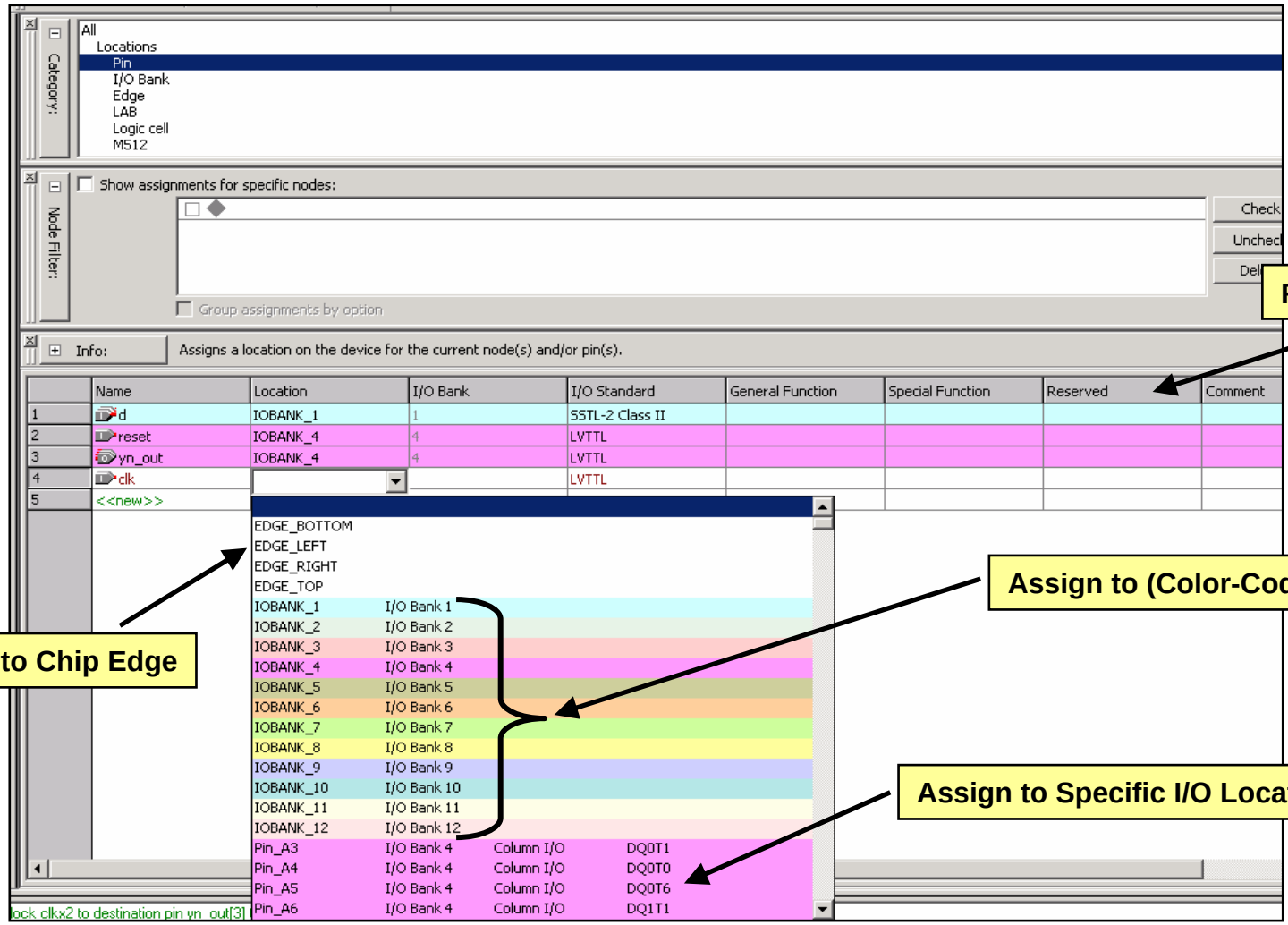
Copyright © 2004 Altera Corporation

I/O (Pin) Assignments

- Make I/O Assignments Quickly
  - Assignment Editor
  - Floorplan
  - QSF File
  - Import from Spreadsheet in CSV Format
  - Scripting
- Specific Pin Location Unnecessary
  - IO Bank & Chip Edge Location Assignments
  - Reserved Pins with or without Locations

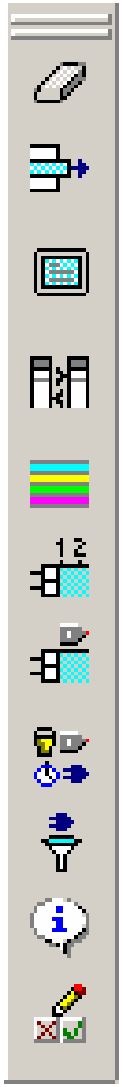
Copyright © 2004 Altera Corporation

Assignment Editor I/O Assignments



Copyright © 2004 Altera Corporation

AE Pin Assignment Features

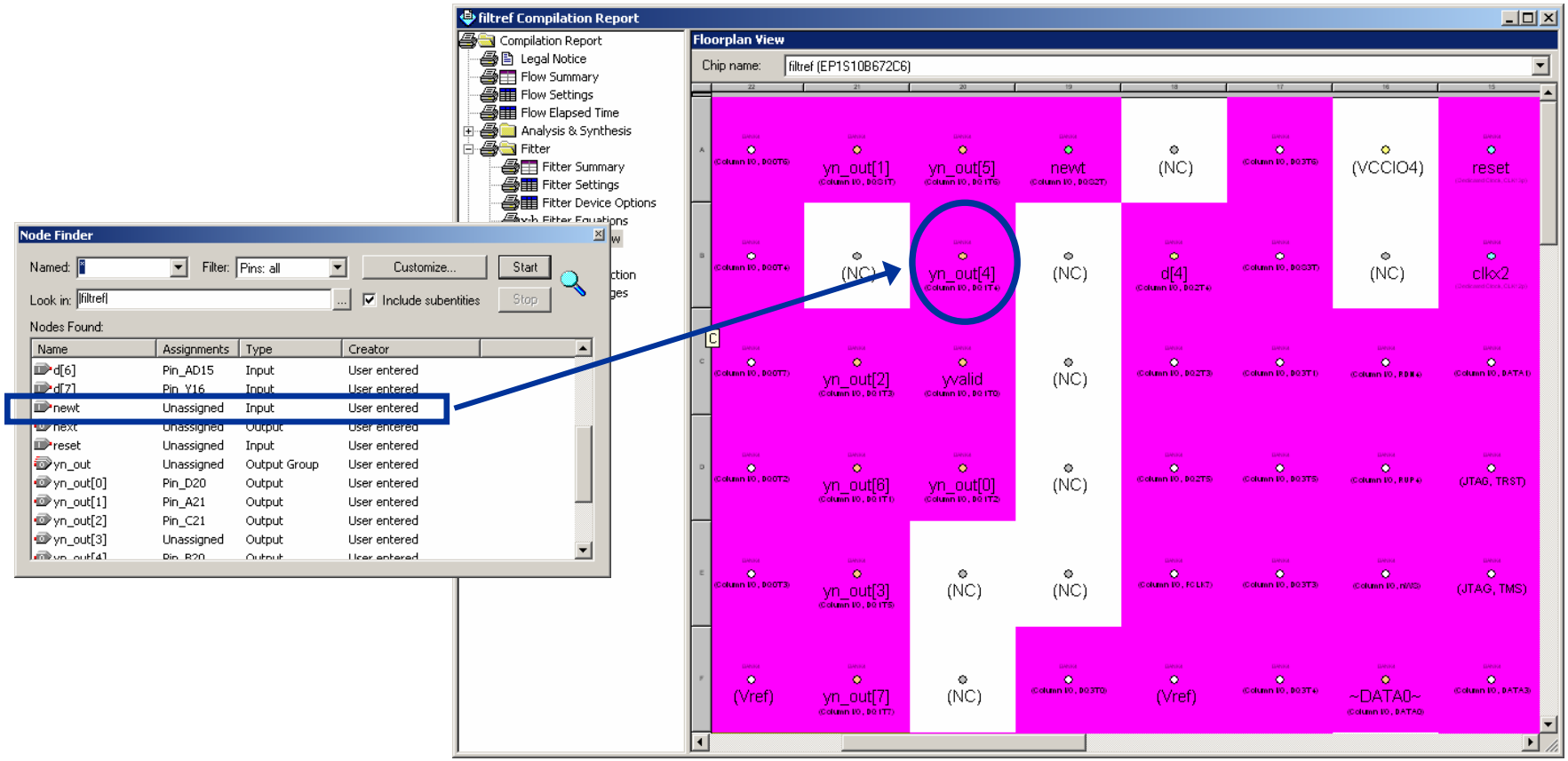


- ⑤Show All Known Pin Names
  - Populates Spreadsheet with List of All Pin Names in Design
    - ③Assigned & Not Assigned
- ⑤ Show All Assignable Pin Numbers
  - Populate Spreadsheet with All Pin Numbers Available for Assignment
- ⑤Show I/O Banks in Color
  - Enable/Disable I/O Color Coding of Spreadsheet Based upon Floorplan

Copyright © 2004 Altera Corporation

Drag & Drop I/O Assignments

⑤Drag & Drop from Node Finder to Floorplan



Copyright © 2004 Altera Corporation

Type I/O Assignments

⑤Type Pin Assignments Directly into QSF

⑤Use Tcl Syntax

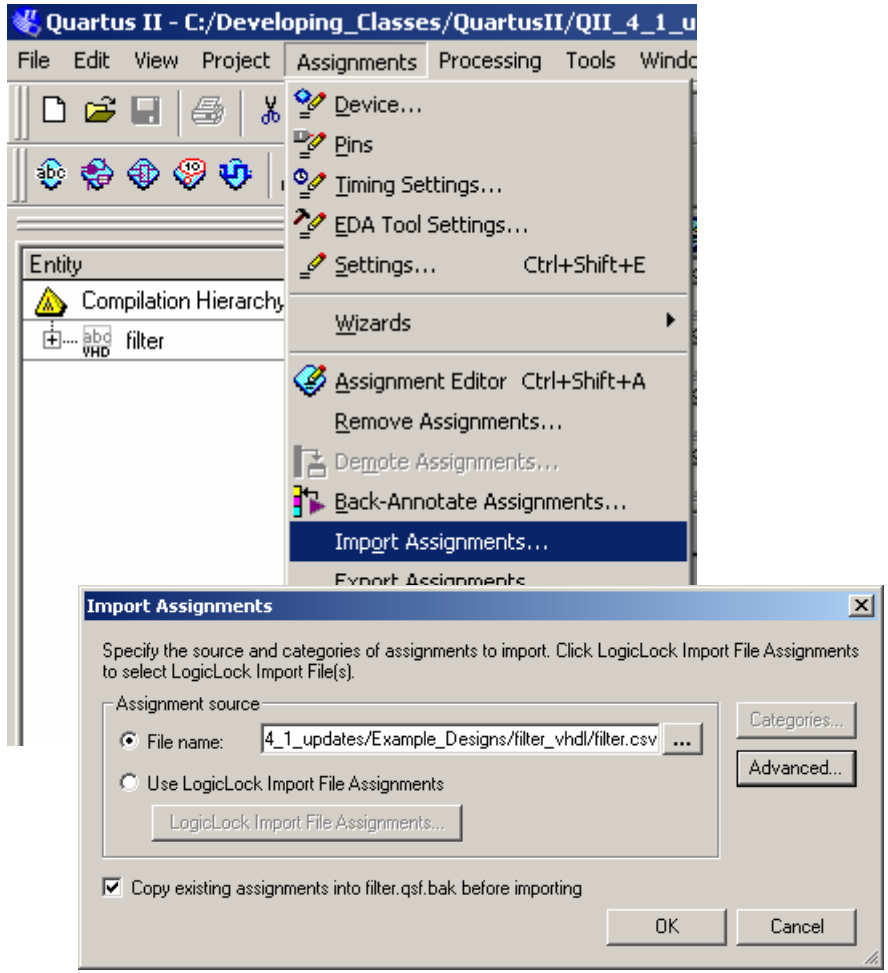
pipemult.qsf*

```
29
30 # Project-Wide Assignments
31 # ========================
32 set_global_assignment -name ORIGINAL_QUARTUS_VERSION 4.0
33 set_global_assignment -name PROJECT_CREATION_TIME_DATE "11:30:40  JANUARY 19, 2004"
34 set_global_assignment -name LAST_QUARTUS_VERSION 4.0
35
36 # Pin & Location Assignments
37 # ==========================
38 set_location_assignment IOBANK_2 -to dataa
39 set_location_assignment IOBANK_2 -to datab
40
41 set_location_assignment PIN_L2 -to clk1
42 set_location_assignment PIN_F22 -to wren
43 set_location_assignment PIN_L8 -to ~DATA0~
44
45 # Timing Assignments
46 # ==================
47 set_global_assignment -name INCLUDE_EXTERNAL_PIN_DELAYS_IN_FMAX_CALCULATIONS OFF
48
49 # Analysis & Synthesis Assignments
50 # ================================
51 set_global_assignment -name FAMILY Stratix
52 set_global_assignment -name DEVICE_FILTER_SPEED_GRADE ANY
53 set_global_assignment -name TOP_LEVEL_ENTITY pipemult
54
55 # Fitter Assignments
56 # ==================
57 set_global_assignment -name PHYSICAL_SYNTHESIS_REGISTER_RETIMING ON
58 set_global_assignment -name DEVICE EP1S10F484C5
59 set_instance_assignment -name IO_STANDARD "2.5 V" -to dataa
60 set_instance_assignment -name IO_STANDARD "2.5 V" -to datab
61
62 # -----------------
```

Copyright © 2004 Altera Corporation

Import CSV Assignments

- Use Spreadsheet to Enter Pin Assignments & I/O Standards
- Column Header Names
  - To
  - Assignment Name (Location)
  - Value (Pin Number)
  - I/O Standard
- Useful for Interfacing to PCB Tools



	A	B	C
1	To	Assignment Name	Value
2	d[7]	Location	PIN_J4
3	d[6]	Location	PIN_H4

Copyright © 2004 Altera Corporation

I/O Assignment Problems

- I/O Selected & Verified Too Far into Design Cycle
- Variety of Complex I/O Standards on Single Device
  - Not All Supported on Every I/O Block
- I/O Placement Limitations
  - Current Strength
  - Single-Ended vs. Differential
  - Single-Ended vs. V_{ref}

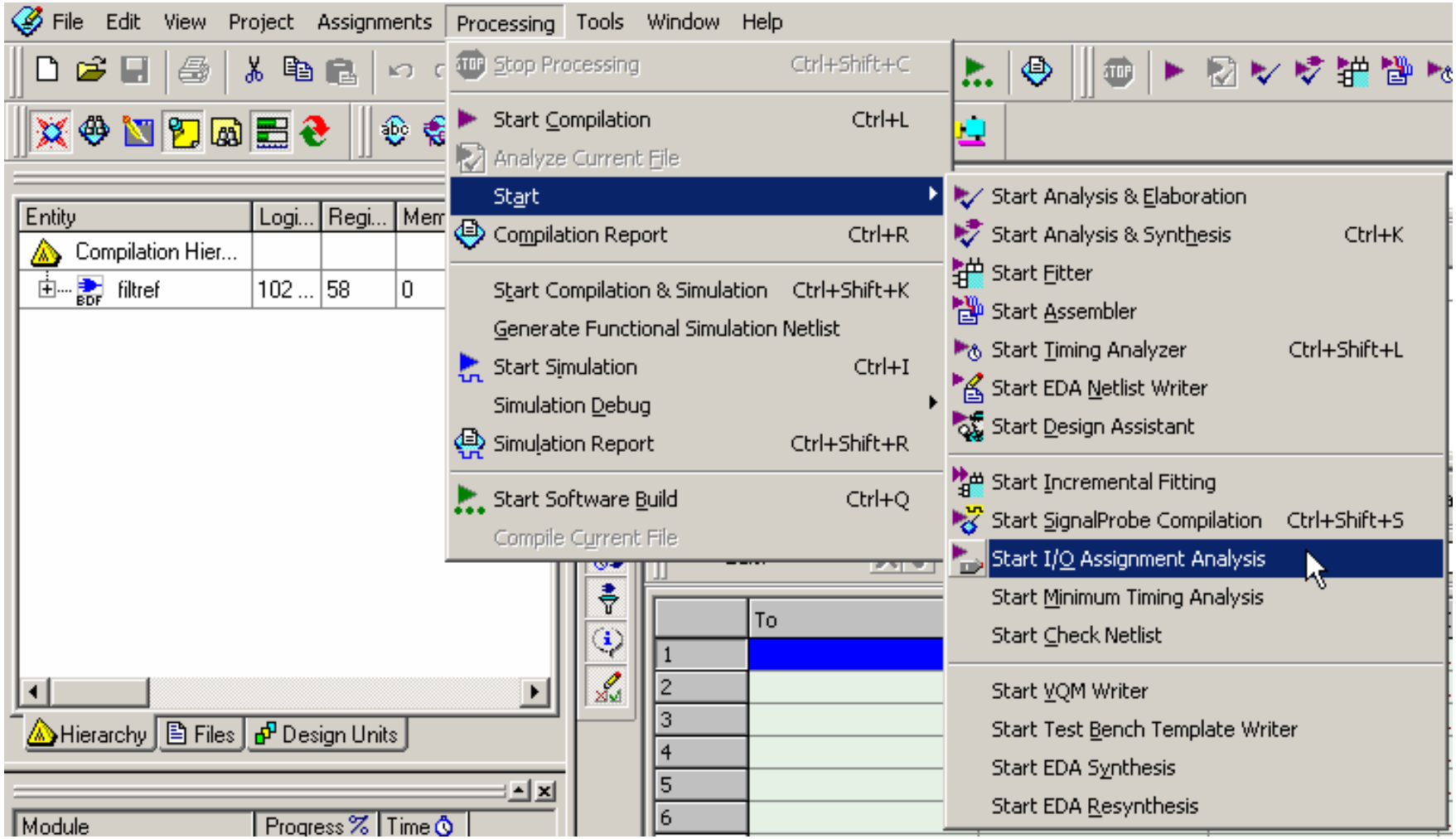
Copyright © 2004 Altera Corporation

I/O Assignment Analysis Command

- Quickly Checks Legality of I/O Assignments
  - Checks All Assignments
  - Does Not Stop at First Failure
- Allows User to Identify & Correct Pin-Related Issues without Full Compilation
- Has Two Usages
  - Performing Legality Checks Based on User Reserved Pin Assignments with Partial or No Design Files
  - Performing Legality Checks Based on User I/O Assignments with a Complete Design

Copyright © 2004 Altera Corporation

I/O Assignment Analysis



Copyright © 2004 Altera Corporation

I/O Analysis Requirements

- ⑤ I/O Declaration
  - HDL Port Declaration
  - Reserved Pin
- ⑤ Pin-Related Assignments
  - I/O Standard
  - Current Strength
  - Pin Location (Pin, Bank, Edge)
  - PCI Clamping Diode
  - Toggle Rate

Copyright © 2004 Altera Corporation

Legality Checks

Rule (subset of the rules)	Incomplete Design	Complete Design
Differential Pin Spacing		●
# of Output/Bidirs Toggling with a VREF		●
Blocks Directly Feeding or Fed by I/O pins		●
Current Strength	●	●
DC Current Draw	●	●
Pin Location	●	●
I/O Bank Capacity	●	●
Voltage Conflict (Vref, VCCIO)	●	●
On-Chip Termination	●	●

Copyright © 2004 Altera Corporation

I/O Assignment Analysis Output

- ⑤Detailed Messages on
  - I/O Assignment Issues
  - Compiler Assumptions
  - Device & Pin Migration Issues
  - I/O Bank Voltages & Standards
- ⑤Compilation Report (Fitter Section)
  - I/O Pin Tables
    - ③Show User & Fitter Assigned I/O
  - Shows Partial Placement Results in Floorplan to Help Debug Fitting Errors

Copyright © 2004 Altera Corporation

I/O Assignment Analysis Messages

Fitter Messages

```
Info: ************************************************************
Info: Running Quartus II Fitter
Info: Command: quartus_fit --import_settings_files=on --export_settings_files=off compile_fir_filter -c filtref --check_ios
Info: Selected device EP1S10B672C6 for design filtref
Info: Selected migration device list is legal with 632 total of migratable pins
Info: Selected device migration path can't use 6 pins as regular I/Os
    Info: Pin AD21
    Info: Pin D7
    Info: Pin G16
    Info: Pin R18
    Info: Pin W11
    Info: Pin Y7
Info: Selected device migration path can't use 5 pins as DQ I/Os
Info: Completed User Assigned Global Signals Promotion Operation
Info: Automatically promoted signal clk to use Global clock in Pin R26
Info: Automatically promoted signal clkx2 to use Global clock in Pin B15
Info: Automatically promoted signal reset to use Global clock in Pin A15
Info: Completed Auto Global Promotion Operation
Info: Statistics of I/O pins that use the same VCCIO and VREF
    Info: There are 8 I/O pins (VREF = unused, VCCIO = 3.30, 2 input, 6 output, 0 bidirectional)
        Info: Used I/O standards LVTTL.
Info: I/O banks and pin(s) statistics before I/O pin placement
Info: I/O banks and pin(s) statistics after I/O pin placement
    Info: Statistics of I/O banks
        Info: I/O bank 1: VREF = unused, VCCIO = unused, used pin 1, available pins 38.
        Info: I/O bank 2: VREF = unused, VCCIO = unused, used pin 1, available pins 38.
        Info: I/O bank 3: VREF = unused, VCCIO = unused, used pin 0, available pins 43.
        Info: I/O bank 4: VREF = unused, VCCIO = 2.50, used pin 7, available pins 38.
        Info: I/O bank 5: VREF = unused, VCCIO = unused, used pin 0, available pins 39.
        Info: I/O bank 6: VREF = unused, VCCIO = unused, used pin 1, available pins 38.
        Info: I/O bank 7: VREF = unused, VCCIO = unused, used pin 3, available pins 42.
        Info: I/O bank 8: VREF = unused, VCCIO = unused, used pin 2, available pins 42.
        Info: I/O bank 9: VREF = unused, VCCIO = 3.30, used pin 6, available pins 0.
        Info: I/O bank 10: VREF = unused, VCCIO = unused, used pin 0, available pins 0.
        Info: I/O bank 11: VREF = unused, VCCIO = unused, used pin 2, available pins 4.
        Info: I/O bank 12: VREF = unused, VCCIO = unused, used pin 0, available pins 0.
Info: Completed I/O Pin Placement Operation
Info: Quartus II Fitter was successful. 0 errors, 0 warnings
Info: Writing report file filtref.fit.rpt
```

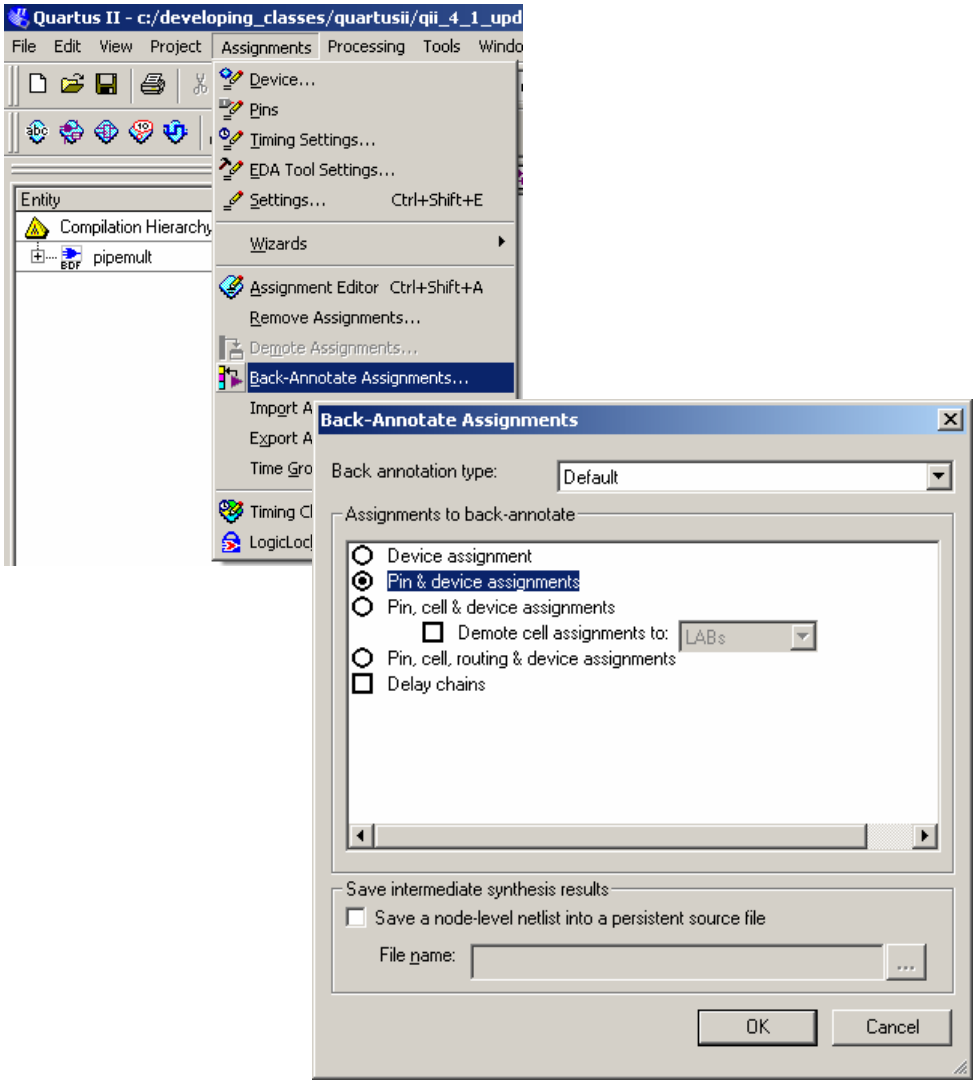
Migration I/O

I/O Standard Statistics

I/O Banks & Pin Usage

Copyright © 2004 Altera Corporation

Back-Annotation



- Copies Device & Resource Assignments Made by Compiler into QSF File
  - Pins
  - Logic
  - Routing
- "Locks Down" Locations in Floorplan

Copyright © 2004 Altera Corporation

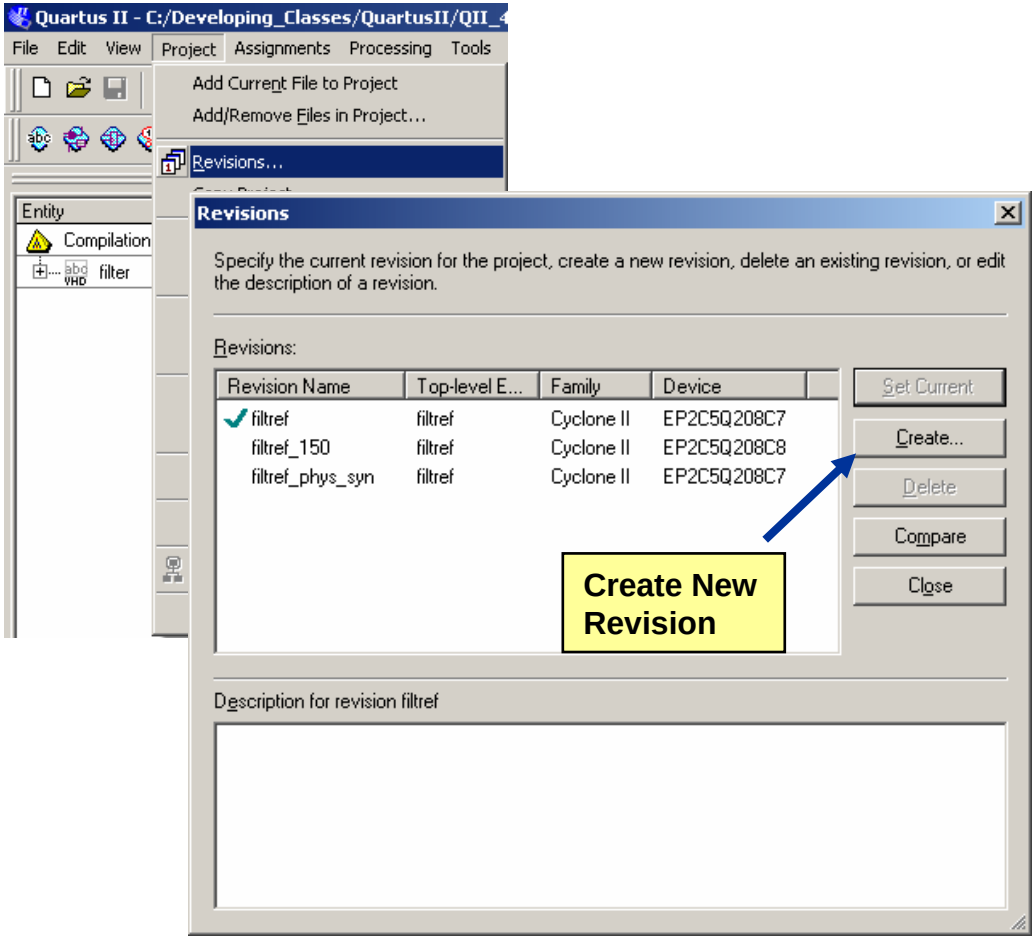
Design File Management

- ⑤Project Archive & Restore
  - Stores All Project Files
    - ③Design Files
    - ③Settings File
    - ③Output Files
- ⑤Revisions
  - Stores Only QSF
  - Allows Designer to Try Different Options
  - Allows Comparison of Revisions

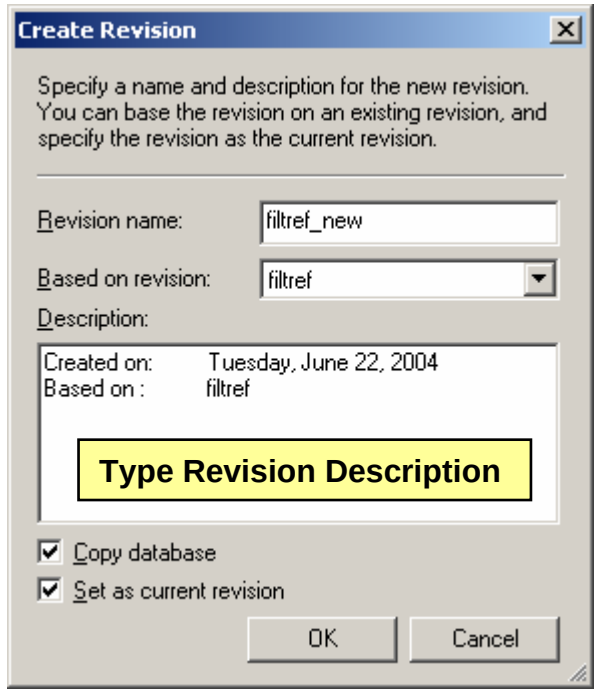
Copyright © 2004 Altera Corporation

Creating a Revision

⑤Project ⇒ Revisions



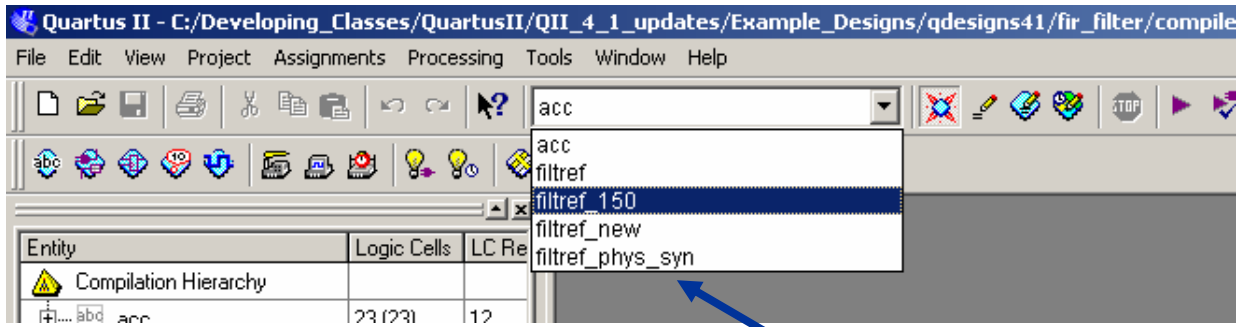
Base Revision on Any Previous Revision



Copyright © 2004 Altera Corporation

Project Revision Support

- Active Revision Names Stored in QPF
- QSF Created for Each Revision
  - <Revision_name>.QSF
- Text File Created for Each Revision
  - <Revision_Name>_description.TXT



Easily Switch between Revisions

Copyright © 2004 Altera Corporation

Compare Revisions

- ⑤ Detailed Summary of Results
  - Synthesis
    - ③ No Timing Data
  - Fitting
    - ③ Area
    - ③ Timing
- ⑤ Compare Results with Other Projects

Compare Revisions

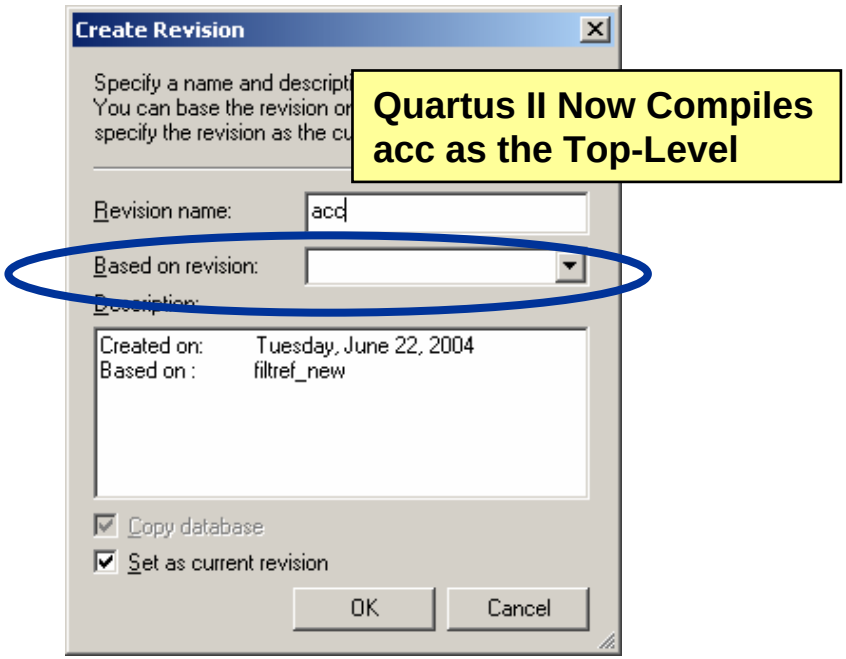
	C:/qdesigns41/tutorial/fir_filter.qpf Revison filtref	C:/qdesigns41/tutorial/fir_filter.qpf Revison PhysicalSynthesis	C:/qdesigns41/tutorial/fir_filter.qpf Revison MyStratixII
Analysis & Synthesis			
Fitter			
Flow Status	Successful - Thu Apr 29 15:39:1...	Successful - Thu Apr 29 16:21:4...	Successful - Wed Apr 28 16:07:...
Quartus II Version	4.1 Internal Build 144 4/25/2004...	4.1 Internal Build 144 4/25/2004...	4.1 Internal Build 144 4/25/2004...
Revision Name	filtref	PhysicalSynthesis	MyStratixII
Top-level Entity Name	filtref	filtref	filtref
Family	Stratix	Stratix	Stratix II
Device	EP1S10B672C7	EP1S10B672C7	EP2S15F484C4
Timing Models	Production	Production	Preliminary
Total logic elements	101 / 10,570 (< 1 %)	101 / 10,570 (< 1 %)	
Total pins	22 / 346 (6 %)	22 / 346 (6 %)	22 / 343 (6 %)
Total memory bits	0 / 920,448 (0 %)	0 / 920,448 (0 %)	0 / 419,328 (0 %)
DSP block 9-bit elements	0 / 48 (0 %)	0 / 48 (0 %)	0 / 96 (0 %)
Total PLLs	0 / 6 (0 %)	0 / 6 (0 %)	0 / 6 (0 %)
Total DLLs	0 / 2 (0 %)	0 / 2 (0 %)	0 / 2 (0 %)
Total ALUTs			92 / 12,480 (< 1 %)
Total registers			57
Timing Analyzer			
Worst-case tsu			
Worst-case tco			
Slack	N/A	N/A	N/A
Required Time	None	None	None
Actual Time	9.374 ns	9.374 ns	6.285 ns
From	state_m:inst1|filter~27	state_m:inst1|filter~27	inst5[1]
To	follow	follow	yn_out[1]
From Clock	clk	clk	clkx2
To Clock			
Failed Paths	0	0	0
Worst-case th			
Worst-case Minimum tco			
Clock Setup: 'clkx2'			
Slack	-2.042 ns	-2.042 ns	-0.853 ns
Required Time	200.00 MHz (period = 5.000 ns)	200.00 MHz (period = 5.000 ns)	200.00 MHz (period = 5.000 ns)
Actual Time	N/A	N/A	N/A
From	inst4	inst4	inst4
To	inst5[7]	inst5[7]	inst5[7]
From Clock	clk	clk	clk
To Clock	clkx2	clkx2	clkx2
Failed Paths	16	16	16
Clock Setup: 'clk'			
Slack	0.472 ns	0.472 ns	3.216 ns
Required Time	100.00 MHz (period = 10.000 ns)	100.00 MHz (period = 10.000 ns)	100.00 MHz (period = 10.000 ns)
Actual Time	104.95 MHz (period = 9.528 ns)	104.95 MHz (period = 9.528 ns)	147.41 MHz (period = 6.784 ns)
From	state_m:inst1|filter~30	state_m:inst1|filter~30	state_m:inst1|filter~29
To	acc:inst3|result[11]	acc:inst3|result[11]	acc:inst3|result[11]
From Clock	clk	clk	clk
To Clock	clk	clk	clk
Failed Paths	0	0	0
Total number of failed paths			
Slack			
Required Time			
Actual Time			
From			
To			
From Clock			
To Clock			
Failed Paths	16	16	16

Compare to other project... OK

Copyright © 2004 Altera Corporation

Compiling a Submodule

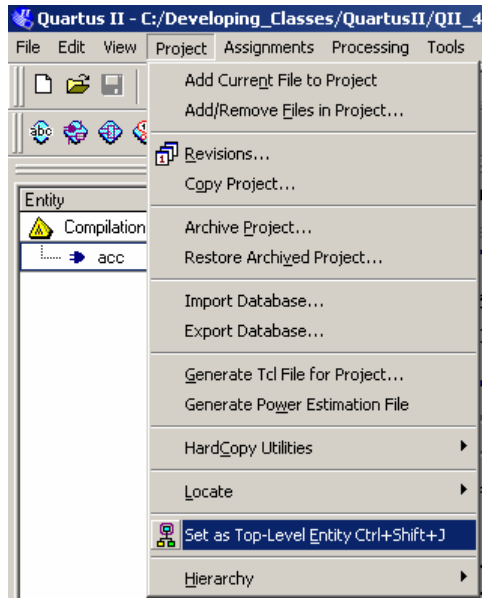
1. Create a New Revision
2. Leave “Based on revision” blank



Quartus II Now Compiles acc as the Top-Level

Creates New .QSF File as a Revision

1. Open Submodule File
2. Select “Set as Top-Level Entity” from Project Menu



Reuses Current .QSF File

Copyright © 2004 Altera Corporation

Compilation Entry Summary

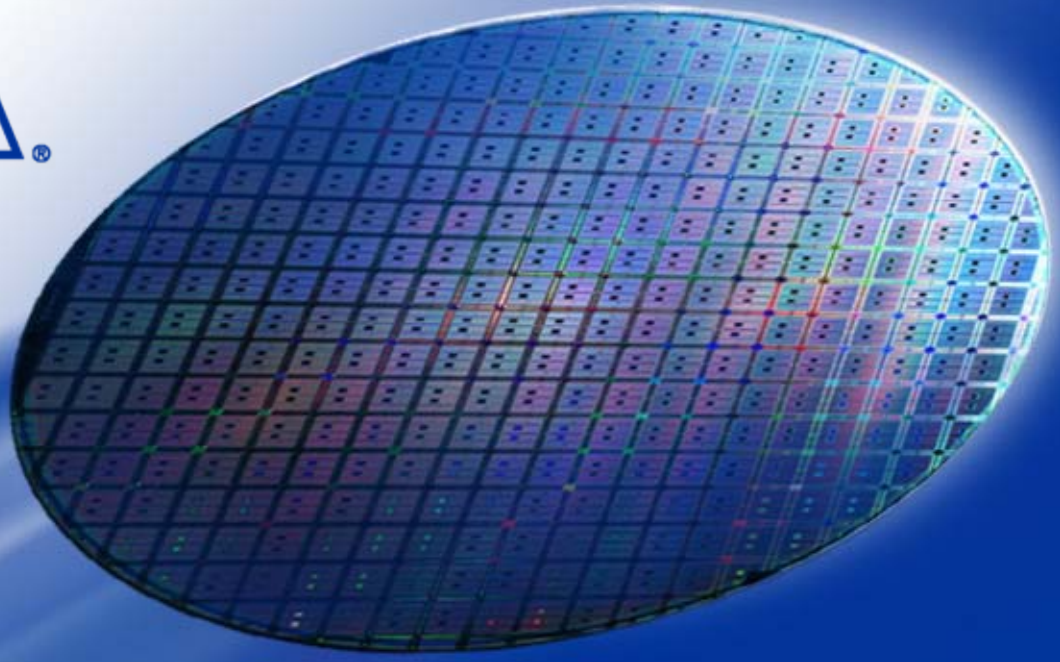
- Compilation Includes Synthesis & Fitting
- Compilation Report Contains Detailed Results
- Settings & Assignments Control Compilation
- Pin Assignments Can Be Performed in Many Ways
- Revisions Store Settings, Assignments & Compilation Results for Comparison

Copyright © 2004 Altera Corporation

Summary

- Synthesis & Fitter Options
- Assignment Editor
- Compilation Reports
- Floorplan

Copyright © 2004 Altera Corporation



Quartus II Timing Analysis

Copyright © 2004 Altera Corporation

Features

- Built-In Static Timing Analysis
- Single & Multi-Clock Timing Analysis
- Standard & Minimum Timing Analysis
- Timing Export for Analysis in Another EDA Tool

Copyright © 2004 Altera Corporation

Timing Analysis Agenda

- Single Clock Analysis
- Timing Assignments
  - Global & Individual
- Multi-Clock Analysis
- Multi-Cycle Analysis
- Minimum Timing

Copyright © 2004 Altera Corporation

Running Timing Analysis

⑤Automatically

- Use Full Compilation

⑤Manually

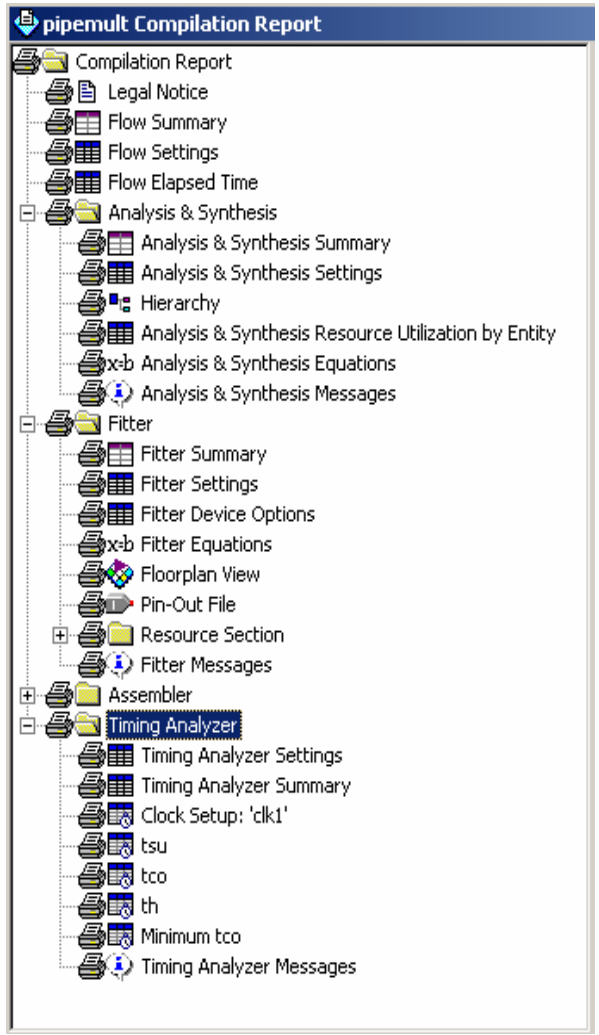
- Processing Menu ⇒Start ⇒Start Timing Analysis
- Tcl Scripts
- Uses
  - ③Changing Speed Grade
  - ③Annotating Netlist with Delay Information

Copyright © 2004 Altera Corporation

Reporting Timing Results

⑤ Timing Analyzer Section of Compilation Report

- Summary
  Timing Analyses
- Clock Setup (fmax)
- Clock Hold
- tsu (Input Setup Times)
- th (Input Hold Times)
- tco (Clock to Out Delays)
- tpd (Pin to Pin Delays)
- Minimum tpd & tco

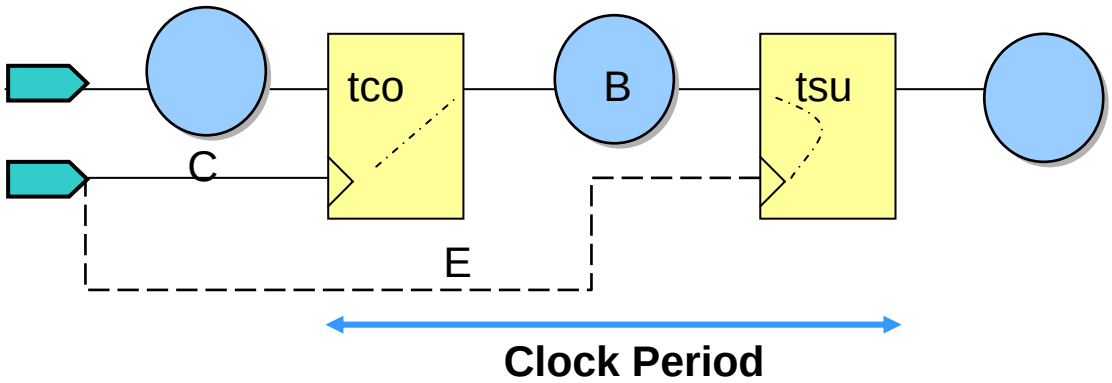


Copyright © 2004 Altera Corporation

Clock Setup (fmax)

⑤Worst-Case Clock Frequency

– Without Violating Internal Setup & Hold Times



Clock Period	**= Clock-to-Out + Data Delay + Setup Time - Clock Skew**
	= tco + B + tsu - (E - C)
fmax	**= 1/Clock Period**

Copyright © 2004 Altera Corporation

Clock Setup (fmax) Tables

Fmax Values Are Listed in Ascending Order; Worst Fmax Is Listed on the Top

Worst fmax

two_d_dct Compilation Report

- Compilation Report
  - Legal Notice
  - Flow Summary
  - Flow Settings
  - Flow Elapsed Time
  - Flow Log
  - Analysis & Synthesis
  - Fitter
  - Assembler
  - Timing Analyzer
    - Timing Analyzer Settings
    - Timing Analyzer Summary
    - Clock Settings Summary
    - Clock Setup: 'clk'
    - tsu
    - tco
    - th
    - inimum tco
    - ming Analyzer Messages

Clock Setup: 'clk'

	Slack	Actual fmax (period)	From	To	From Clock	To Clock	Required Se
1	N/A	231.11 MHz (period = 4.327 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
2	N/A	231.96 MHz (period = 4.311 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
3	N/A	233.05 MHz (period = 4.291 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
4	N/A	235.02 MHz (period = 4.255 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
5	N/A	237.02 MHz (period = 4.219 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
6	N/A	238.72 MHz (period = 4.189 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
7	N/A	238.95 MHz (period = 4.185 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
8	N/A	239.06 MHz (period = 4.183 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|			
9	N/A	239.23 MHz (period = 4.180 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|			
10	N/A	239.52 MHz (period = 4.175 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|			
11	N/A	239.64 MHz (period = 4.173 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|			
12	N/A	239.87 MHz (period = 4.169 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|			
13	N/A	240.44 MHz (period = 4.159 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
14	N/A	240.79 MHz (period = 4.153 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
15	N/A	241.02 MHz (period = 4.149 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
16	N/A	241.08 MHz (period = 4.148 ns)	data_valid_state~22	column_dct:col_dct_in...	clk	clk	None
17	N/A	241.14 MHz (period = 4.147 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
18	N/A	241.31 MHz (period = 4.144 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
19	N/A	241.60 MHz (period = 4.139 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
20	N/A	242.90 MHz (period = 4.117 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
21	N/A	243.13 MHz (period = 4.113 ns)	transpose_matrix:tp_...	row_dct:row_dct_inst|...	clk	clk	None
22	N/A	243.25 MHz (period = 4.111 ns)	column_dct:col_dct...	column_dct:col_dct_in...	clk	clk	None

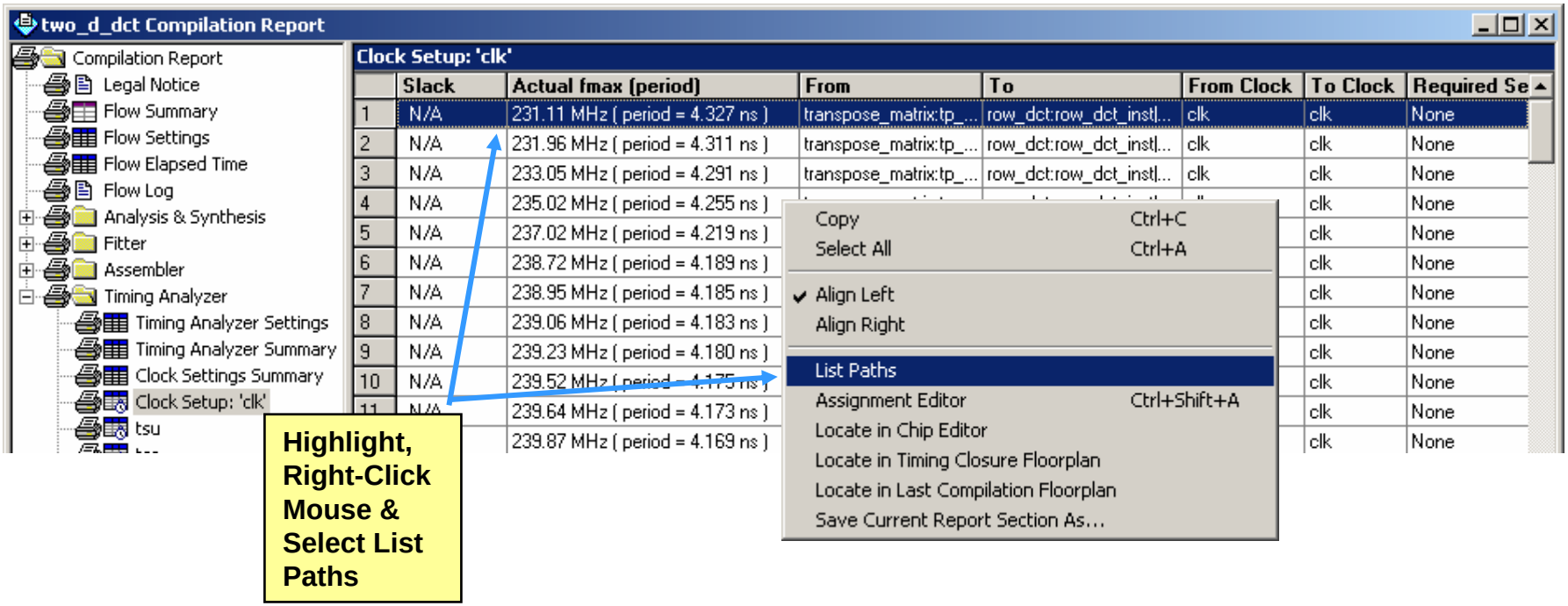
Source, Destination Registers & Associated Fmax Values

Select Clock Setup

Copyright © 2004 Altera Corporation

fmax Analysis

⑤To Analyze the Path More Closely



Similar Steps for All Timing Path Analysis in Quartus II

Copyright © 2004 Altera Corporation

fmax Analysis Details

Data Delay (B)

Destination Register Clock Delay (E)

Source Register Clock Delay (C)

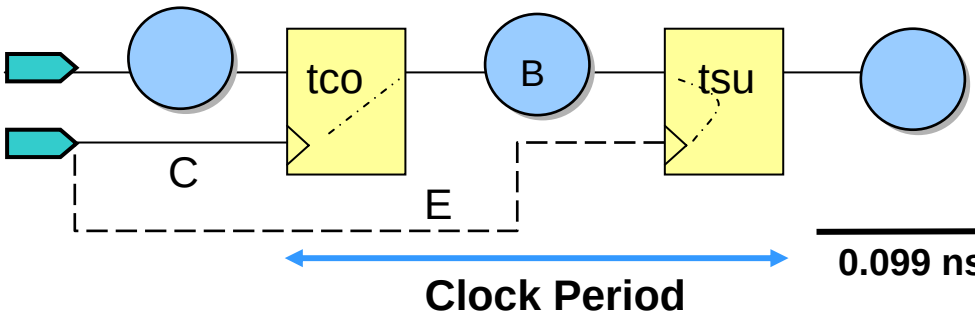
Clock to Output (tco)

Messages Window (System Tab) in Quartus II

Info: Clock clk has Internal fmax of 231.11 MHz between source register transpose_matrix:tp_mtx_inst|lpm_counte
Info: + Longest register to register delay is 4.135 ns
Info: - Smallest clock skew is 0.000 ns
Info: + Shortest clock path from clock clk to destination register is 2.548 ns
Info: - Longest clock path from clock clk to source register is 2.548 ns
Info: + Micro clock to output delay of source is 0.099 ns
Info: + Micro setup delay of destination is 0.093 ns

Setup Time (tsu)

Processing | System



$$\frac{1}{0.099\text{ ns} + 4.135\text{ ns} + 0.093\text{ ns} - 0.000\text{ ns}} = 231.11\text{ MHz}$$

Copyright © 2004 Altera Corporation

fmax Analysis Details (cont.)

Info: + Longest register to register delay is 4.135 ns
Info: 1: + IC(0.000 ns) + CELL(0.000 ns) = 0.000 ns; Loc. = LCFF_X17_Y11_N23; Fanout = 4; REG Node = 'transpose_matrix:tp_mtx_inst|lpm_
Info: 2: + IC(0.278 ns) + CELL(0.348 ns) = 0.626 ns; Loc. = LCCOMB_X17_Y11_N8; Fanout = 2; COMB Node = 'transpose_matrix:tp_mtx_inst|
Info: 3: + IC(0.336 ns) + CELL(0.160 ns) = [illegible] N18; Fanout = 3; COMB Node = 'transpose_matrix:tp_mtx_inst
Info: 4: + IC(0.255 ns) + CELL(0.225 ns) = [illegible] N10; Fanout = 2; COMB Node = 'transpose_matrix:tp_mtx_inst
Info: 5: + IC(0.225 ns) + CELL(0.055 ns) = [illegible] N26; Fanout = 85; COMB Node = 'transpose_matrix:tp_mtx_in
Info: 6: + IC(0.400 ns) + CELL(0.548 ns) = 2.830 ns; Loc. = LCCOMB_X19_Y11_N0; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 7: + IC(0.000 ns) + CELL(0.036 ns) = 2.866 ns; Loc. = LCCOMB_X19_Y11_N2; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 8: + IC(0.000 ns) + CELL(0.036 ns) = 2.902 ns; Loc. = LCCOMB_X19_Y11_N4; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 9: + IC(0.000 ns) + CELL(0.036 ns) = 2.938 ns; Loc. = LCCOMB_X19_Y11_N6; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 10: + IC(0.000 ns) + CELL(0.036 ns) = 2.974 ns; Loc. = LCCOMB_X19_Y11_N8; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 11: + IC(0.000 ns) + CELL(0.036 ns) = 3.010 ns; Loc. = LCCOMB_X19_Y11_N10; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u
Info: 12: + IC(0.000 ns) + CELL(0.036 ns) = 3.046 ns; Loc. = LCCOMB_X19_Y11_N12; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u
Info: 13: + IC(0.000 ns) + CELL(0.216 ns) = 3.262 ns; Loc. = LCCOMB_X19_Y11_N14; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u
Info: 14: + IC(0.000 ns) + CELL(0.036 ns) = 3.298 ns; Loc. = LCCOMB_X19_Y10_N0; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 15: + IC(0.000 ns) + CELL(0.036 ns) = 3.334 ns; Loc. = LCCOMB_X19_Y10_N2; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 16: + IC(0.000 ns) + CELL(0.036 ns) = 3.370 ns; Loc. = LCCOMB_X19_Y10_N4; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 17: + IC(0.000 ns) + CELL(0.036 ns) = 3.406 ns; Loc. = LCCOMB_X19_Y10_N6; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 18: + IC(0.000 ns) + CELL(0.036 ns) = 3.442 ns; Loc. = [illegible] Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 19: + IC(0.000 ns) + [illegible] Fanout = 2; COMB [illegible]
Info: 20: + IC(0.000 ns) + [illegible] Fanout = 2; COMB [illegible]
Info: 21: + IC(0.000 ns) + CELL(0.216 ns) = 3.730 ns; Loc. = [illegible] Fanout = 2; COMB [illegible]
Info: 22: + IC(0.000 ns) + CELL(0.036 ns) = 3.766 ns; Loc. = LCCOMB_X19_Y9_N0; Fanout = 2; COMB No[illegible]
Info: 23: + IC(0.000 ns) + CELL(0.036 ns) = 3.802 ns; Loc. = LCCOMB_X19_Y9_N2; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 24: + IC(0.000 ns) + CELL(0.036 ns) = 3.838 ns; Loc. = LCCOMB_X19_Y9_N4; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 25: + IC(0.000 ns) + CELL(0.036 ns) = 3.874 ns; Loc. = LCCOMB_X19_Y9_N6; Fanout = 2; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 26: + IC(0.000 ns) + CELL(0.036 ns) = 3.910 ns; Loc. = LCCOMB_X19_Y9_N8; Fanout = 1; COMB Node = 'row_dct:row_dct_inst|dct:u1|a
Info: 27: + IC(0.000 ns) + CELL(0.126 ns) = 4.036 ns; Loc. = LCCOMB_X19_Y9_N10; Fanout = 1; COMB Node = 'row_dct:row_dct_inst|dct:u1
Info: 28: + IC(0.000 ns) + CELL(0.099 ns) = 4.135 ns; Loc. = LCFF_X19_Y9_N11; Fanout = 1; REG Node = 'row_dct:row_dct_inst|dct:u1|adder
Info: Total cell delay = 2.641 ns (63.87 %)
Info: Total interconnect delay = 1.494 ns (36.13 %)
Info: - Smallest clock skew is 0.000 ns
Info: + Shortest clock path from clock clk to destination register is 2.548 ns
Info: 1: + IC(0.000 ns) + CELL(0.937 ns) = 0.937 ns; Loc. = IOC_X0_Y10_N1; Fanout = 1; CLK Node = 'clk'
Info: 2: + IC(0.123 ns) + CELL(0.000 ns) = 1.060 ns; Loc. = CLKCTRL_X0_Y11_N2; Fanout = 3115; COMB Node = 'clk~clkctrl'
Info: 3: + IC(0.857 ns) + CELL(0.631 ns) = 2.548 ns; Loc. = LCFF_X19_Y9_N11; Fanout = 1; REG Node = 'row_dct:row_dct_inst|dct:u1|adc
Info: Total cell delay = 1.568 ns (61.54 %)
Info: Total interconnect delay = 0.980 ns (38.46 %)
Info: - Longest clock path from clock clk to source register is 2.548 ns
Info: + Micro clock to output delay of source is 0.099 ns
Info: + Micro setup delay of destination is 0.093 ns

Node Fan-Out

Interconnect Delay

Cell Delay

Running Total

Destination Cell Name & Type

This signal path consists of 28 locations

% Interconnect vs. % Cell Delay

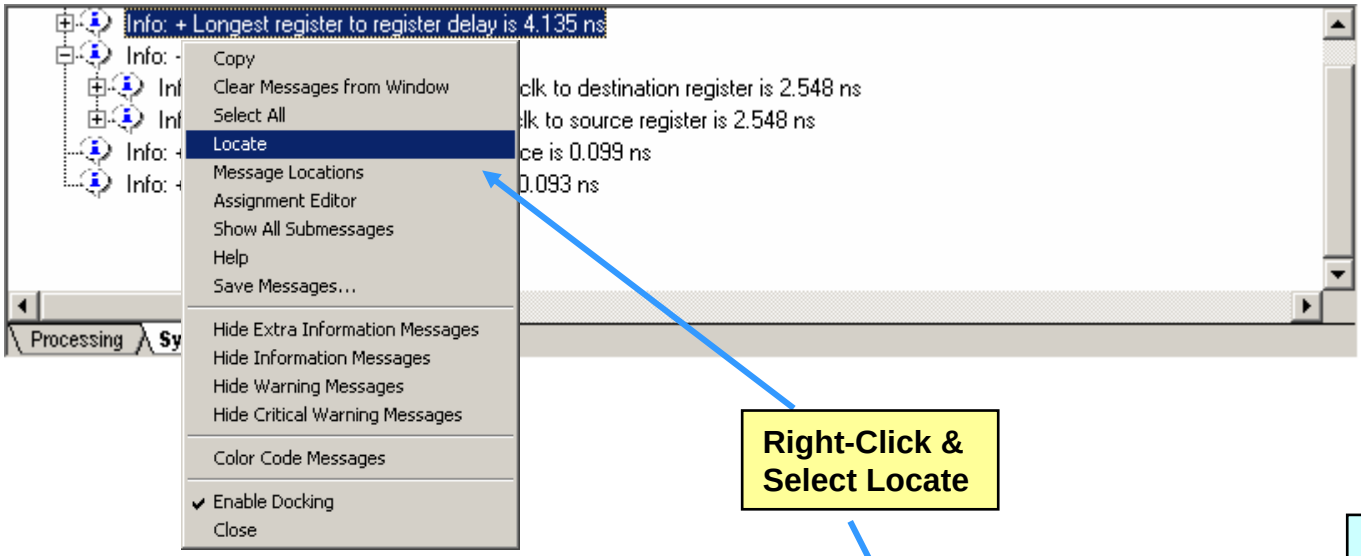
Notes:

1) **This Timing Convention Is Similar for All Timing Path Analysis**
2) **Carry Chain Delay Is Included in Cell Delay Number**

Copyright © 2004 Altera Corporation

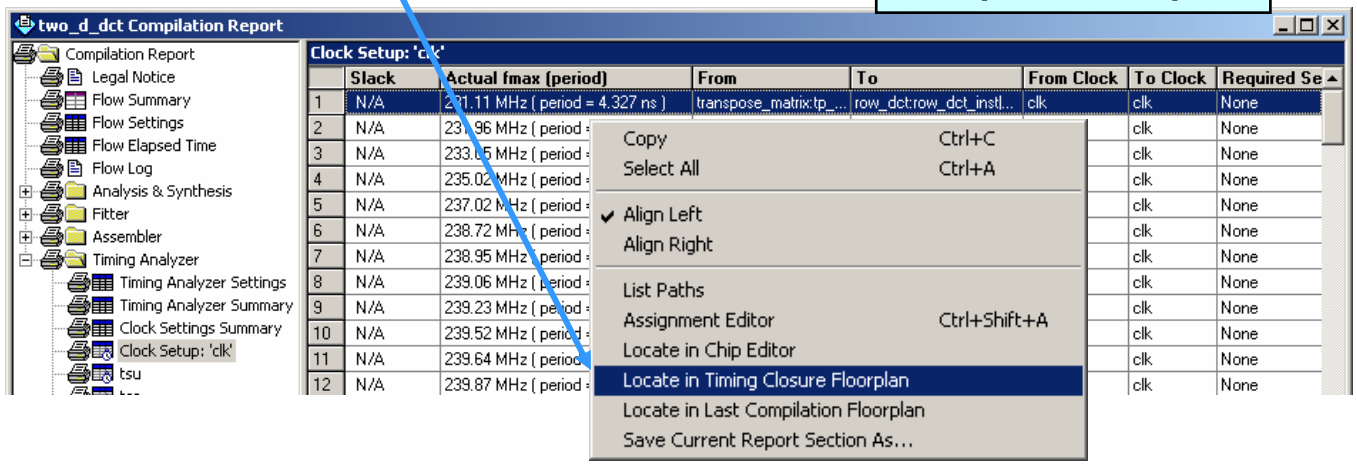
Locate Delay Path in Floorplan

Message Window



Right-Click & Select Locate

Compilation Report

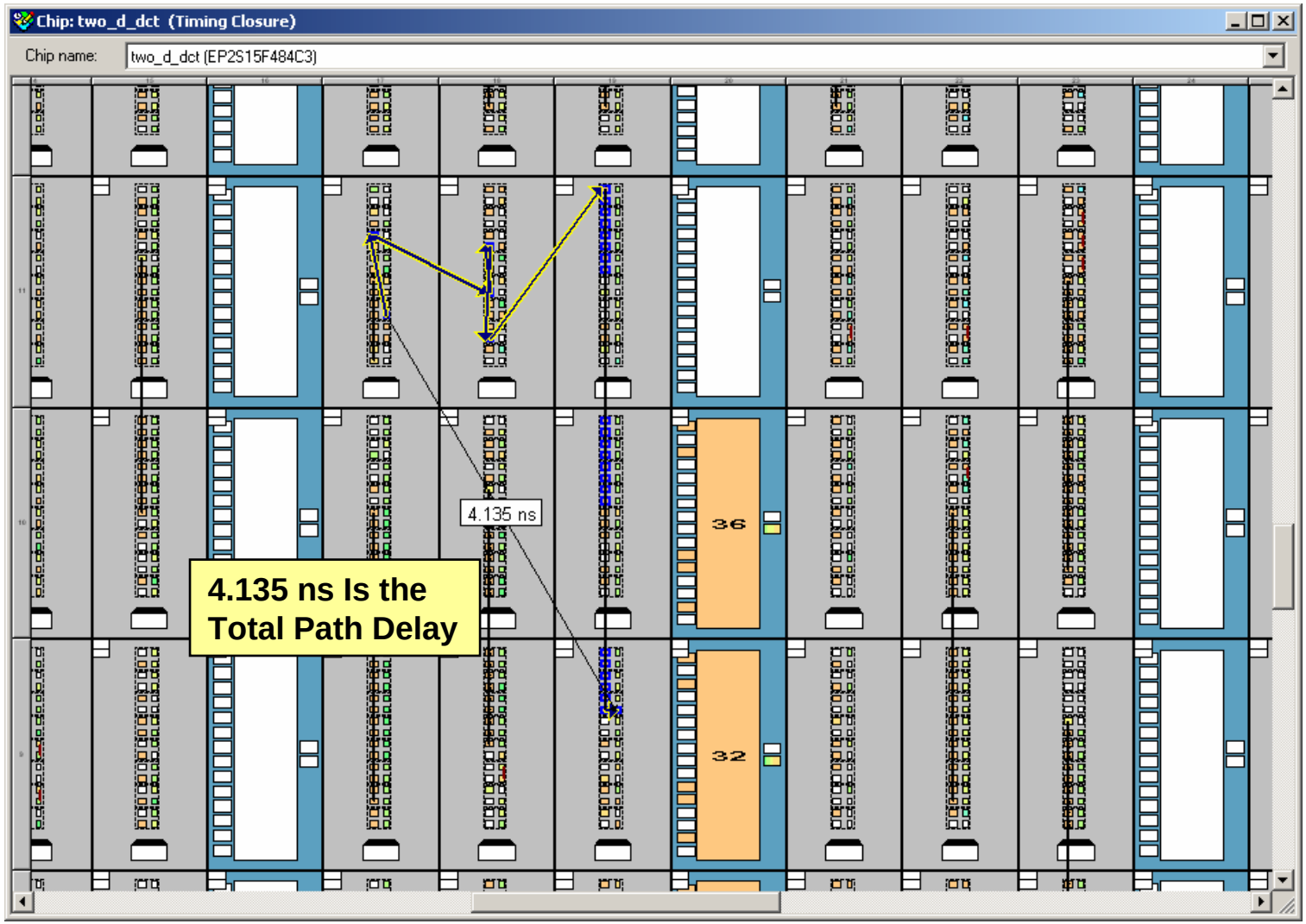


Notes:

1) **Use Similar Procedure for All Timing Path Analysis**

Copyright © 2004 Altera Corporation

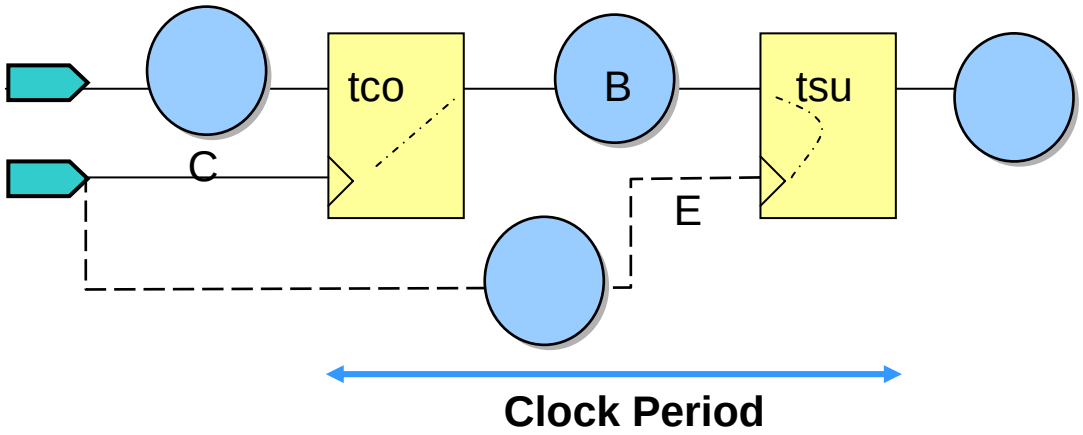
Locate Delay Path in Floorplan

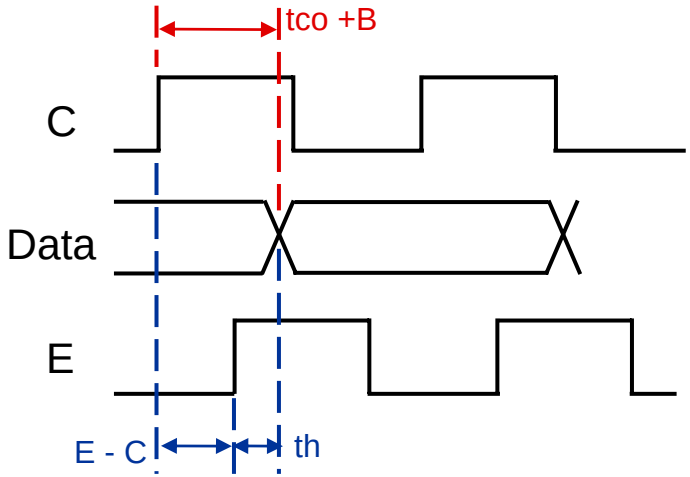


Copyright © 2004 Altera Corporation

Clock Hold Analysis

- Checks Internal Register-Register Timing
  - Report Occurs Only When Hold Violations Occur
- Results Usually When Data Delay (B) is Greater than Clock Skew (E-C)
  - Non-Global Clock Routing
  - Gated Clocks





Copyright © 2004 Altera Corporation

Hold Time Violations Table

filter Compilation Report

- Compilation Report
  - Legal Notice
  - Flow Summary
  - Flow Settings
  - Flow Elapsed Time
  - Flow Log
  - Analysis & Synthesis
  - Fitter
  - Assembler
  - Timing Analyzer
    - Timing Analyzer Settings
    - Timing Analyzer Summary
    - Clock Settings Summary
    - Clock Setup: 'clk'
    - Clock Hold: 'clk'
    - tsu
    - tco
    - th
    - Minimum tco
    - Timing Analyzer Messages

Clock Hold: 'clk'

	Minimum Slack	From	To	From Clock	To Clock	Required Hold
1	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[0]	clk	clk	None
2	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[1]	clk	clk	None
3	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[4]	clk	clk	None
4	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[2]	clk	clk	None
5	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[7]	clk	clk	None
6	Not operational: Clock Skew > Data Delay	mult_mega:u4|lpm_mult:lpm_mult_co...	acc:u5|y_tmp[3]	clk	clk	None

Discover Internal Hold Time Issues before Simulation

Not Operational:
Clock Skew < Data Delay

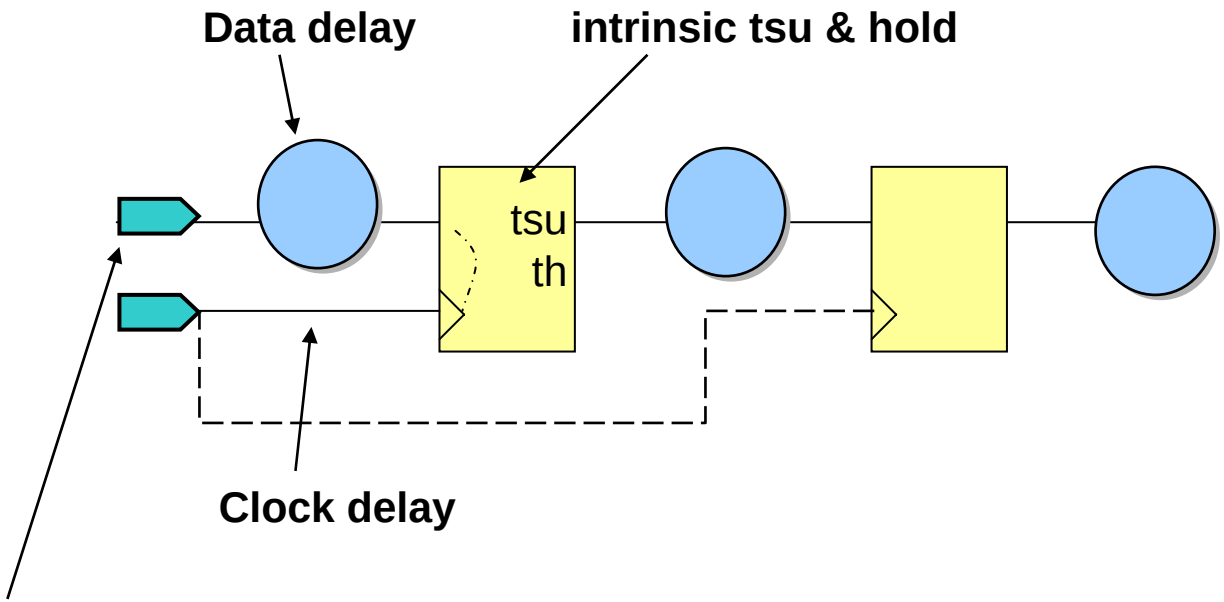
- Info: Found hold time violation between source pin or register mult_mega:u4|lpm_mult:lpm_mult_component|altshift:external_late
  - Info: + Largest clock skew is 1.018 ns
    - Info: + Longest clock path from clock clk to destination register is 3.566 ns
    - Info: - Shortest clock path from clock clk to source register is 2.548 ns
  - Info: - Micro clock to output delay of source is 0.099 ns
  - Info: - Shortest register to register delay is 0.648 ns
  - Info: + Micro hold delay of destination is 0.000 ns

Processing | System

List Paths Window

Copyright © 2004 Altera Corporation

I/O Setup (tsu) & Hold (th) Analysis

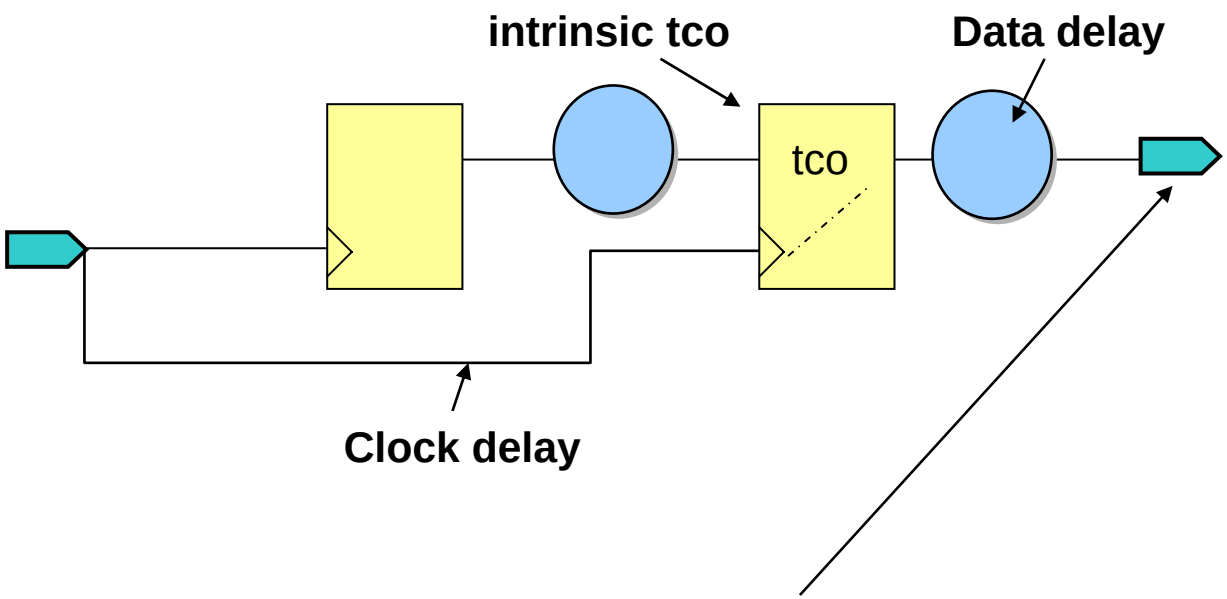


tsu = data delay - clock delay + intrinsic tsu

th = clock delay - data delay + intrinsic th

Copyright © 2004 Altera Corporation

I/O Clock-to-Output Analysis (tco)



clock delay + intrinsic tco + data delay = tco

Copyright © 2004 Altera Corporation

I/O Timing Analyzer

tsu, tco, th Will All Show up in the Timing Analyzer Report

filtref Compilation Report:1

- Compilation Report
  - Legal Notice
  - Flow Summary
  - Flow Settings
  - Flow Elapsed Time
  - Flow Log
  - Analysis & Synthesis
  - Fitter
    - Fitter Summary
    - Fitter Settings
    - Fitter Device Options
    - Fitter Equations
    - Floorplan View
    - Resource Section
    - Fitter Messages
  - Assembler
  - Timing Analyzer
    - Timing Analyzer Settings
    - Timing Analyzer Summary
    - Clock Settings Summary
    - tsu
    - tco
    - th
    - Minimum tco
    - Timing Analyzer Messages

tsu

	Slack	Required tsu	Actual tsu	From	To	To Clock
1	N/A	None	2.702 ns	d[3]	taps:inst|xn[3]~reg0	clk
2	N/A	None	2.664 ns	d[0]	taps:inst|xn[0]~reg0	clk
3	N/A	None	2.648 ns	d[7]	taps:inst|xn[7]~reg0	clk
4	N/A	None	2.642 ns	d[4]	taps:inst|xn[4]~reg0	clk
5	N/A		2.587 ns	d[1]	taps:inst|xn[1]~reg0	clk
6	N/A		2.573 ns	d[2]	reg0	clk
7	N/A	None	2.546 ns	d[6]	reg0	clk
8	N/A	None	2.521 ns	d[5]	taps:inst|xn[5]~reg0	clk
9	N/A	None	0.550 ns	newt	taps:inst|xn_2[1]~reg0	clk
10	N/A	None	0.550 ns	newt	taps:inst|xn[1]~reg0	clk
11	N/A	None	0.550 ns	newt	taps:inst|xn_1[1]~reg0	clk
12	N/A	None	0.550 ns	newt	taps:inst|xn_3[1]~reg0	clk
13	N/A	None	0.550 ns	newt	taps:inst|xn[2]~reg0	clk
14	N/A	None	0.550 ns	newt	taps:inst|xn_2[2]~reg0	clk
15	N/A	None	0.550 ns	newt	taps:inst|xn_3[2]~reg0	clk
16	N/A	None	0.550 ns	newt	taps:inst|xn_1[2]~reg0	clk
17	N/A	None	0.550 ns	newt	taps:inst|xn_2[4]~reg0	clk
18	N/A	None	0.550 ns	newt	taps:inst|xn[4]~reg0	clk
19	N/A	None	0.550 ns	newt	taps:inst|xn_3[4]~reg0	clk
20	N/A	None	0.550 ns	newt	taps:inst|xn_1[4]~reg0	clk
21	N/A	None	0.550 ns	newt	taps:inst|xn_2[5]~reg0	clk
22	N/A	None	0.550 ns	newt	taps:inst|xn_3[5]~reg0	clk
23	N/A	None	0.550 ns	newt	taps:inst|xn_1[5]~reg0	clk

Value

Pin Name

Clock Name

Select Parameter

***Note**: Timing Analysis of tpd is similar*

Copyright © 2004 Altera Corporation

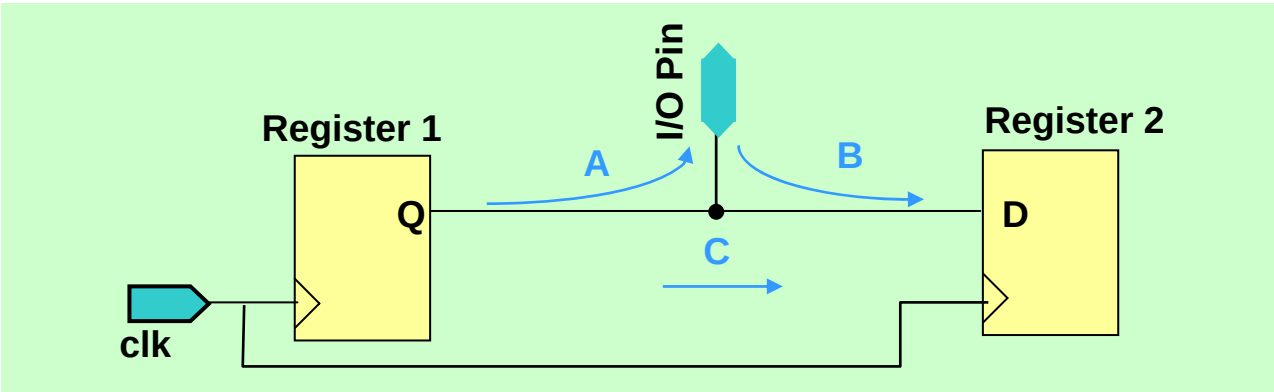
Timing Analysis Options

- Used to Limit which Paths Are Displayed
- Global Cut Timing Options (On by Default)
  - Cut Paths between Unrelated Clock Domains
  - Cut Off Feedback from I/O Pins (Next Slide)
  - Cut Off Clear & Preset Signal Paths
  - Cut Off Read during Write Signal Paths
- Timing Analyzer Options
  - Display Paths that Do Not Meet Timing Only

Copyright © 2004 Altera Corporation

Cut Off Feedback from I/O Pin

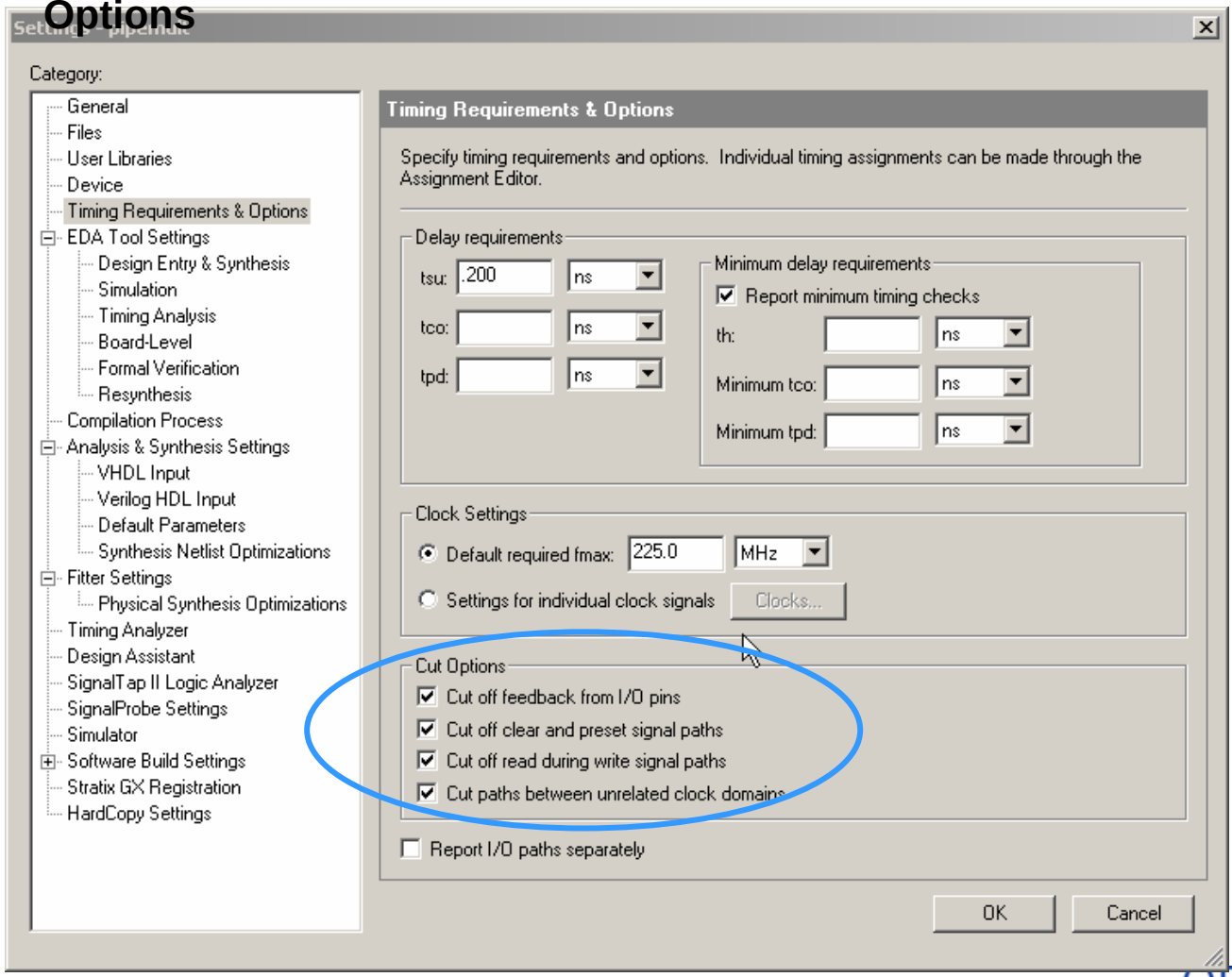
- Breaks Bidirectional I/O Pin from Analysis
- When On, Paths A & B Are Valid; C Is Not
- When Off, Paths A, B, & C Are Valid



Copyright © 2004 Altera Corporation

Cut Options

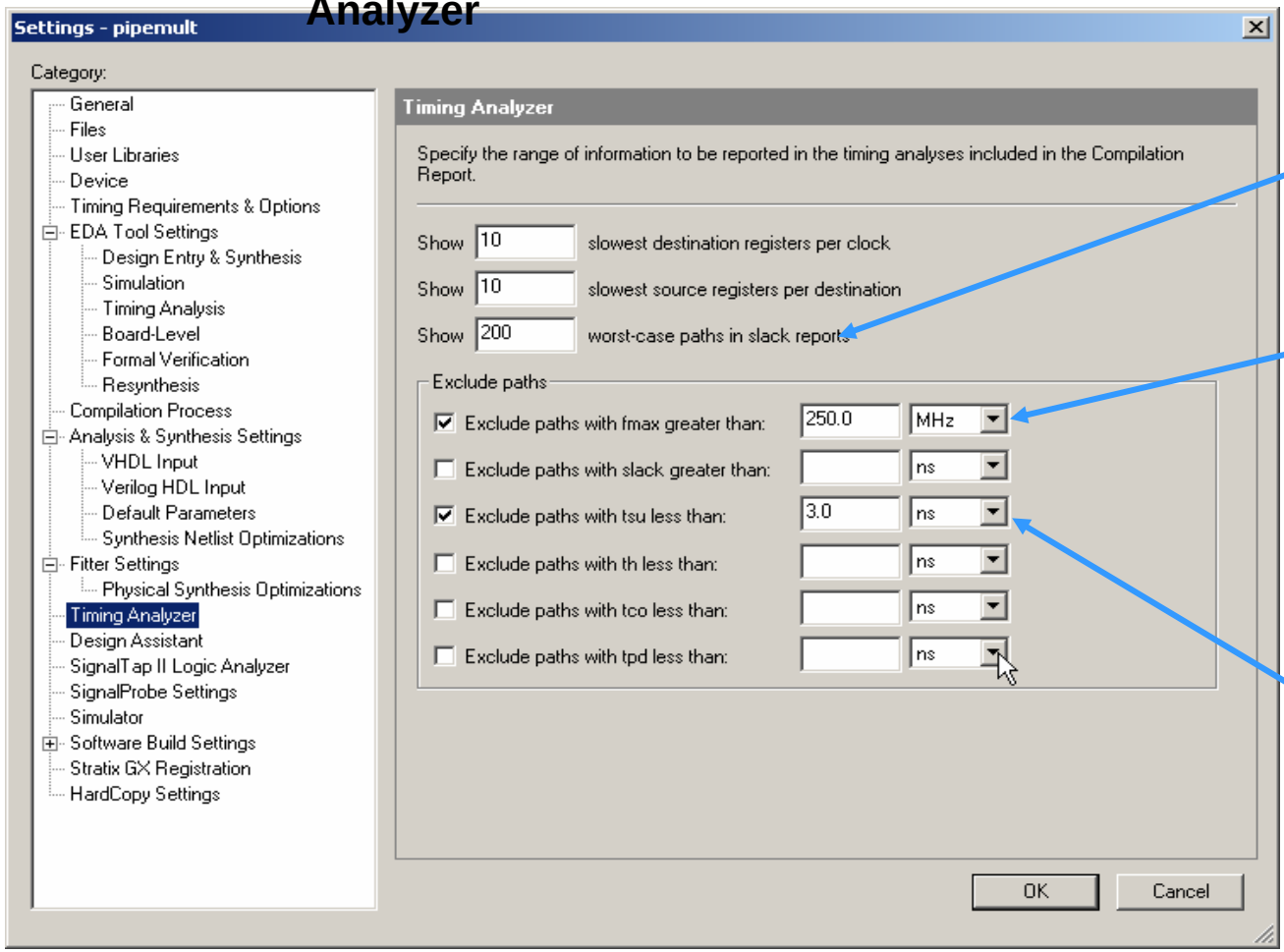
Assignments ⇒ Settings ⇒ Timing Requirements & Options



Copyright © 2004 Altera Corporation

Timing Analyzer Options

Assignments ⇒ Settings ⇒ Timing Analyzer



List 200 Paths

List Paths with Fmax Less than 250 MHz

List Paths with Tsu Greater than 3 ns

Copyright © 2004 Altera Corporation

Single-Clock Timing Analysis

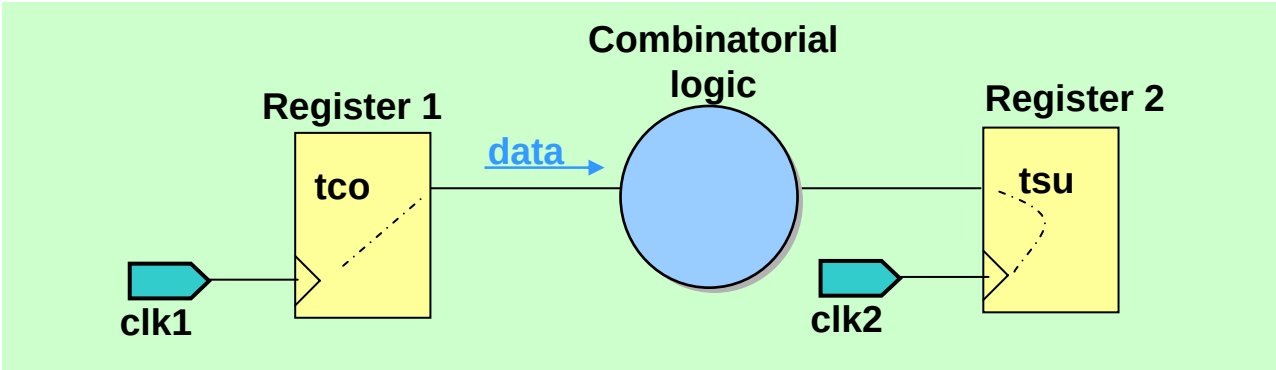
- Performed Automatically during Each Compile
- Detects Clocks Automatically If No Assignments Are Made
  - Single or Multiple Asynchronous Clock Domains
  - Still Important to Specify Clock Frequencies
- Displays Results in the Timing Analyzer Report

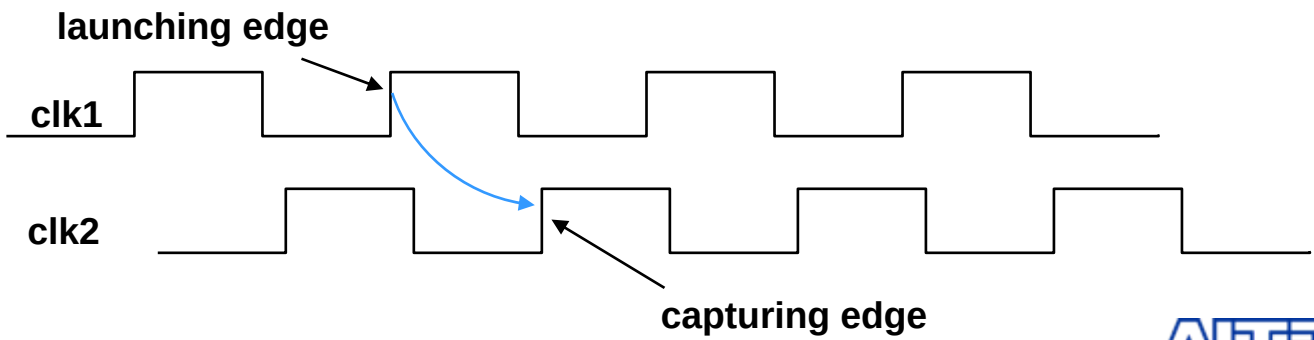
Copyright © 2004 Altera Corporation

Multi-Clock Frequency Analysis

⑤ Analyzes Timing on Register-to-Register Paths Controlled by Different Synchronous Clocks

- Individual Clocks Treated as Same Frequency & Phase Unless Specified





Copyright © 2004 Altera Corporation

Minimum Timing Analysis

- Reports Best-Case I/O Results
  - Minimum tco
  - th
  - Minimum tpd
- Uses Fastest Timing Model
  - Fastest Process
  - Highest Voltage
  - Lowest Temperature

Copyright © 2004 Altera Corporation

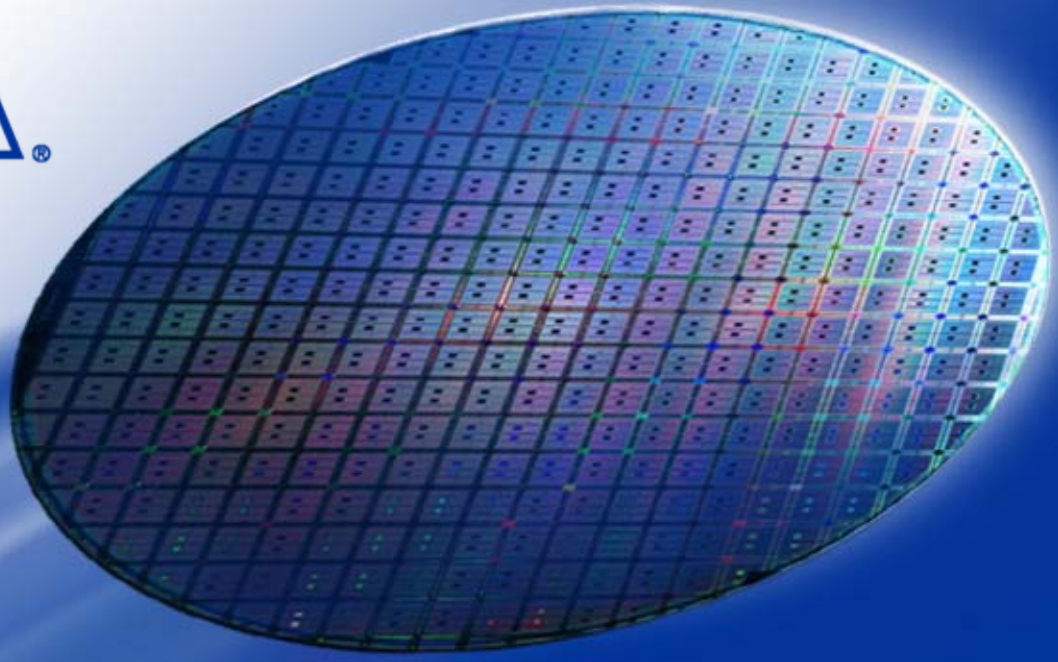
Please go to Exercise 3

Copyright © 2004 Altera Corporation

Exercise 3 Review

- Quartus II Project Setup
  - Project Wizard
- Quartus II Compilation
  - Assigning a device
  - Choosing a focus point
- Reviewing the Compilation Report
  - fmax, tsu, tco
  - Device statistics

Copyright © 2004 Altera Corporation



Quartus II NativeLink

Copyright © 2004 Altera Corporation

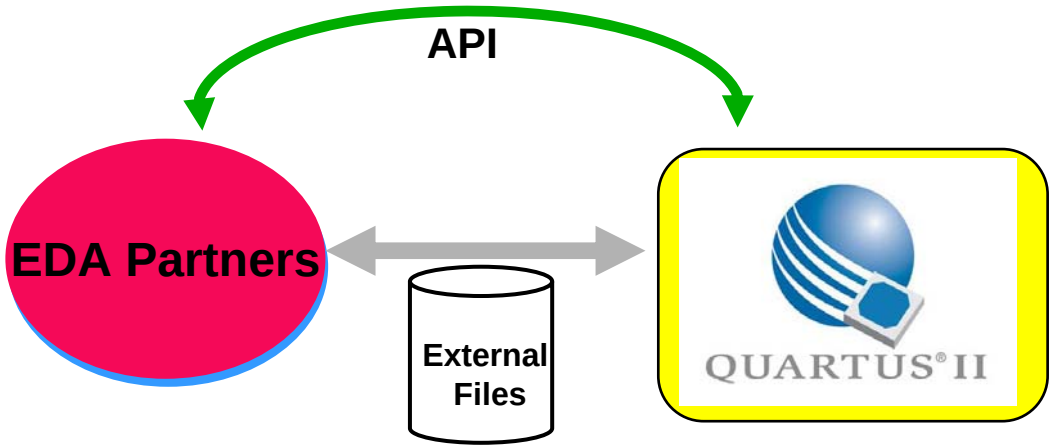
EDA Interfaces Introduction

- Interface with Industry-Standard EDA Tools that Generate a Netlist File
  - EDIF 200
  - VHDL ’87 or ’93
  - Verilog
- NativeLink Interface Provides Seamless Integration with 3rd-party EDA Software Tools
  - Tools Pass Information/Commands in Background
  - Designers Can Complete Entire Design in One Tool

Copyright © 2004 Altera Corporation

NativeLink

- ⑤ Comprised of Two Components
  - External Files
    - ③WYSIWYG (What You See Is What You Get) ATOM Netlist Files (EDIF, Verilog, VHDL)
    - ③Cross Reference Files (Ex. XRF)
    - ③Timing Files (Ex. SDO)
  - Application Programming Interface (API)
    - ③Pre-Defined Interface of Commands/Functions



Copyright © 2004 Altera Corporation

WYSIWYG ATOM Primitives

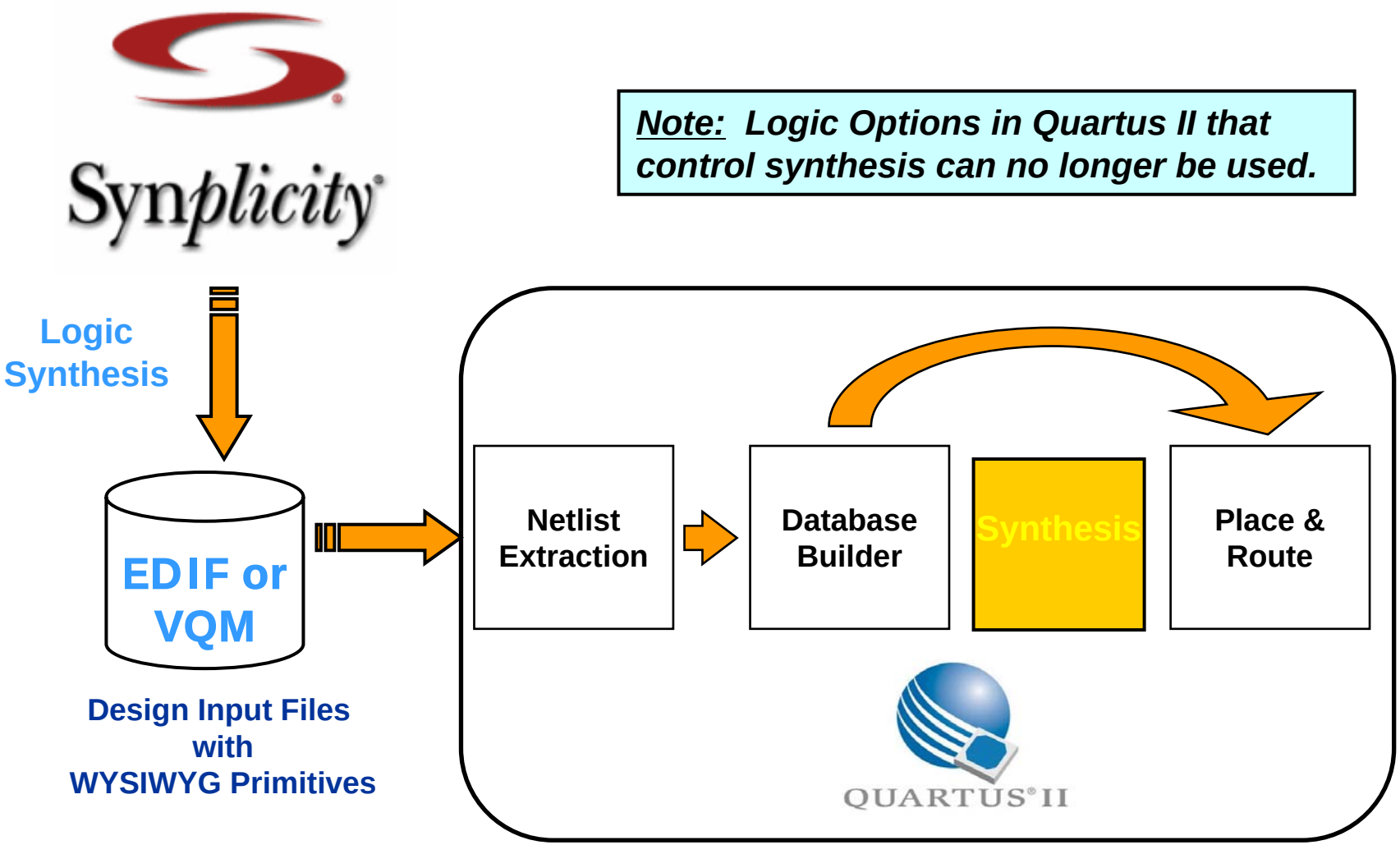
- Set of Design Primitives that Support WYSIWYG Compilation
- Provide Direct Control of How a Design Is Technology-Mapped to a Specific Target Device
- Allow Synthesis Vendors to Provide an Optimal Realization of a Design for Each Architecture

```
00104   stratix_lcell par_err_high_0_and2_62_Z (
00105     .combout(par_err_high_0_and2_62),
00106     .dataa(ram_out_c_11),
00107     .datab(ram_out_c_10),
00108     .datac(ram_out_c_9),
00109     .datad(ram_out_c_8)
00110 );
00111 defparam par_err_high_0_and2_62_Z.operation_mode="normal";
00112 defparam par_err_high_0_and2_62_Z.lut_mask="9669";
00113 defparam par_err_high_0_and2_62_Z.synch_mode="off";
00114 defparam par_err_high_0_and2_62_Z.sum_lutc_input="datac";
```

Synplify .vqm

Copyright © 2004 Altera Corporation

WYSIWYG Compilation Flow



Note: ***Logic Options in Quartus II that control synthesis can no longer be used.***

Copyright © 2004 Altera Corporation

Quartus II / Synplify Pro Design Flows

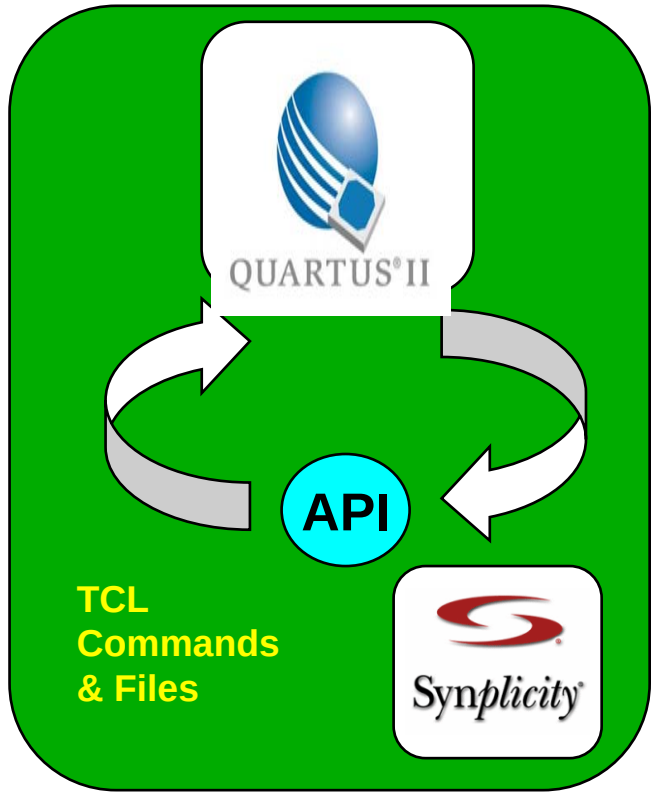
- Quartus II Driven Flow
  - Quartus II launches Synplify Pro in Background to Perform Verilog or VHDL Synthesis
- Synplify Pro Driven Flow
  - Synplify Pro Starts Quartus II to Perform Place & Route
- File Based Flow
  - Manual File Transfer
  - Quartus II & Synplify Pro Executed Independently

Copyright © 2004 Altera Corporation

Quartus II Driven Flow

Quartus II drives Synplify Pro

- Synplify Pro launched from within Quartus II
- Quartus II automatically
  - ③Generates netlists (AHDL & Graphics)
  - ③Reads in the netlists (VHDL or Verilog)
- No Manual Setup of Synplify Pro Required



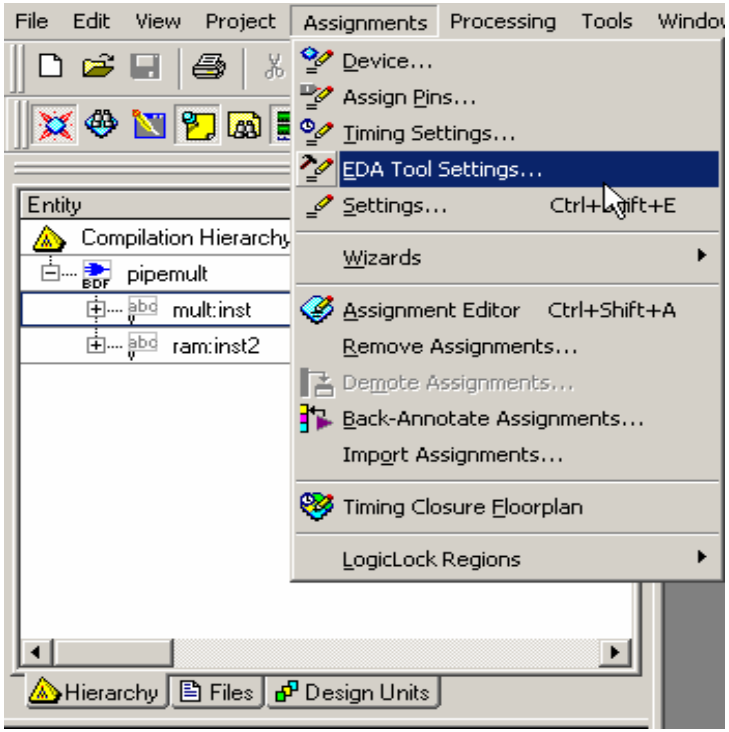
Copyright © 2004 Altera Corporation

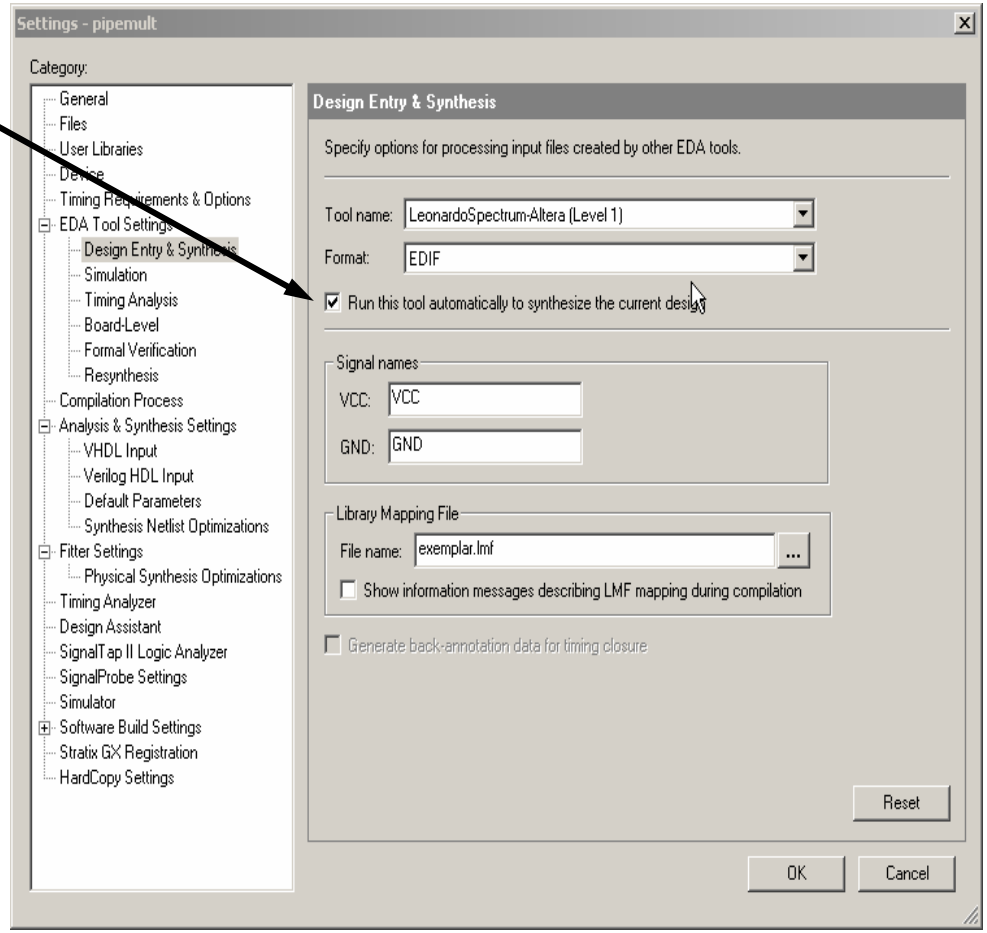
Quartus II Driven Flow

Assignments ⇒ EDA Tool Settings

Check the "Run …" Button to Launch the EDA Tool in the Background

Leave Unchecked for File Based Flow



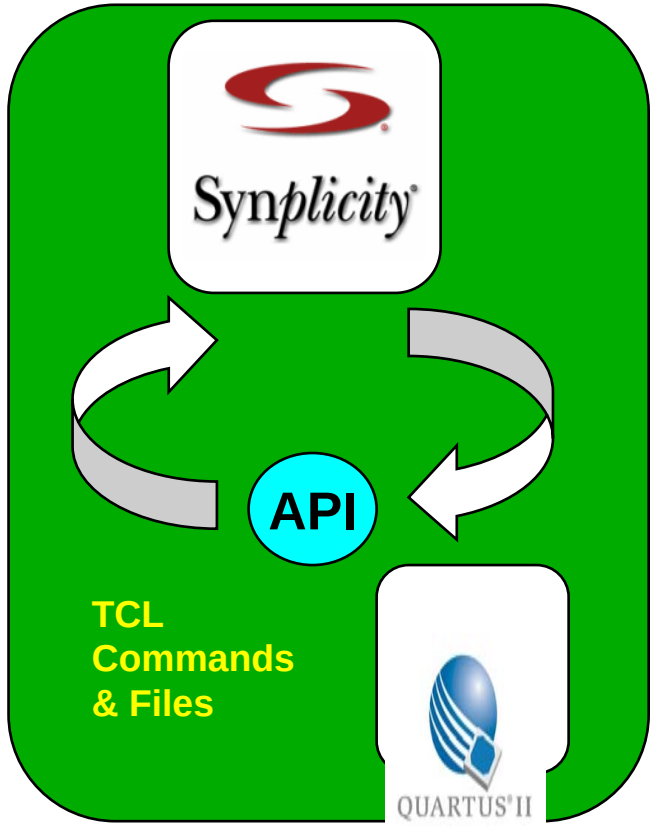


Copyright © 2004 Altera Corporation

Synplify Pro Driven Flow

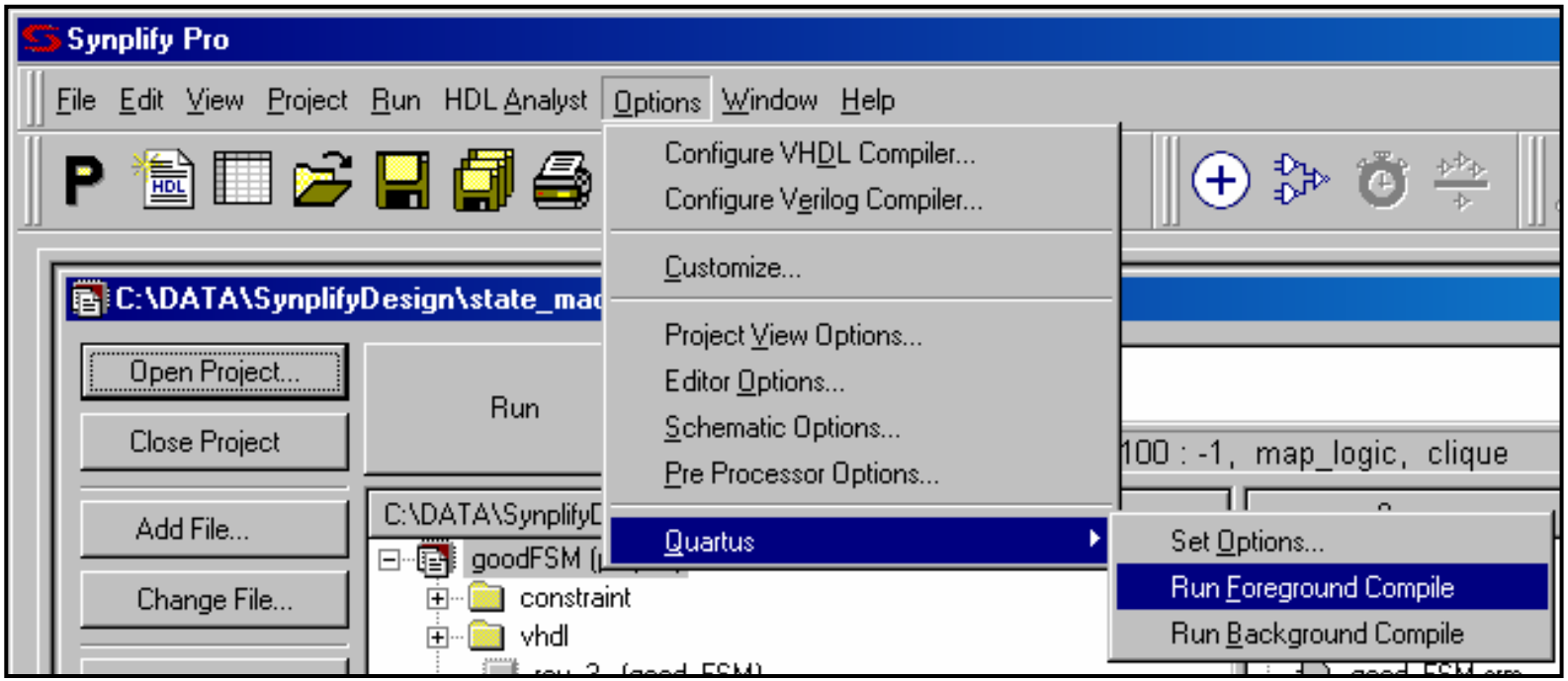
Synplify Pro drives Quartus II

- Offers the tightest code-level integration between tools
- Quartus II appears ‘Native’ in Synplify Pro



Copyright © 2004 Altera Corporation

Synplify Pro Driven Flow



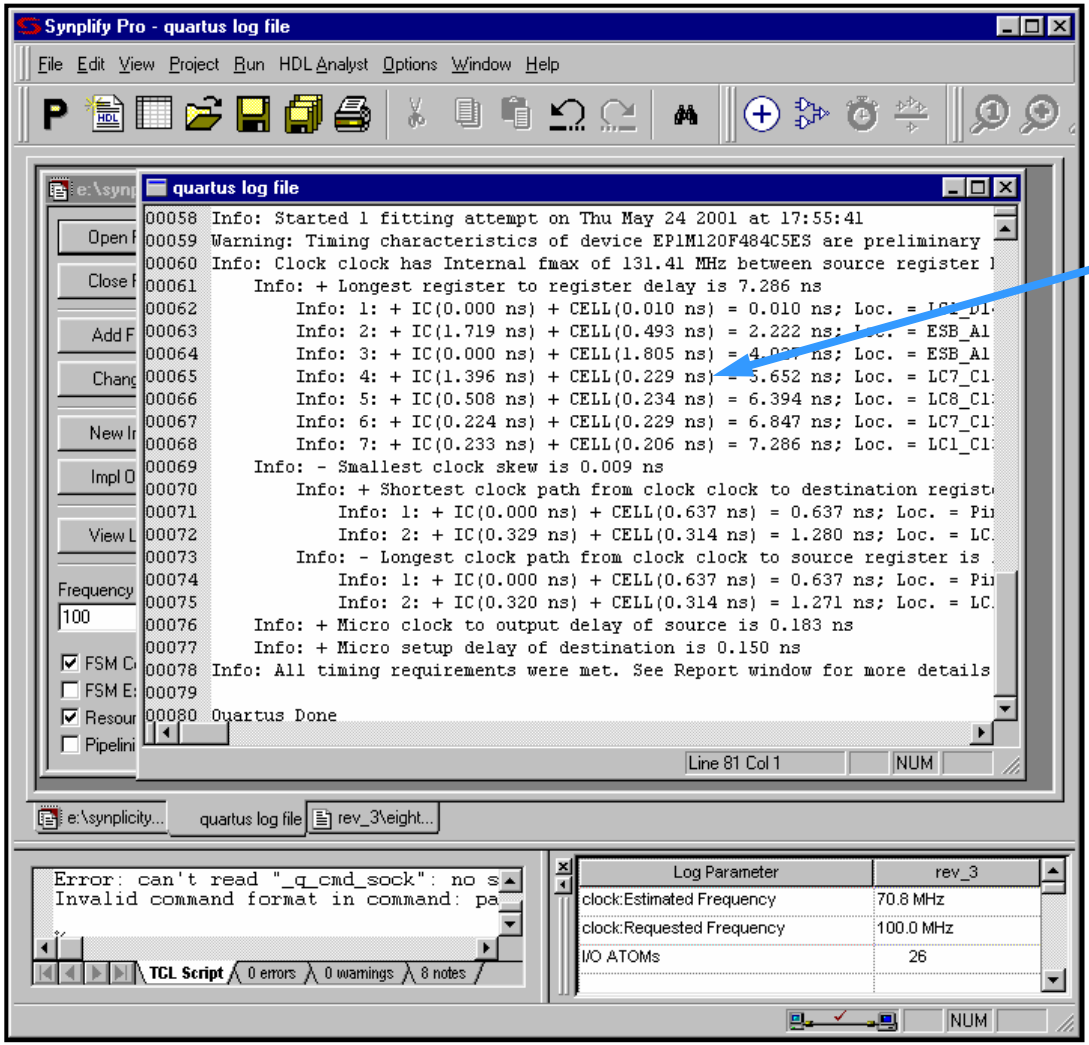
Copyright © 2004 Altera Corporation

SynplifyPro to QuartusII Interface

- Set Option
  - Invokes Quartus II with Project Settings from the Synplify Pro Run
- Run Foreground Compile
  - Runs Quartus II with the Project Settings from the Synplify Pro Run
- Run Background Compile
  - Runs Quartus II in Batch Mode with the Project Settings from the Synplify Pro Run
  - Quartus II Log File Is Displayed in the Synplify Pro User Interface with Updates of the Quartus II Compilation

Copyright © 2004 Altera Corporation

Background Compile



Quartus II Log File

All Quartus II Compilation Messages Appear in the in the Synplify Pro Window as if "Native" to the Software.

Copyright © 2004 Altera Corporation

Synplify Pro Output TCL Files

- <PROJECT_NAME>_CONS.TCL
  - Creates Quartus II Project
  - Starts Compilation
- <PROJECT_NAME>.TCL
  - Forward Annotates Device Selection
  - Forward Annotates Timing Constraints & Settings
- <PROJECT_NAME>_RM.TCL
  - Removes Previous Constraints
  - Saves Old Constraints in <PROJECT_NAME>_PREV.TCL

Copyright © 2004 Altera Corporation

Constraint Forward Annotation

⑤ These Constraints Can Be Passed to Quartus II

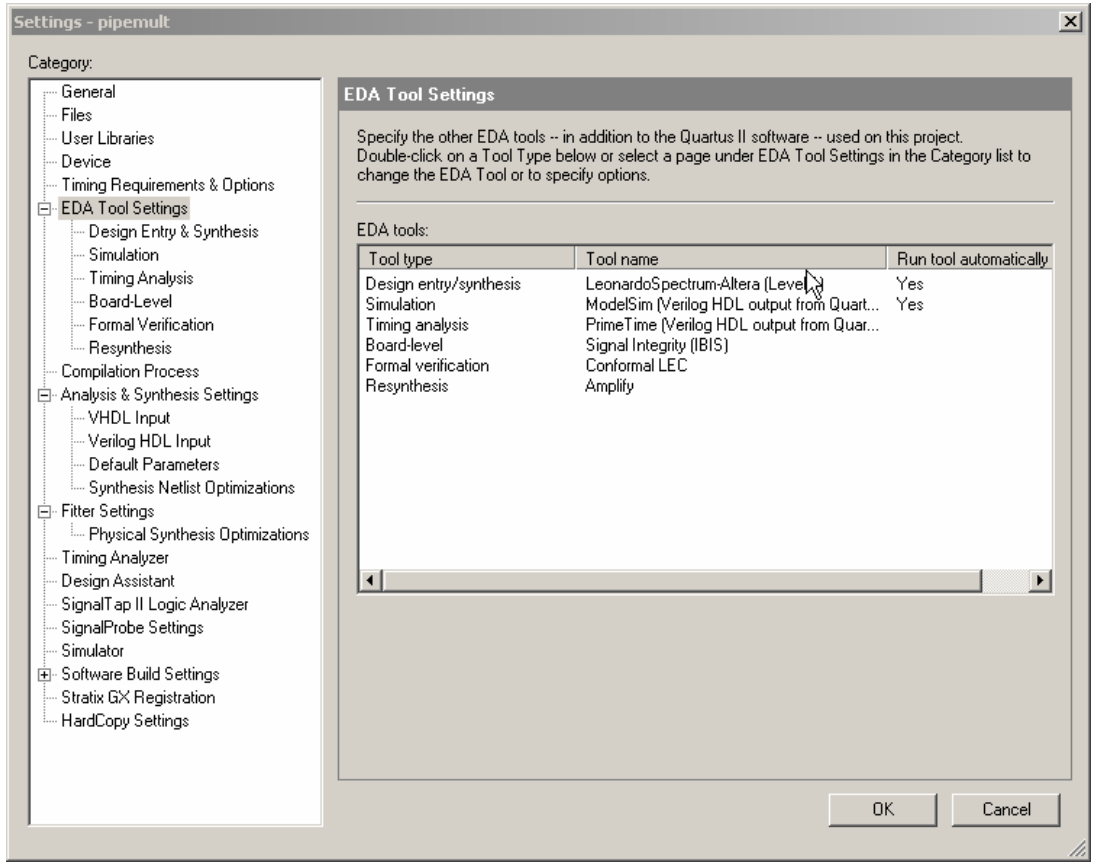
- I/O Delay
- Clock Frequency
- Clock Duty Cycle
- Multi-Clock Settings
- Multi-Cycle Paths
- Pin Assignments
- Cliques/LogicLock Regions

Copyright © 2004 Altera Corporation

File Based Flow

⑤Work Separately in Individual Tools

- Synthesis
- Simulation
- Timing Analysis
- Board-Level
- Formal Verification
- Resynthesis



Copyright © 2004 Altera Corporation

Third Party Tool Support

Synthesis Tools

- LeonardoSpectrum™
- Precision
- DesignCompiler-FPGA
- FPGA Compiler II™
- FPGA Express™
- Synplify
- Synplify Pro
- Amplify

Verification Tools

- ModelSim®
- ModelSim-Altera
- Cadence Verilog-XL
- Cadence NC-Verilog
- Cadence NC-VHDL
- Innoveda BLAST
- PrimeTime®
- Synopsys® VCS & VSS
- Mentor Graphics® Tau
- Synopsys Scirocco

Copyright © 2004 Altera Corporation

Please go to Exercise 4

Copyright © 2004 Altera Corporation

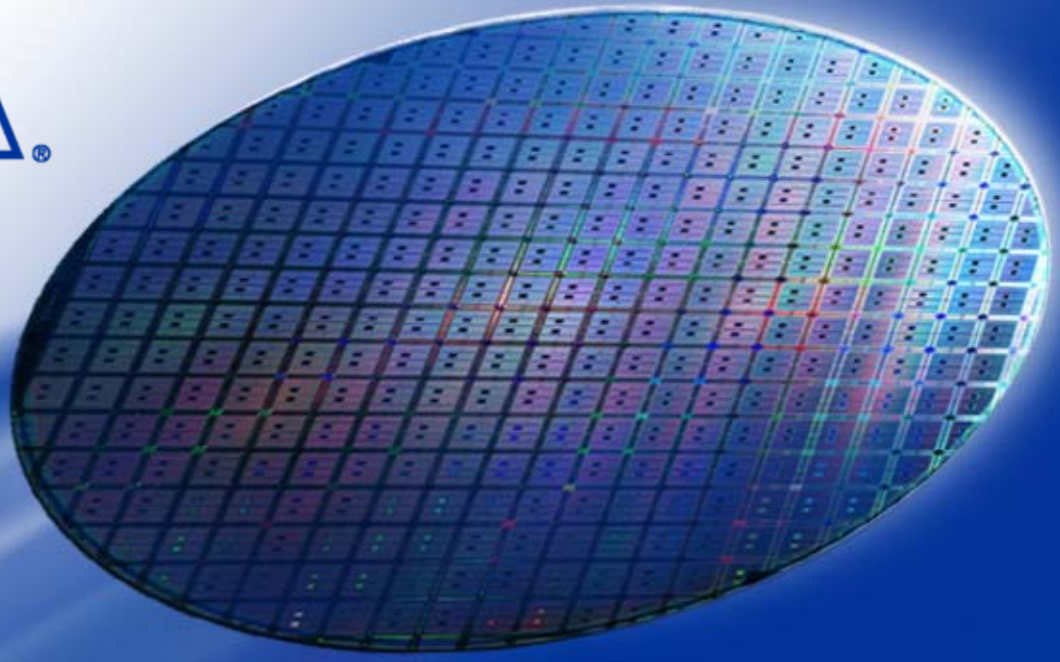
What has been covered so far…?

…The EDA Synthesis Flow

⑤ Synplify Pro
- Design Entry
- Synthesis Settings
- Synthesis

⑤ Quartus II
- Design Entry
- Place & Route
- Compilation Statistics
  - ③Performance
  - ③Area
- Device-Specific Logic Options

Copyright © 2004 Altera Corporation



Optimizing Designs with Synplify Pro & Quartus II

Copyright © 2004 Altera Corporation

Optimizing Designs

- ⑤Hierarchical Optimization
- ⑤“Black-Boxing”
- ⑤Attributes & Directives
- ⑤Optimization Suggestions

Copyright © 2004 Altera Corporation

Hierarchical Designs

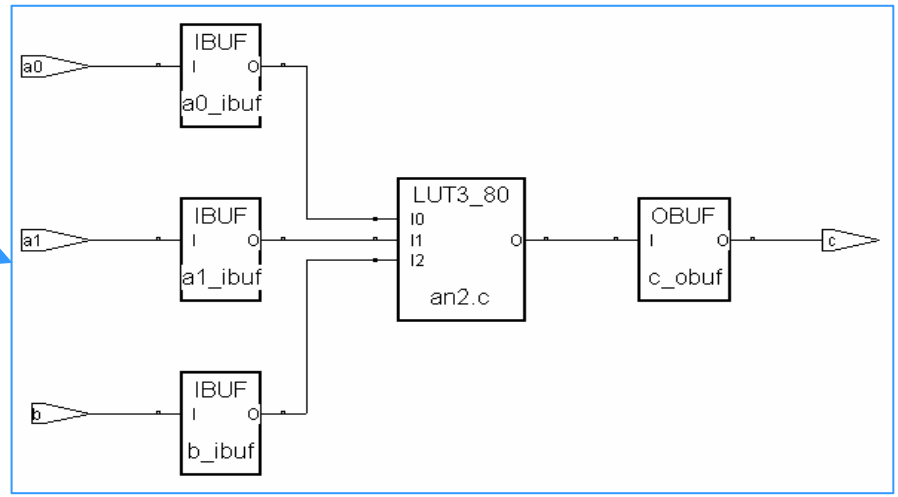
- Mapper Starts with the Original hierarchy
- Flattens Hierarchy & Reduces Logic
- Performs Global Analysis and Optimizes Hierarchy by Creating New Structures, if Needed
- Rebuilds Hierarchy by Keeping Existing Boundaries or Creating New Ones
- Provides Timing Budget for Each Hierarchical Block
- Converges on the Timing Goal by Making an Area/Time Tradeoff

Copyright © 2004 Altera Corporation

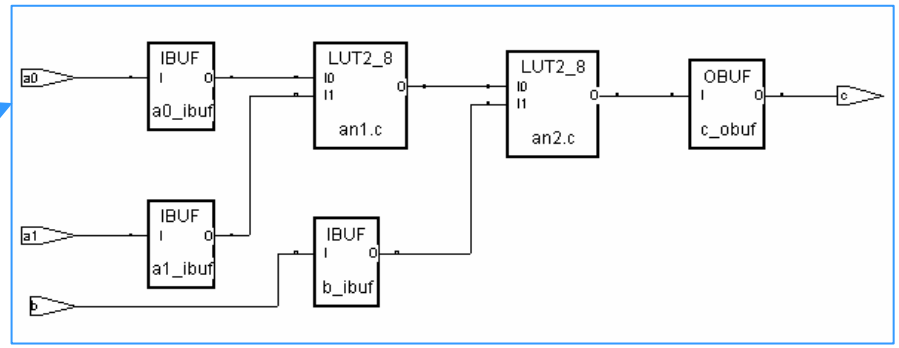
Hierarchy Optimization (cont.)

Logic Implemented in One Lookup table with Hierarchical Boundary Optimization

```
module and2_x(a, b, c);
input a, b;
output c;
assign c = a & b;
endmodule

module top(a0, a1, b, c);
input a0, a1, b;
output c;
and2_x an1(a0, a1, an1_out);
and2_x an2(an1_out, b, c);
endmodule
```



Two Levels of Logic Generated without Hierarchical Boundary Optimization



Copyright © 2004 Altera Corporation

Black-Boxing

What is a "black-box" and why is it used?

- A Module/Entity that Is Left Untouched (Un-Synthesized) by Synplify Pro
  - Surrounding Logic Still Synthesized
- Output Netlist Includes Port Connections to "Black-Box"
- "Black-Box" code is Inserted by Quartus II
- Use for LPMs or Altera Specific Megafunctions
  - CDR (Clock Data Recovery)
  - RAM/ROM
  - CAM (Content Addressible Memory)
  - CLOCKLOCK® /CLOCKBOOST®

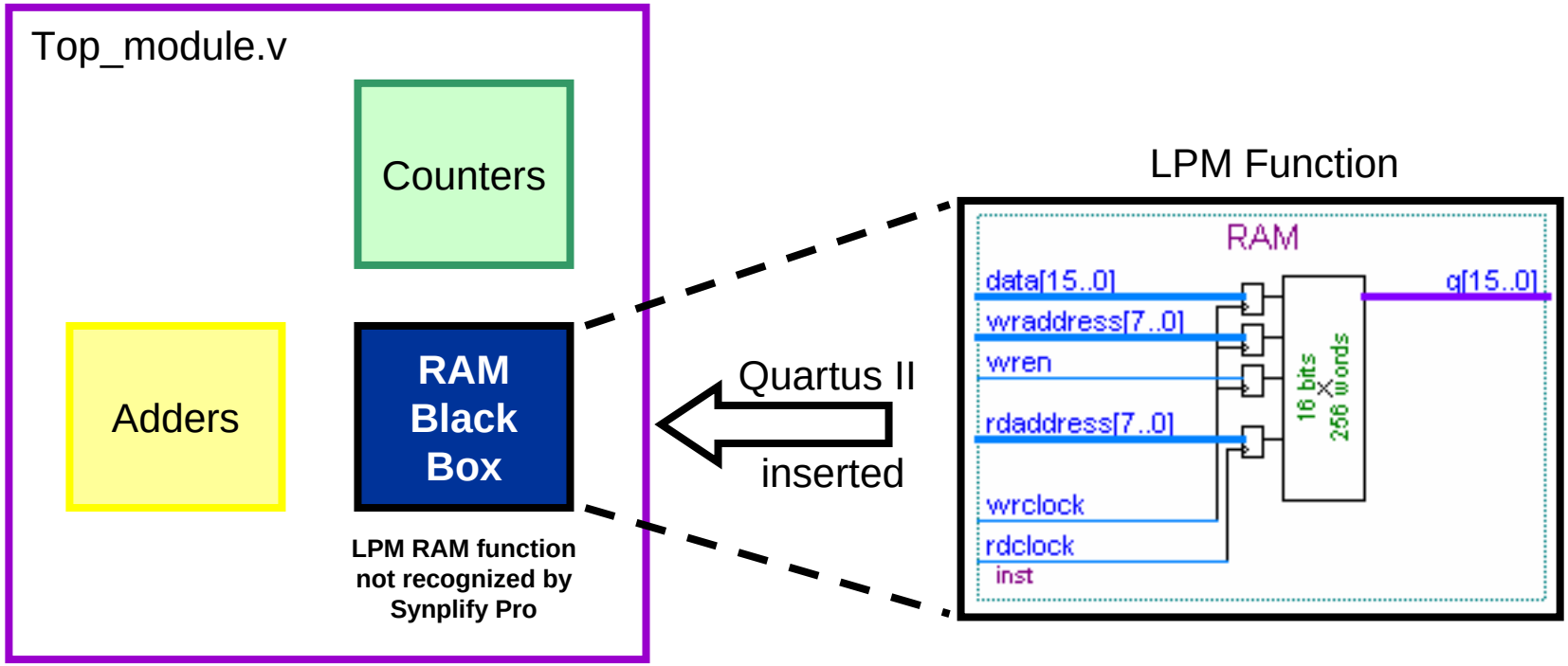
Copyright © 2004 Altera Corporation

LPM Functions

- LPM Library Elements
  - **L**ibrary of **P**arameterized **M**odules
    - Pre-Designed Modules You Can Customize for Your Design
  - Edit
    - I/O Ports
    - Parameters
      - Width of Data
      - Memory Depth
  - Optimized for All Altera Architectures
  - Utilize the Architectural Features of All Altera Devices
    - ESBs/EABs
    - CDR
    - PLLs
    - LVDS

Copyright © 2004 Altera Corporation

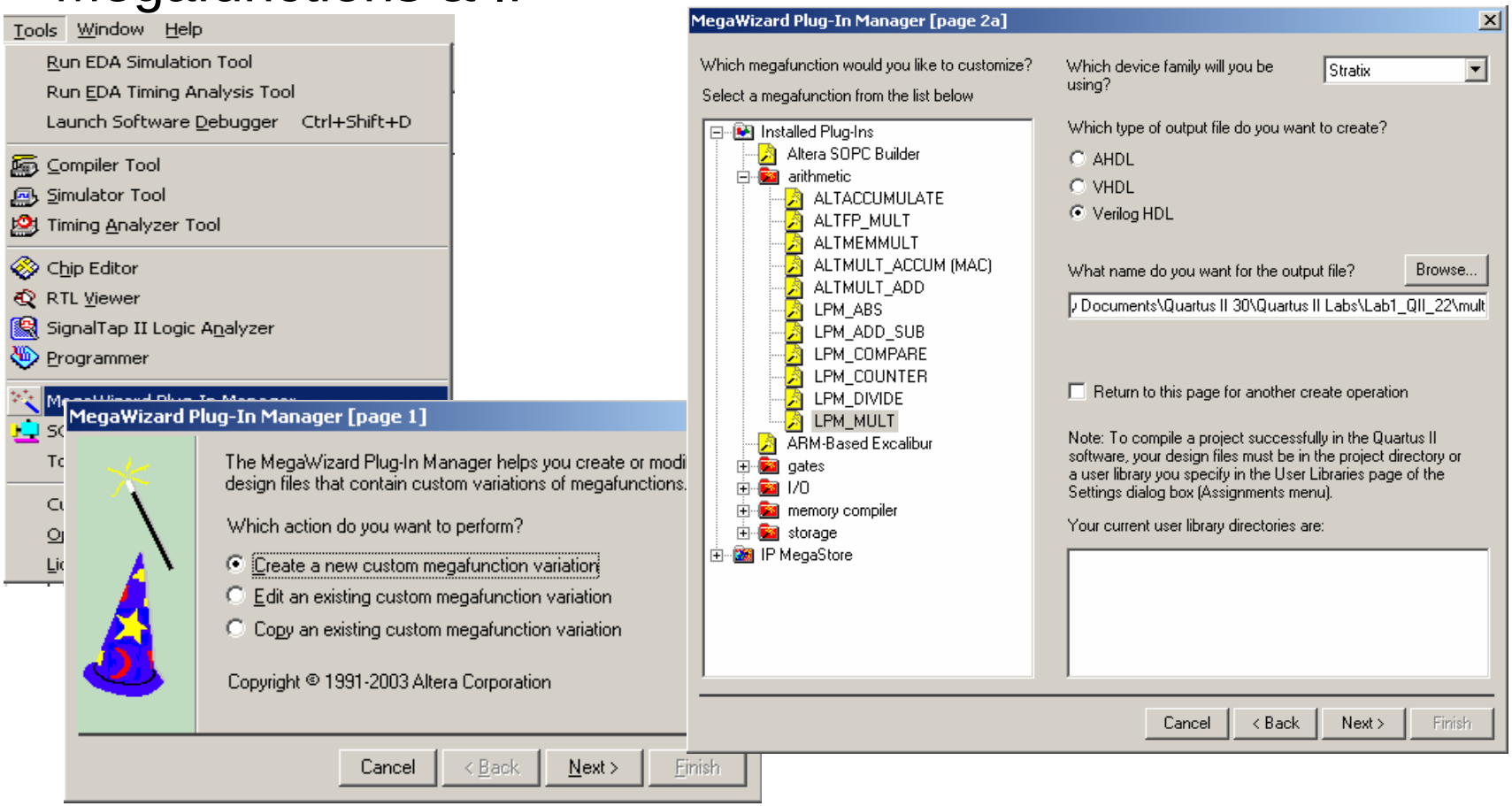
Black-boxing



Copyright © 2004 Altera Corporation

MegaWizard Plug-In Manager

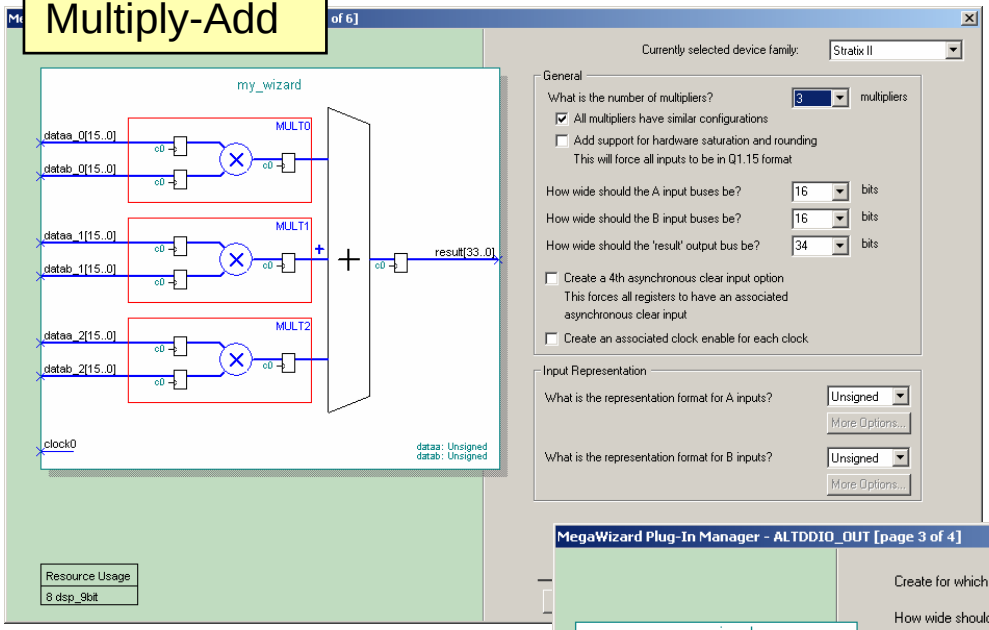
⑤ Eases Implementation of Megafunctions & IP



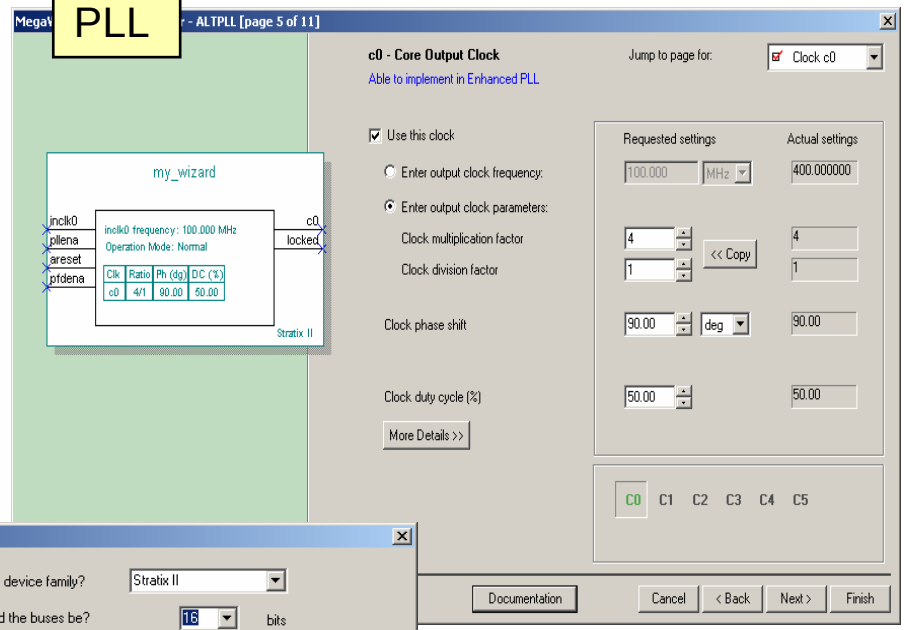
Copyright © 2004 Altera Corporation

MegaWizard Examples

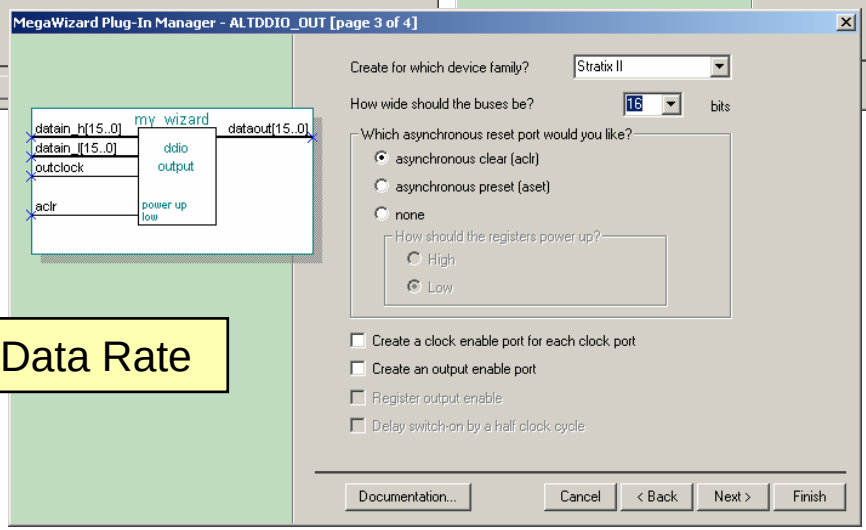
Multiply-Add



PLL



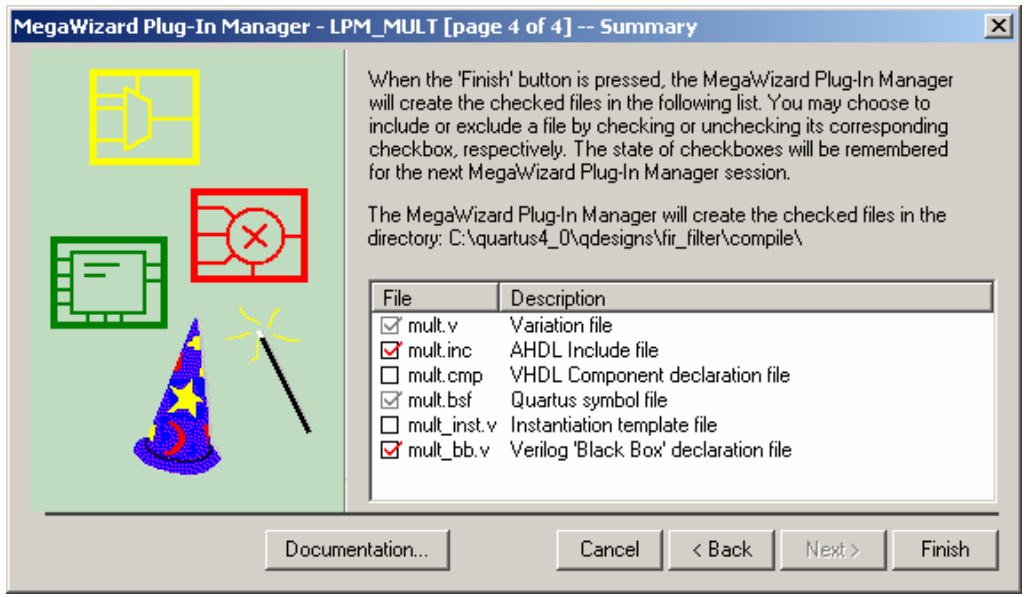
Double-Data Rate



Copyright © 2004 Altera Corporation

MegaWizard Output File Selection

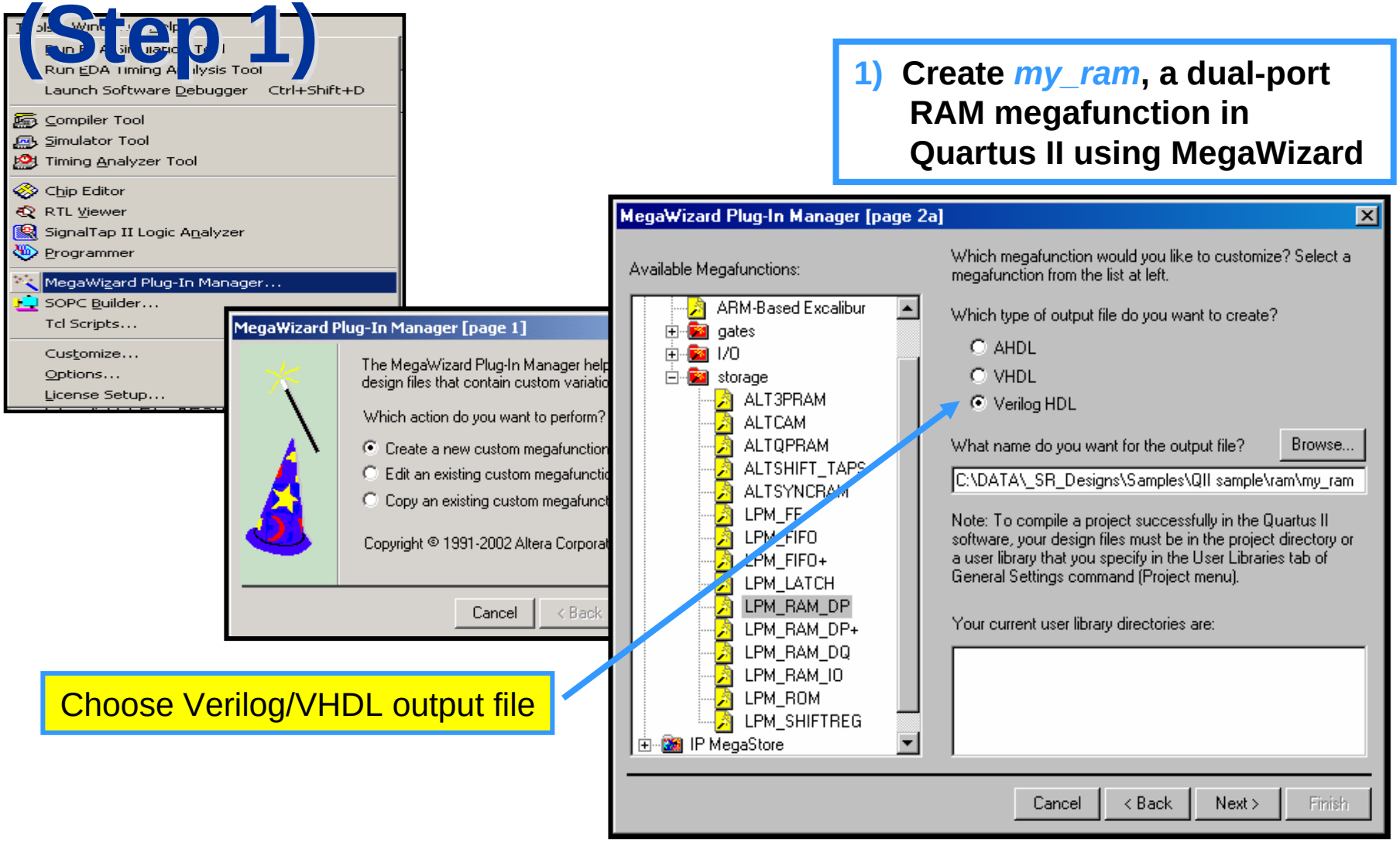
- Default
  - HDL Wrapper File
- Selectable
  - HDL Instantiation Template
  - VHDL Component Declaration (CMP)
  - Quartus II Symbol (BSF)
  - Verilog Black Box



Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 1)

1) **Create *my_ram*, a dual-port RAM megafunction in Quartus II using MegaWizard**

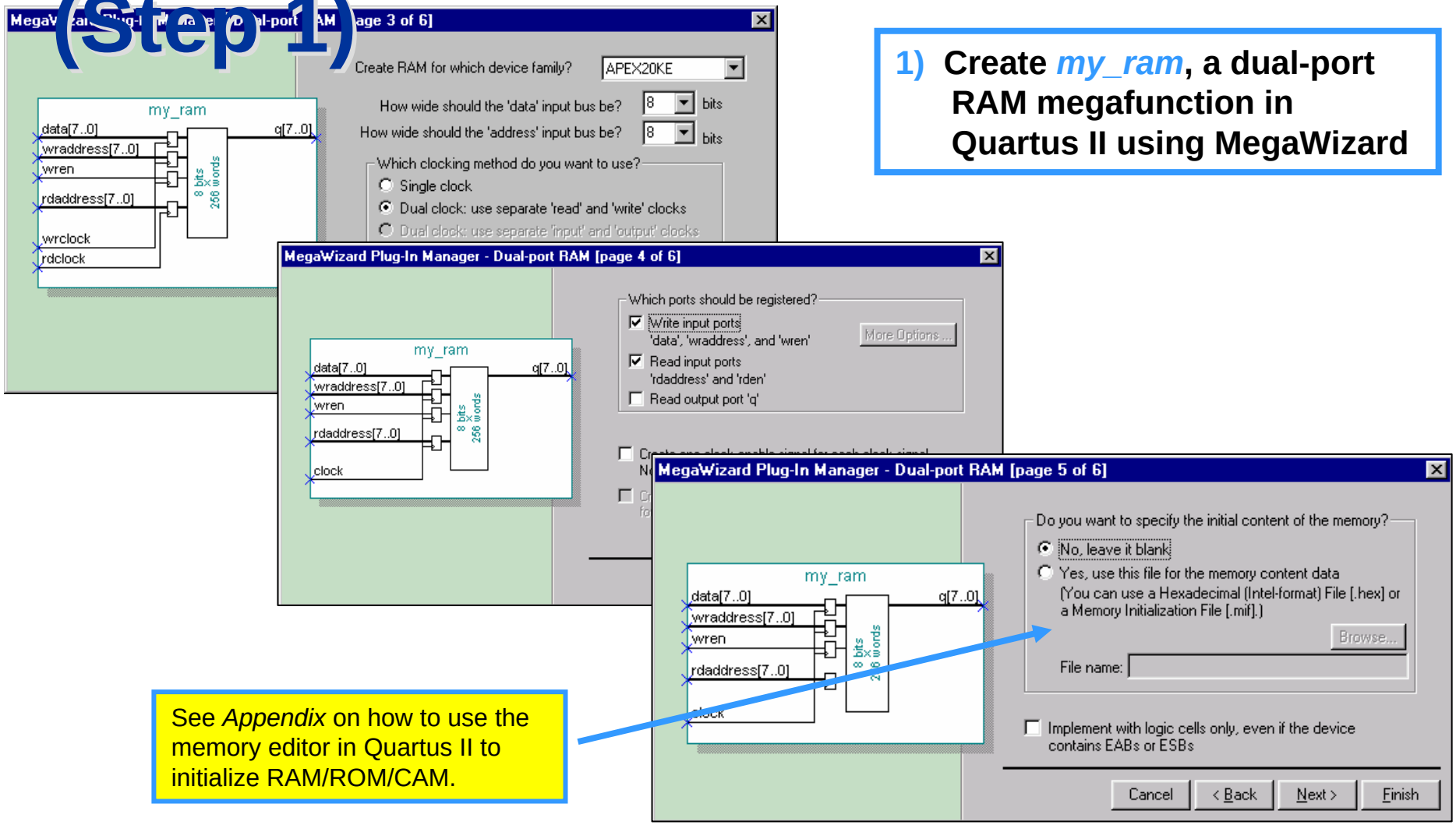


Choose Verilog/VHDL output file

Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 1)

1) **Create *my_ram*, a dual-port RAM megafunction in Quartus II using MegaWizard**



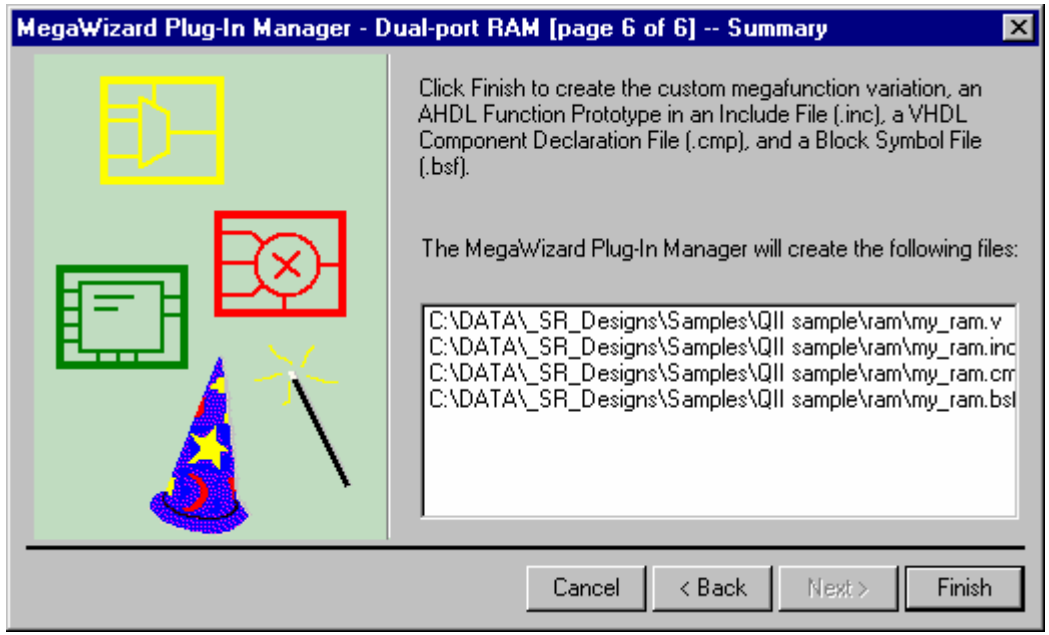
See *Appendix* on how to use the memory editor in Quartus II to initialize RAM/ROM/CAM.

Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 1)

Finished!!

1) Create *my_ram*, a dual-port RAM megafunction in Quartus II using MegaWizard



Copyright © 2004 Altera Corporation

Verilog Wrapper Function

Verilog Dual-Port RAM function

A "wrapper" function instantiates the LPM and sets its ports and parameters

```
module my_ram (
                data,
                wraddress,
                rdaddress,
                wren,
                wrclock,
                rdclock,
                q);

                input       [7:0]  data;
                input       [7:0]  wraddress;
                input       [7:0]  rdaddress;
                input        wren;
                input        wrclock;
                input        rdclock;
                output      [7:0]  q;

                wire [7:0] sub_wire0;
                wire [7:0] q = sub_wire0[7:0];

                lpm_ram_dp      lpm_ram_dp_component (
                                                .rdclock (rdclock),
                                                .wren (wren),
                                                .wrclock (wrclock),
                                                .data (data),
                                                .rdaddress (rdaddress),
                                                .wraddress (wraddress),
                                                .q (sub_wire0));
                defparam
                                lpm_ram_dp_component.lpm_width = 8,
                                lpm_ram_dp_component.lpm_widthad = 8,
                                lpm_ram_dp_component.lpm_indata = "REGISTERED",
                                lpm_ram_dp_component.lpm_wraddress_control = "REGISTERED",
                                lpm_ram_dp_component.lpm_rdaddress_control = "REGISTERED",
                                lpm_ram_dp_component.lpm_outdata = "UNREGISTERED",
                                lpm_ram_dp_component.lpm_hint = "USE_EAB=ON";

endmodule
```

Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 2)

2) "Black-box" my_ram in Synplify Pro and synthesize the *top_ram*

Note: DO NOT include my_ram.v in the Synplify Pro input file list

Include empty module for *my_ram* *The Megawizard automatically generates a black box file named my_ram_bb.v*

```
// Verilog code which will call dual-port RAM

module top_ram (top_data, top_wradd, top_rdadd, top_we, top_wrclk, top_rdclk,
                top_q);

input [7:0]     top_data, top_wradd, top_rdadd;
input           top_we, top_wrclk, top_rdclk;
output [7:0]    top_q;

my_ram U1 (.data (top_data), .wraddress (top_wradd), .rdaddress (top_rdadd),
           .wren (top_we), .wrclock (top_wrclk), .rdclock (top_rdclk),
           .q (top_q));

endmodule



module my_ram ( data, wraddress, rdaddress, wren, wrclock, rdclock, q)
           /* synthesis syn_black_box */;

input        [7:0]  data;
input        [7:0]  wraddress;
input        [7:0]  rdaddress;
input         wren;
input         wrclock;
input         rdclock;
output       [7:0]  q;

endmodule
```

Use Synthesis Directive

Copyright © 2004 Altera Corporation

VHDL Wrapper Function

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
LIBRARY lpm;
USE lpm.lpm_components.all;

ENTITY my_ram IS
        PORT
        (
        data            : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        wraddress       : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        rdaddress       : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        wren            : IN STD_LOGIC  := '1';
        wrclock         : IN STD_LOGIC ;
        rdclock         : IN STD_LOGIC ;
        q               : OUT STD_LOGIC_VECTOR (7 DOWNTO 0)
        );
END my_ram;

ARCHITECTURE SYN OF my_ram IS

        SIGNAL sub_wire0    : STD_LOGIC_VECTOR (7 DOWNTO 0);

        COMPONENT lpm_ram_dp
        GENERIC (
        lpm_width                       : NATURAL;
        lpm_widthad                     : NATURAL;
        lpm_indata                      : STRING;
        lpm_wraddress_control           : STRING;
        lpm_rdaddress_control           : STRING;
        lpm_outdata                     : STRING;
        lpm_hint                        : STRING
        );
        PORT (
        rdclock         : IN STD_LOGIC ;
        wren            : IN STD_LOGIC ;
        wrclock         : IN STD_LOGIC ;
        q               : OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
        data            : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        rdaddress       : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        wraddress       : IN STD_LOGIC_VECTOR (7 DOWNTO 0)
        );
        END COMPONENT;

END SYN;
```

VHDL Dual-Port RAM function

```
BEGIN
        q   <= sub_wire0(7 DOWNTO 0);

        lpm_ram_dp_component : lpm_ram_dp
        GENERIC MAP (
                lpm_width => 8,
                lpm_widthad => 8,
                lpm_indata => "REGISTERED",
                lpm_wraddress_control => "REGISTERED",
                lpm_rdaddress_control => "REGISTERED",
                lpm_outdata => "UNREGISTERED",
                lpm_hint => "USE_EAB=ON"
        )
        PORT MAP (
                rdclock => rdclock,
                wren => wren,
                wrclock => wrclock,
                data => data,
                rdaddress => rdaddress,
                wraddress => wraddress,
                q => sub_wire0
        );

END SYN;
```

Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 2)

2) **"Black-box" my_ram in Synplify Pro and synthesize the *top_ram***

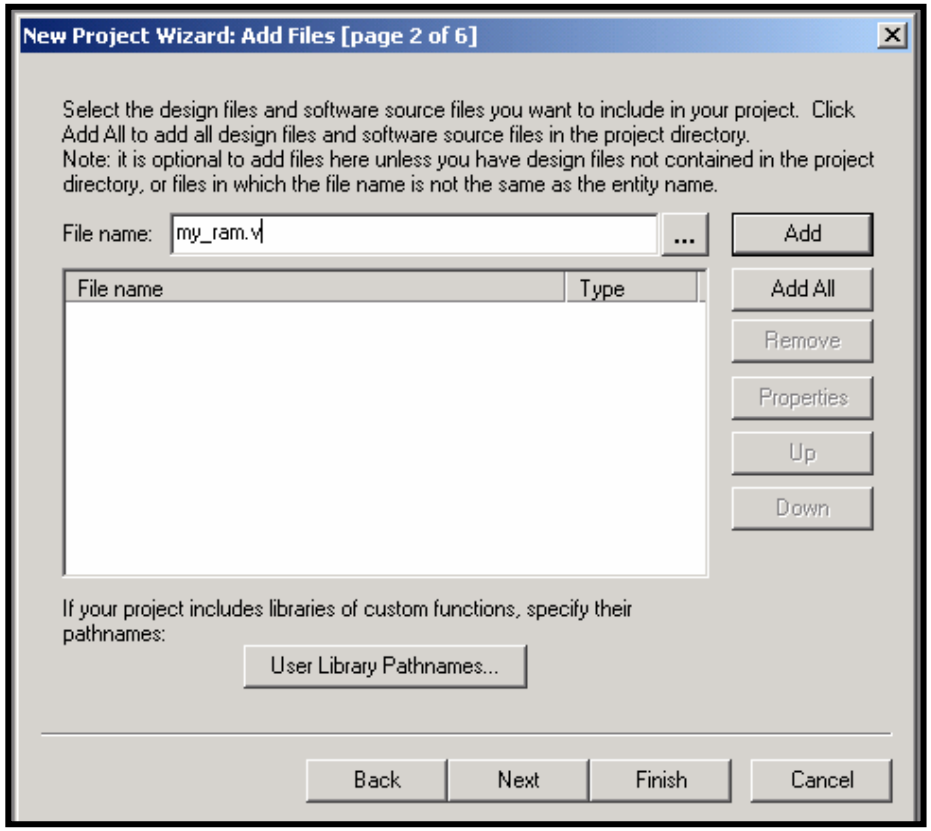
```
-- VHDL code which will call dual-port RAM
ENTITY top_ram  IS
    (top_data, top_wradd, top_rdadd        : IN STD_LOGIC_VECTOR
                                                    (7 DOWNTO 0);
     top_we, top_wrclk, top_rdclk          : IN STD_LOGIC;
     top_q                                 : OUT STD_LOGIC );
END top_ram;

ARCHITECTURE inst OF top_ram IS

COMPONENT my_ram PORT (
                data          : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
                wraddress     : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
                rdaddress     : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
                wren          : IN STD_LOGIC  := '1';
                wrclock       : IN STD_LOGIC ;
                rdclock       : IN STD_LOGIC ;
                q             : OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END COMPONENT;
ATTRIBUTE syn_black_box : BOOLEAN;
ATTRIBUTE syn_black_box OF my_ram:  COMPONENT IS TRUE;

BEGIN
    U1 : my_ram PORT MAP ( data => top_data, wraddress => top_wradd,
                           rdaddress => top_rdadd, wren => top_we,
                           wrclock => top_wrclk, rdclock => top_rdclk,
                           q => top_q);
END inst;
```

Note: DO NOT include my_ram.vhd in the Synplify Pro input file list

Copyright © 2004 Altera Corporation

Black-boxing a Dual-Port RAM (Step 3)

3) Compile the Output File in Quartus II with my_ram.v (or .vhd) in the project directory or library directory.



Copyright © 2004 Altera Corporation

Please go to Exercise 5

Copyright © 2004 Altera Corporation

Laboratory Exercise 5 Review

- Creating a MegaWizard function
- “Black-boxing” the MegaWizard function
- Synthesizing the design with “black-box”
- Compiling the design in Quartus II

Copyright © 2004 Altera Corporation

Attributes & Directives

- Information Attached to Design Objects to Control Design Analysis, Optimization & Mapping
- Use after Initial Synthesis (& Simple Settings) to Guide Synplify Pro to Better Results

- Directives
  - Source Code Only

- Attributes
  - SCOPE
  - Source Code
  - HDL Analyst Window

Note: Please See Synplify Pro Reference Manual for a More Detailed Explanation on All Directives & Attributes

Copyright © 2004 Altera Corporation

Directives

- ⑤ *translate_on/off*
- ⑤ *syn_keep*
- ⑤ *syn_black_box*

Copyright © 2004 Altera Corporation

Directives in Verilog Code

⑤ Verilog

- Add after the Design Object & before the Possible Semicolon
- Use Verilog Block or Line Comment

BLOCK COMMENT

```
module module_name (port list)  /* synthesis <directive_name> = <value> */ ;
```

LINE COMMENT

```
case (select)  // synthesis <directive_name> = <value>
```

Copyright © 2004 Altera Corporation

Directives in VHDL Code

⑤ VHDL

- Add after the Design Object Declaration
- Redefine Directive Every Time It Is Used

Define Attribute

```
<design unit declaration> ;
ATTRIBUTE <directive_name> : <data_type> ;
ATTRIBUTE <directive_name> OF <object_name> : <object_kind> IS <value> ;
```

Set Attritue & Value to Design Object

Copyright © 2004 Altera Corporation

Directive - *syn_encoding*

- Controls the Encoding of Each Individual State Register
- Overrides FSM Compiler
- Apply to State Register Declaration
- Settings
  - *sequential* , *onehot* , *gray*
  - *safe* (Forces Reset State If Invalid State is Reached)

Verilog

```
reg [7:0] current_state  /*  synthesis syn_encoding = "sequential"  */  ;
```

VHDL

```
SIGNAL current_state  :  std_logic_vector (7 DOWNTO 0);
ATTRIBUTE syn_encoding :  BOOLEAN;
ATTRIBUTE syn_encoding OF current_state : SIGNAL IS TRUE;
```

Copyright © 2004 Altera Corporation

Directive – *translate_on/off*

- Tells Synplify Pro to Ignore HDL Code Found between *translate_off* & *translate_on*
- Is Used to Ignore Simulation-Specific Code or Code Written for Another Tool
- Should Always Be Used as a Pair
- Nesting Translate Commands Is Illegal

Verilog

```
assign s1 = a & b ;
// synthesis translate_off
assign s2 = a| b ;
// synthesis translate_on
```

VHDL

```
s1 <= a AND b ;
--  synthesis translate_off
s2 <= a OR b ;
--  synthesis translate_on
```

Copyright © 2004 Altera Corporation

Directive – *syn_keep*

- Tells Synplify Pro to Maintain a Net throughout Synthesis & Optimization
- Preserve Nets for
  - Probing During Simulation
  - Maintain Certain Control Signals (Ex. Clock Enable)
- Causes a *keepbuf* Primitive to be Generated which Acts like a Placeholder
  - Does Not Go into Final Output Netlist
- Is Applied to Verilog Wires & VHDL Signals

Verilog

```
wire clock_enable /*  synthesis syn_keep  */ ;
```

VHDL

```
SIGNAL clock_enable : STD_LOGIC;
ATTRIBUTE syn_keep : BOOLEAN;
ATTRIBUTE syn_keep OF clock_enable : SIGNAL IS TRUE;
```

Copyright © 2004 Altera Corporation

Directive – *syn_black_box*

- Tells Synplify Pro to Ignore the Internal Logic of the Module/Entity
- Preserves Module/Entity Interface
- Can Be Used for
  - Altera-Specific IP, Primitives & Megafunctions
  - Functions Designed in Schematic or AHDL

Verilog

```
module global (in, out) /*  synthesis syn_black_box */  ;
input in;
output out;
endmodule
```

VHDL

```
COMPONENT global PORT (
          a_in : IN STD_LOGIC;
          a_out : OUT STD_LOGIC);
ATTRIBUTE syn_black_box :  BOOLEAN;
ATTRIBUTE syn_black_box OF global : COMPONENT IS TRUE;
```

Copyright © 2004 Altera Corporation

Other Black Box Directives

- Directives
  - black_box_tri_pins
  - black_box_pad_pin
  - syn_tpd
  - syn_tsu
  - syn_tco

- Use When Instantiating IP Cores as Black-Boxes
  - Black-Box Connectivity
  - Black-Box Timing

Copyright © 2004 Altera Corporation

Attributes

- *syn_maxfan*
- *syn_hier*
- *syn_useioff*
- *altera_io_powerup*

Copyright © 2004 Altera Corporation

Attributes in HDL Code

⑤Verlilog/VHDL

– Same Syntax as Directives

Verilog BLOCK COMMENT

```
module module_name (port list)  /* synthesis <attribute_name> = <value> */ ;
```

Verilog LINE COMMENT

```
case (select)  // synthesis <attribute_name> = <value>
```

VHDL

```
<design unit declaration> ;
ATTRIBUTE <attribute_name> : <data_type> ;
ATTRIBUTE <attribute_name> OF <object_name> : <object_kind> IS <value> ;
```

Copyright © 2004 Altera Corporation

Attributes in SCOPE Editor

- Recommended Way to Set Attributes
  - All Constraints (except Directives) in One Location
- Selecting the Object Type Limits the Object Drop-Down Menu to Only Those of That Type
- Selecting the Object Limits the Attributes Drop-Down Menu to Only Those Applicable to That Object
- Description Column Defines the Attributes Functionality

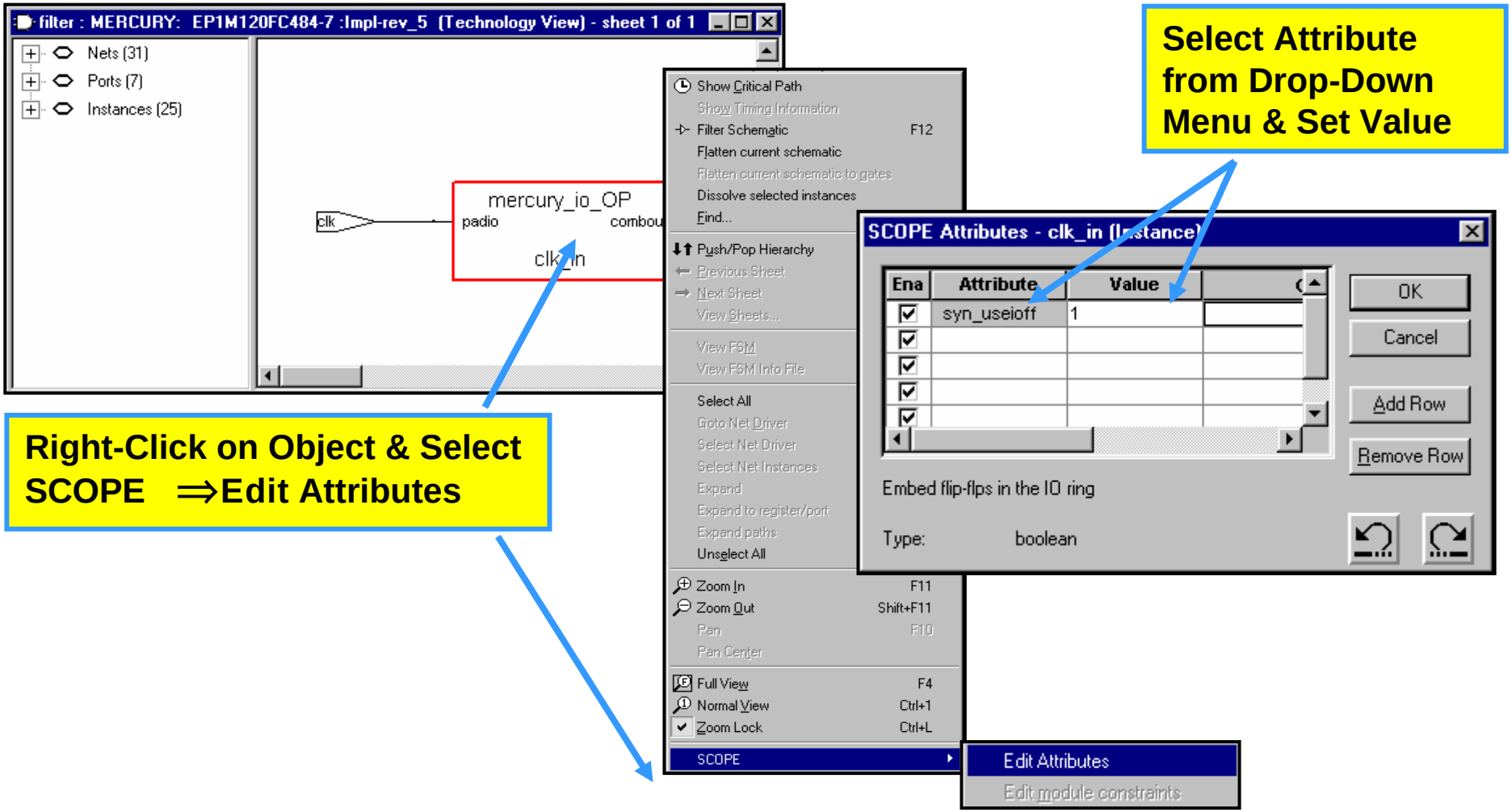
u_control.sdc (constraint File) *

	Enabled	Object Type	Object	Attribute	Value	Val Type	Description	Comment
1	☑	port	porta[7:0]	altera_chip_pin_lc	56	string	I/O pin location	
2	☑							
3	☑							
4	☑							
5	☑							
6	☑							
7	☑							
8	☑							

Clocks | Inputs/Outputs | Registers | Multi-Cycle Paths | False Paths | **Attributes** | Other

Copyright © 2004 Altera Corporation

Attributes in HDL Analyst



Copyright © 2004 Altera Corporation

Attribute – *syn_maxfan*

- Controls the maximum fanout
- Results in duplication of the Driver of the Net
- Specify Integer Value
- Is Applied to Input, Registered Output, or Net

Verilog

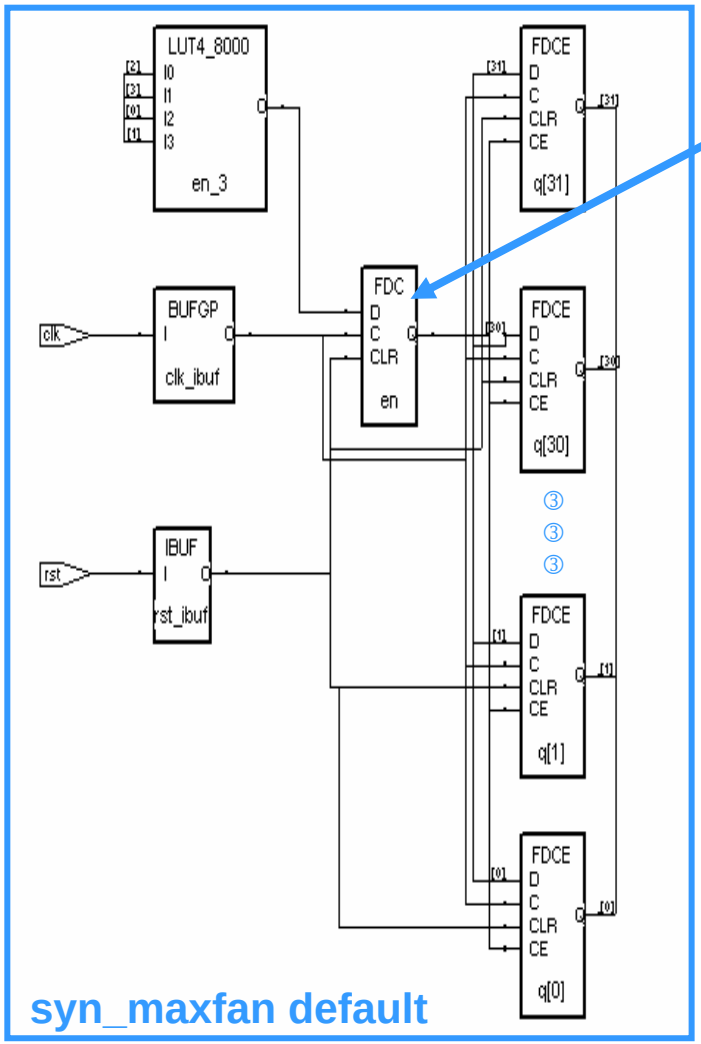
```
reg en /*  synthesis syn_maxfan = 10  */ ;
```

VHDL

```
SIGNAL en : STD_LOGIC;
ATTRIBUTE syn_maxfan : INTEGER;
ATTRIBUTE syn_maxfan OF en : SIGNAL IS 10;
```

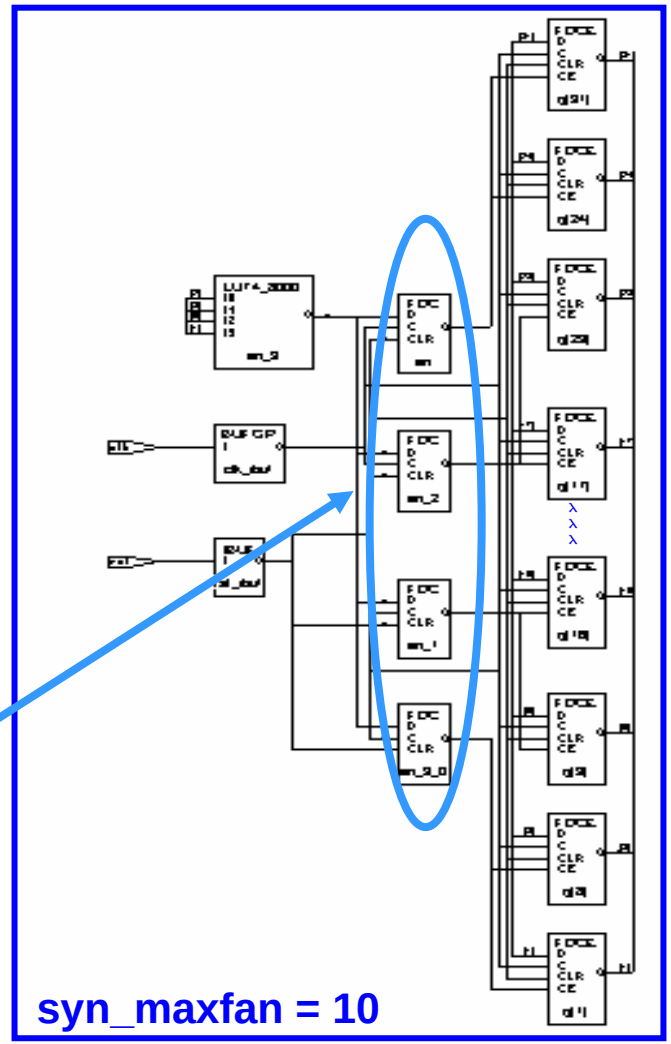
Copyright © 2004 Altera Corporation

Example: *syn_maxfan*



One Register Driving 32 Registers

Register Replicated into Four



Copyright © 2004 Altera Corporation

Attribute – *syn_hier*

- Controls the Amount of Flattening Across Instance Boundaries
- Is Applied to Modules/Architectures
- Settings
  - Soft (Default) – Flattens Current Level if Needed for Optimization
  - Flatten : Flattens All Hierarchy Levels below but Not Current Level
  - Firm – Preserves Current Level's I/O Interface but Allows Cell Packing
  - Hard – Preserves Current Level I/O Interface
  - Remove – Removes Current Level; Lower Levels Unaffected
- Can Use Multiple Settings Together

Verilog

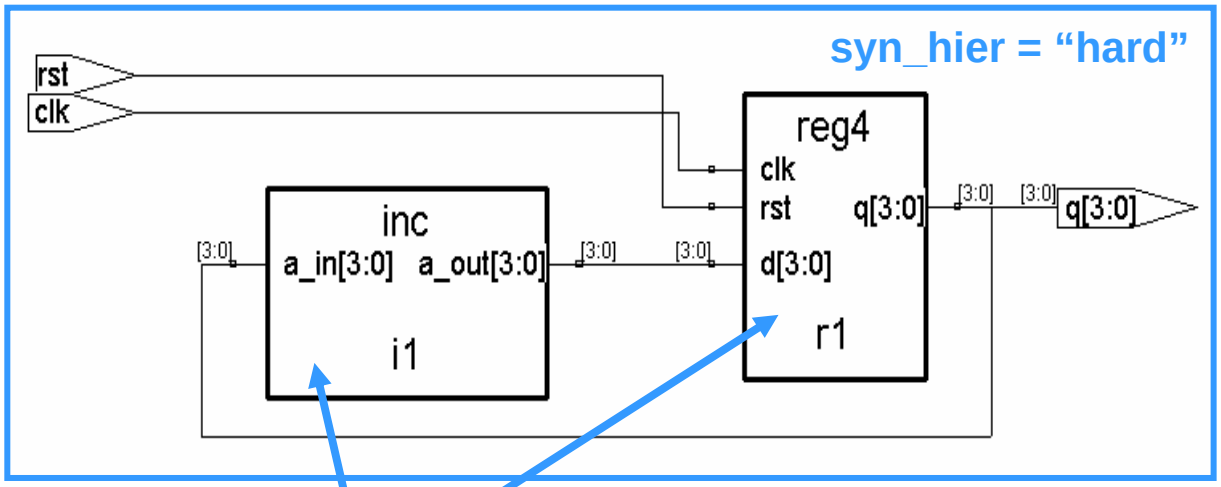
```
module reg4 (clk, rst, d, q) /*  synthesis syn_hier = "hard"  */ ;
```

VHDL

```
ARCHITECTURE rtl OF reg4 IS
ATTRIBUTE syn_hier : STRING;
ATTRIBUTE syn_hier OF rtl : STRING IS "hard";
```

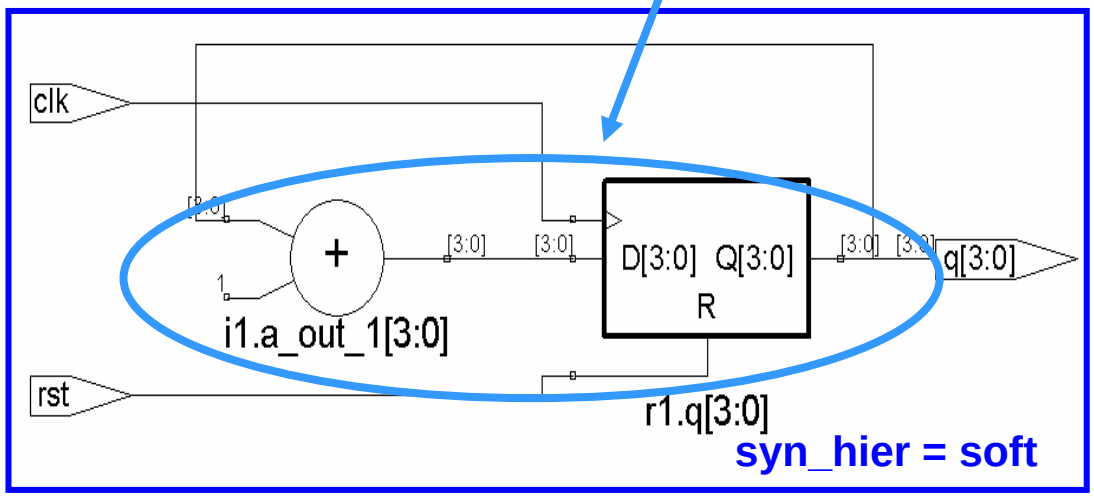
Copyright © 2004 Altera Corporation

Example: *syn_hier*



Sub-Modules/Entities inc (Incrementor) & reg4 (Registers) Maintained

Cross-Boundary Optimization Occurs & a Counter is Inferred



Copyright © 2004 Altera Corporation

Attribute – *syn_useioff*

- Enables/Disables I/O Register Packing
  - Device Must Support
- Is Applied to
  - Modules/Architectures For Global Setting
  - Individual I/O Ports as They Are Declared
- Settings
  - 1 (TRUE) : Enables Register Packing
  - 0 (FALSE) : Default

Verilog (Global Setting)

```
module reg4 (clk, rst, d, q) /*  synthesis syn_useioff = 1  */ ;
```

VHDL (Global Setting)

```
ARCHITECTURE rtl OF reg4 IS
ATTRIBUTE syn_useioff : BOOLEAN;
ATTRIBUTE syn_useioff OF reg4 : ARCHITECTURE IS TRUE;
```

Copyright © 2004 Altera Corporation

Attribute – *syn_useioff* (cont.)

Verilog (Individual I/Os)

```
module reg4 (clk, d, q);
input clk ;
input [3:0] d     /*  synthesis syn_useioff = 1  */ ;
output [3:0]  q     /*  synthesis syn_useioff = 1  */ ;
```

VHDL (Individual I/Os)

```
ENTITY reg4 IS PORT (
          clk : IN STD_LOGIC;
          d : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
          q : OUT STD_LOGIC_VECTOR (3 DOWNTO 0) );
     ATTRIBUTE syn_useioff : BOOLEAN;
     ATTRIBUTE syn_useioff OF d : SIGNAL IS TRUE;
     ATTRIBUTE syn_useioff OF q : SIGNAL IS TRUE;
END ENTITY reg4;
```

After End of Port Declaration and before End of Entity

Copyright © 2004 Altera Corporation

Attribute – *altera_io_powerup*

- Controls the Power-Up State of Device I/O Registers
- Register Automatically Packed (*syn_useioff* = 1 (TRUE))
- Applied to I/O Port Definitions
- Settings
  - Low (Default)
  - High
- I/O Register Must Not Have a Predefined Preset or Clear Condition
  - If Register Has Clear, Power-Up is Set to Low
  - If Register Has Preset, Power-Up is Set to High

Note: Please Review Device Datasheets to See if Device I/O Support This Functionality

Copyright © 2004 Altera Corporation

Attribute – *altera_io_powerup*

Verilog

```
module reg4 (clk, d, q);
input clk ;
input [3:0] d;
output [3:0]  q  /*  synthesis altera_io_powerup = “high”  */ ;
reg [3:0] q;
```

VHDL

```
ENTITY reg4 IS PORT (
          clk : IN STD_LOGIC;
          d : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
          q : OUT STD_LOGIC_VECTOR (3 DOWNTO 0) );
     ATTRIBUTE altera_io_powerup : STRING;
     ATTRIBUTE altera_io_powerup OF q : SIGNAL IS “high”;
END ENTITY reg4;
```

Copyright © 2004 Altera Corporation

Please go to Exercise 6

Copyright © 2004 Altera Corporation

Exercise 6 Review

- ⑤Setting HDL Attributes in HDL Code
- ⑤Verifying Results in both Synplify Pro & Quartus II

Copyright © 2004 Altera Corporation

Optimization Suggestions

- General Optimization
- Area Optimization
- Timing Optimization

Copyright © 2004 Altera Corporation

General Optimization

- ⑤ Remove Unnecessary Priority Structures On Speed-Critical Designs
  - Use CASE Versus IF-THEN-ELSE
- ⑤ Use Source Code that Allows for Architecture Specific Resources
  - Instantiating or Inferring Memory
  - Carry Chains
  - Cascade Chains

Copyright © 2004 Altera Corporation

Area Optimization

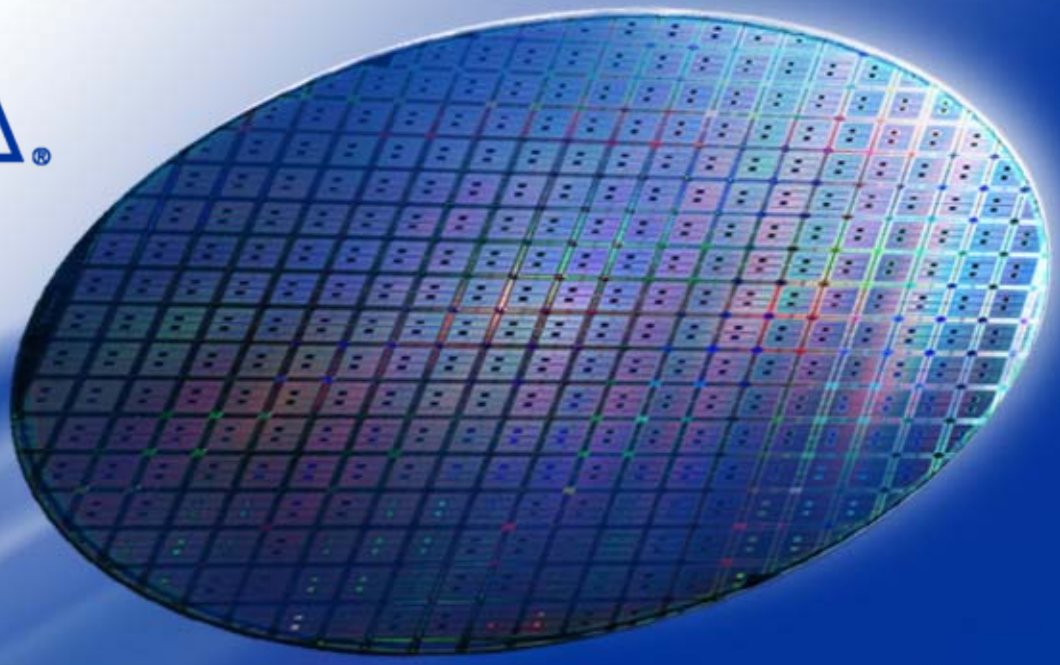
- Increase Fan-Out Limit
- Use Resource Sharing
- Use Sequential Encoding on Large FSMs
- Move Larger Logic Blocks to ESBs/EABs if Unused

Copyright © 2004 Altera Corporation

Timing Optimization

- Synplify Pro
  - Use Realistic Timing Constraints
  - Balance Fan-Out Constraint
  - Disable Resource Sharing
  - Use One-Hot Encoding on FSMs
  - Characterize the Timing on Black-Boxes
- Use NativeLink to Pass Timing to Quartus II
- Quartus II
  - Use Realistic Timing Constraints
  - Use Timing Analyzer to Locate Design Bottlenecks in Silicon
  - Use Floorplan to Evaluate if Placement Can Be Improved

Copyright © 2004 Altera Corporation



Quartus II Scripting

Copyright © 2004 Altera Corporation

Why Use Scripting?

- For Executing Applications Interactively
  - Ex. Integrate Multiple EDA Tools
- For Automating Routine Tasks
- To Use Less Memory that Graphic Interface
- To Perform Distributed Computing

Copyright © 2004 Altera Corporation

Quartus II Support

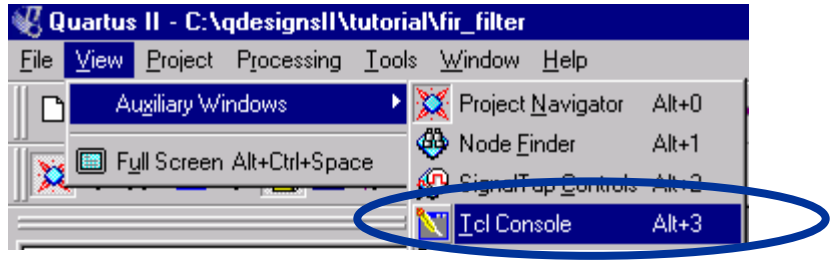
- Command-Line Scripting
- Tcl Scripting

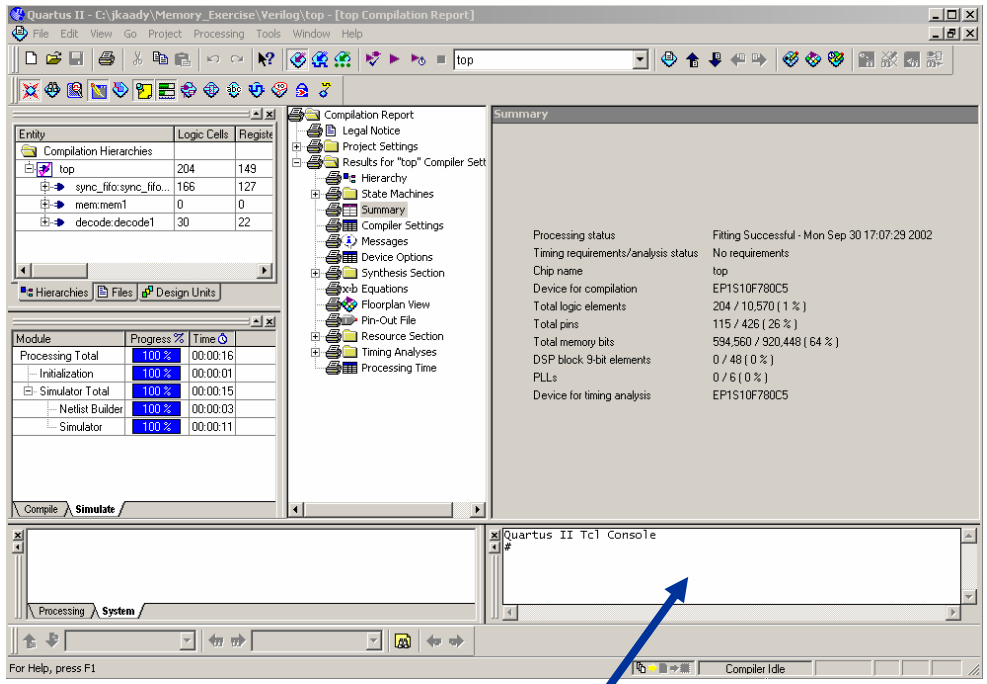
- Can Be Used Together

Copyright © 2004 Altera Corporation

Tcl/TK Console Window

- Console can be invoked using
  - View ⇒ Auxiliary Windows ⇒ Tcl/TK Console
  - Pressing the button
- Use Source Command to Run Script
  - **`source my_file.tcl`**
- Messages will appear in the Message Window under the System tab

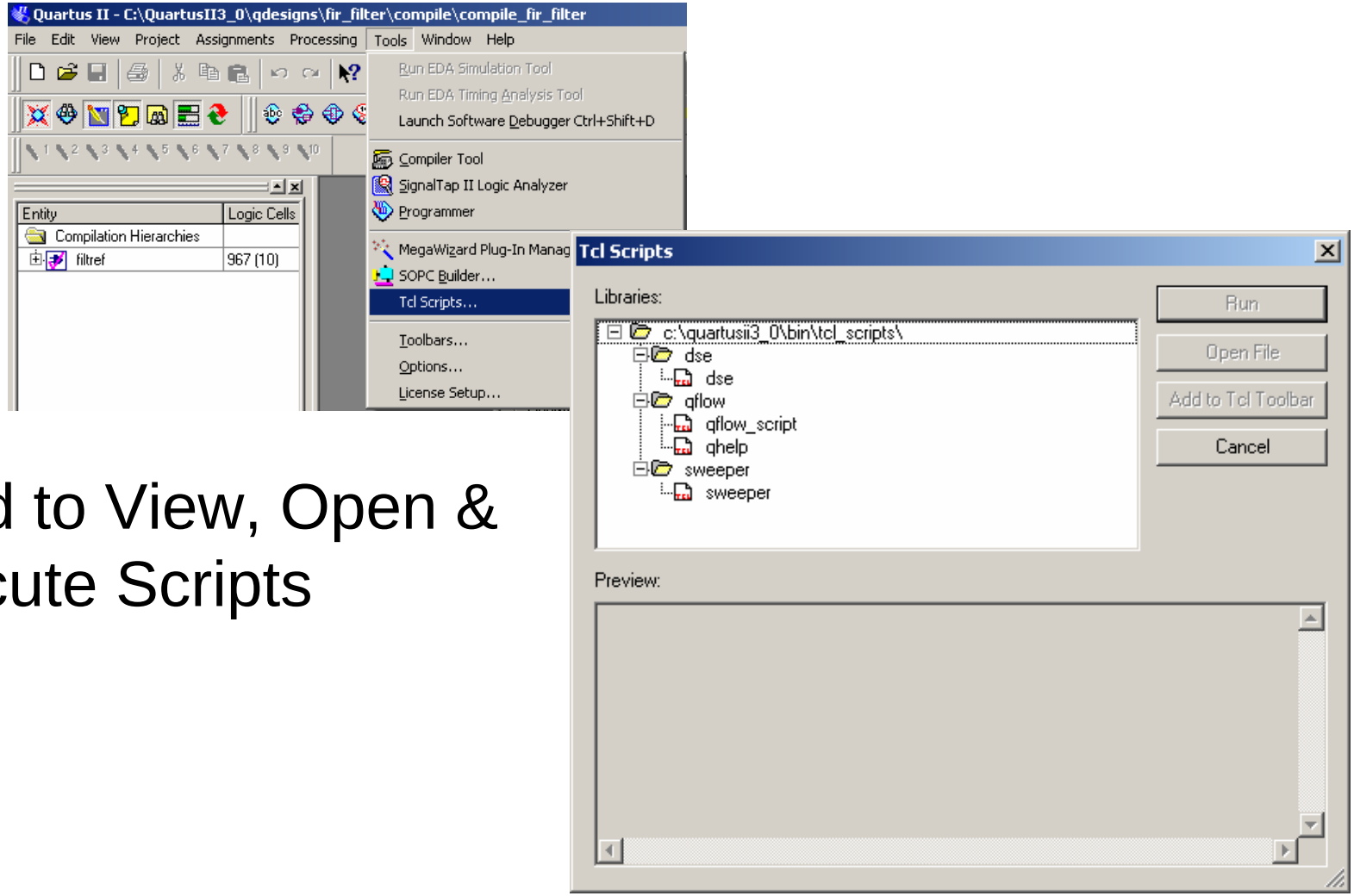




Tcl Console Window

Copyright © 2004 Altera Corporation

Tcl Scripts Dialog Box



⑤ Used to View, Open & Execute Scripts

Copyright © 2004 Altera Corporation

Command-Line Overview

- Separate Executables with Smaller Memory Footprint
  - Improves Operation on Low Powered, Low Memory Workstations
- Provides Command-Line Scripting Capability
  - Flexibility for Shell Scripts, Other Script Languages
  - Makefile Capability Similar to Other Scriptable Tools Such as Synopsys DC, Primetime, etc.
  - Create New Projects or Compile Existing Projects
- Compatible with Quartus II GUI
  - Use GUI & Command-Line Script Interchangeably

Copyright © 2004 Altera Corporation

Command-Line Executables

- ⑤ Each Executable Has Command-Line Arguments
  - Access to Common Settings, Tasks
    - ③ Device, Optimization Style, Etc.
    - ③ **`quartus_fit --part=EP1S10F484C5`**
- ⑤ Each Executable Creates Text-Based Reports
  - Success or Failure
  - Processing Information
- ⑤ Help Available for each Executable & Arguments

Copyright © 2004 Altera Corporation

Command-Line Executables in v4.0

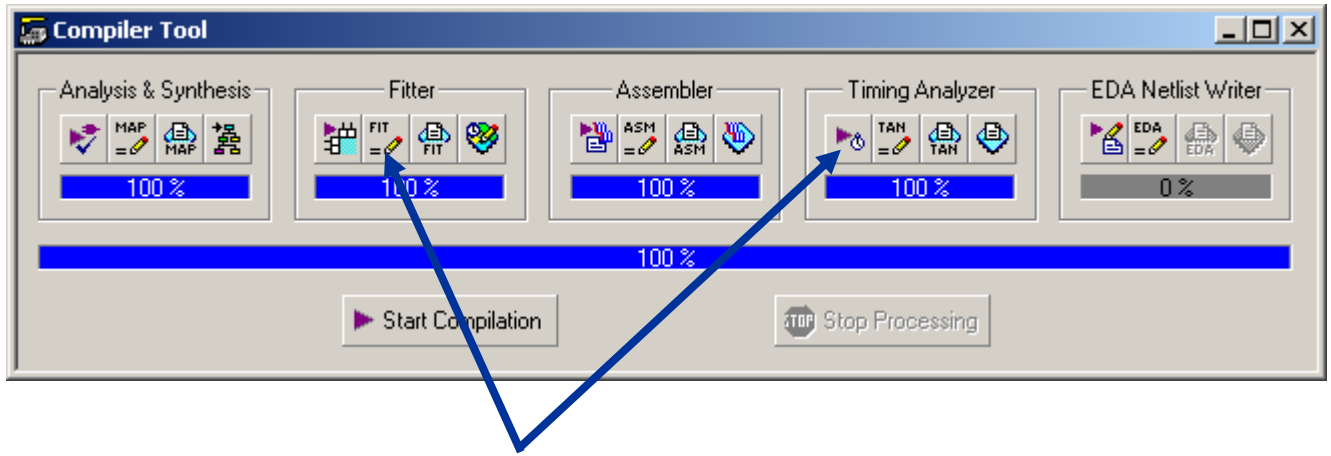
⑤ 13 Executables

– Analysis & Synthesis	(quartus_map)
– Fitter	(quartus_fit)
– Timing Analysis	(quartus_tan)
– Simulator	(quartus_sim)
– Assembler	(quartus_asm)
– Programmer	(quartus_pgm)
– Design Assistant	(quartus_drc)
– Compiler Database Interface	(quartus_cdb)
– EDA Netlist Writer	(quartus_eda)
– Convert Programming file	(quartus_cpf)
– Software Builder	(quartus_swb)
– Quartus Shell	(quartus_sh)
– SignalTap II	(quartus_stp)

Copyright © 2004 Altera Corporation

Executables Integrated into GUI

⑤Tools ⇒Compiler Tool



Buttons Open Appropriate Settings Box or Report

Copyright © 2004 Altera Corporation

quartus_sh --flow

- ⑤Performs Entire Compilation Flow
- ⑤Replaces quartus_cmd in Older Quartus II Scripts
  - Uses Less Memory
  - Use with Existing Projects for Complete Compile
  - Example:
    - ③**`quartus_sh –flow compile filter`**

Copyright © 2004 Altera Corporation

Option & Assignment Precedence

- Quartus II Gets Options & Assignments From
  1. Command-line Options
  2. Project File (.QSF, .QPF)
  3. Database Files (db directory)
  4. Software Defaults
- Options to Skip Project Files
  - --import_settings_files=off (reading settings)
  - --export_settings_files=off (writing settings)

Copyright © 2004 Altera Corporation

Sample Command-Line Script

```
# Perform a batch compile on a new project.

# Create a new project called filtref targeting a Stratix family & optimize for
#   speed
quartus_map filtref --optimize=speed --family=stratix

#Fit the design using Fastfit and a 1S10F484C5 part
quartus_fit filtref --fast_fit=on --part=ep1s10f484c5

#Create a POF for the project
quartus_asm filtref

# Perform Timing Analysis – Must be run before Generate EDA Simulation Netlist
quartus_tan filtref

# Generate Verilog EDA Simulation Netlist for NC-Sim, maintaining the design
#   hierarchy
quartus_eda filtref --simulation --tool=ncsim --format=verilog
   --maintain_design_hierarchy=on

# Rerun Timing Analysis targeting a different speedgrade
quartus_tan filtref --speed=7
```

Copyright © 2004 Altera Corporation

Getting Help

- Help Is Integrated with each Executable
  - Description, Options, Examples
- Get Help by Typing
  - **`<executable> -h`**
    - Command Description
    - Available Options
  - **`<executable> --help=<option>`**
    - Option Description
  - **`quartus_sh –qhelp`**
    - Command Browser

Copyright © 2004 Altera Corporation

Command-Line Help: <command> -h

```
C:\WINDOWS\System32\cmd.exe
C:\Documents and Settings\mmendoza>quartus_map -h
Quartus II Analysis & Synthesis
Version 4.0 Internal Build 182 12/16/2003 SJ Full Version
Copyright (C) 1991-2004 Altera Corporation

Usage:
------

quartus_map [-h | --help[=<option|topic>] | -v]
quartus_map <project name> [<options>]

Description:
------------

Quartus(R) II Analysis & Synthesis builds a single project
database that integrates all the design files in a design
entity or project hierarchy, performs logic synthesis to
minimize the logic of the design, and performs technology
mapping to implement the design logic using device
resources such as logic elements.

Options:
--------

        -f <argument file>
        -c <revision name> | --rev=<revision name>
        -l <path> | --lib_path=<path>
        --lower_priority
        --optimize=<area|speed|balanced>
        --family=<device family>
        --part=<device>
        --state_machine_encoding=<auto|minimal_bits|one_hot|user_encoded>
        --enable_register_retiming[=on|off]
        --enable_wysiwyg_resynthesis[=on|off]
        --ignore_carry_buffers[=on|off]
        --ignore_cascade_buffers[=on|off]
        --analyze_project
        --analyze_file=<design file>
        --generate_symbol=<design file>
        --generate_inc_file=<design file>
        --convert_bdf_to_verilog=<.bdf file>
        --convert_bdf_to_vhdl=<.bdf file>
        --export_settings_files[=on|off]
        --generate_functional_sim_netlist
        --source=<source file>
        --update_wysiwyg_parameters

Help Topics:
------------

        arguments
        makefiles

For more information on specific options, use --help=<option|topic>.
```

Copyright © 2004 Altera Corporation

Command-Line Option Help: <command> --help=<option>

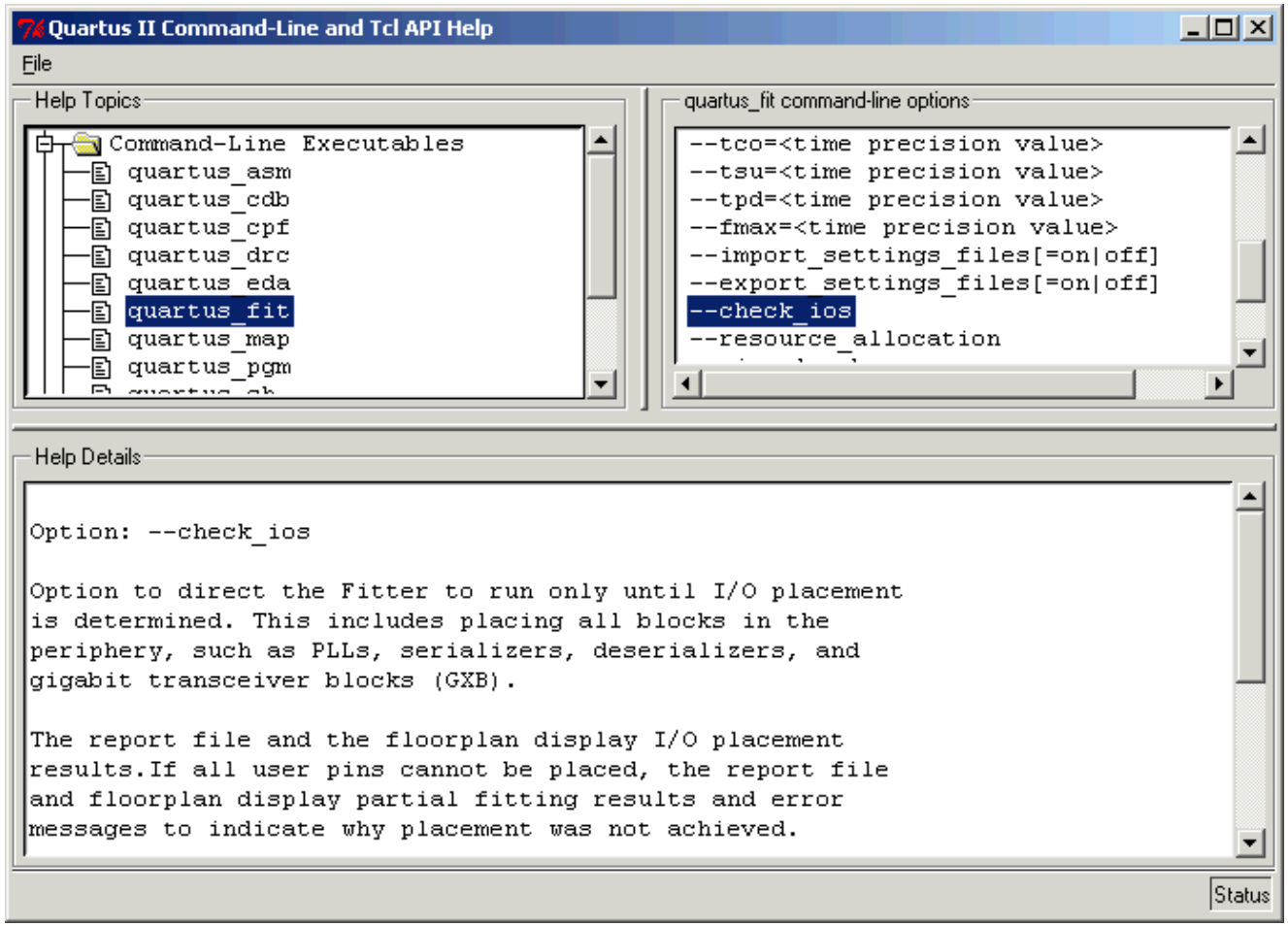
Command Prompt

```
C:\qdesigns\Phil>quartus_map --help=family

Option: --family=<device family>

Specifies the device family to use.

This option overrides the settings specified in the
compiler settings files.  Refer to the help topic on
"--csf" option for more details on compiler settings files.


C:\qdesigns\Phil>
```

Copyright © 2004 Altera Corporation

Help: quartus_sh --qhelp

⑤ Command-Line & Tcl API Help Browser



Copyright © 2004 Altera Corporation

Tcl API Overview

- New Tcl API Improves Ease of Use
  - Align with Major EDA Vendor Solutions
  - Similar Format to Synopsys SDC Used by PrimeTime & Design Compiler
- Reduce Memory Requirements
- Commands Are Grouped in Tcl Packages
  - Loaded on Demand
  - Reduces Runtime Memory of Executable

Copyright © 2004 Altera Corporation

Tcl/Tk Basics

- Tcl = Tool Command Language
  - Tk Is an Extension of Tcl, Used for GUIs
- Works on PCs & Workstations
- Interpreted Programming Language
  - Scripts Do Not Need to Be Compiled
- Quartus II Uses Tcl/Tk Supplied by Scriptics
- Quartus II Contains a Tcl Text Editor with Syntax Coloring

Copyright © 2004 Altera Corporation

Tcl Syntax

- Tcl is Case Sensitive
- Basic Syntax
  - command arg1 arg2 arg3 …
  - Example : **`set var 3`**

Sets variable "var" equal to 3

Name preceded by "$" is replaced with value of name

- Variable Substitution
  - **`puts "The variable value is $var"`**
  - Writes "The variable value is 3" to the Message Window

Command within " [ ] " is replaced with results of command

- Command Substitution
  - **`puts "The variable value is [expr $var + $var]"`**
  - Writes "The variable value is 6" to the Message Window

Copyright © 2004 Altera Corporation

Executable Support for Tcl API

- Four Executables Are Also Tcl Interpreters
  - quartus_sh, quartus_tan, quartus_sim, quartus_cdb
- Run a Specific Tcl Script
  - **`quartus_sh -t <script file>`**
- Run a Particular Tcl Command
  - **`quartus_sh --tcl_eval <Tcl command>`**
- Open an Interactive Tcl Shell (Similar to PrimeTime)
  - **`quartus_sh -s`**

Copyright © 2004 Altera Corporation

Executable Support for Tcl API

⑤Uses for Tcl with each Executable

- quartus_sh
  - ③Make Project Settings, Do Basic Reporting
- quartus_tan
  - ③Perform Basic & Advanced Timing Analysis
- quartus_cdb
  - ③Do Back Annotation, LogicLock, Chip Editor
- quartus_sim
  - ③Perform Simulations

⑤Different Packages Available in Each Shell

Copyright © 2004 Altera Corporation

Executable Support for Tcl API

- When Do I Use Tcl Interpreter Functionality?
  - Use -t, --tcl_eval, or -s Switches Only if Processing Tcl Commands or Scripts
  - Running Executables with Other Command-Line Options Is Like Running It From GUI
- Tcl Interpreter Executables Align with Tcl Package Functionality

Copyright © 2004 Altera Corporation

Tcl Packages

- 13 Packages
  - Project Creation, Assignments ::quartus::project
  - Device Information ::quartus::device
  - Advanced Device Information ::quartus::advanced_device
  - Simple Compiler Flows ::quartus::flow
  - Compute, Report Timing Paths ::quartus::timing
  - Advanced Timing Netlist Info ::quartus::advanced_timing
  - Perform Simulations ::quartus::simulator
  - Access Report Data ::quartus::report
  - List Paths Functionality ::quartus::timing_report
  - Back Annotation ::quartus::backannotate
  - Manipulate LogicLock Regions ::quartus::logiclock
  - Access Chip Editor Functions ::quartus::chip_editor
  - Miscellaneous Commands ::quartus::misc
- Some Packages Must Be Loaded Manually Before Use
  - See Appendix

Copyright © 2004 Altera Corporation

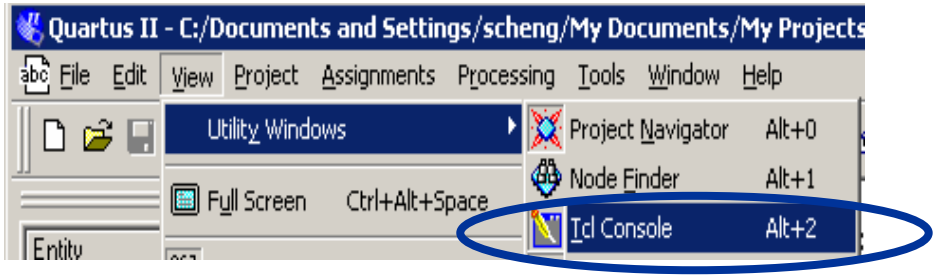
Running a Quartus II Tcl Script

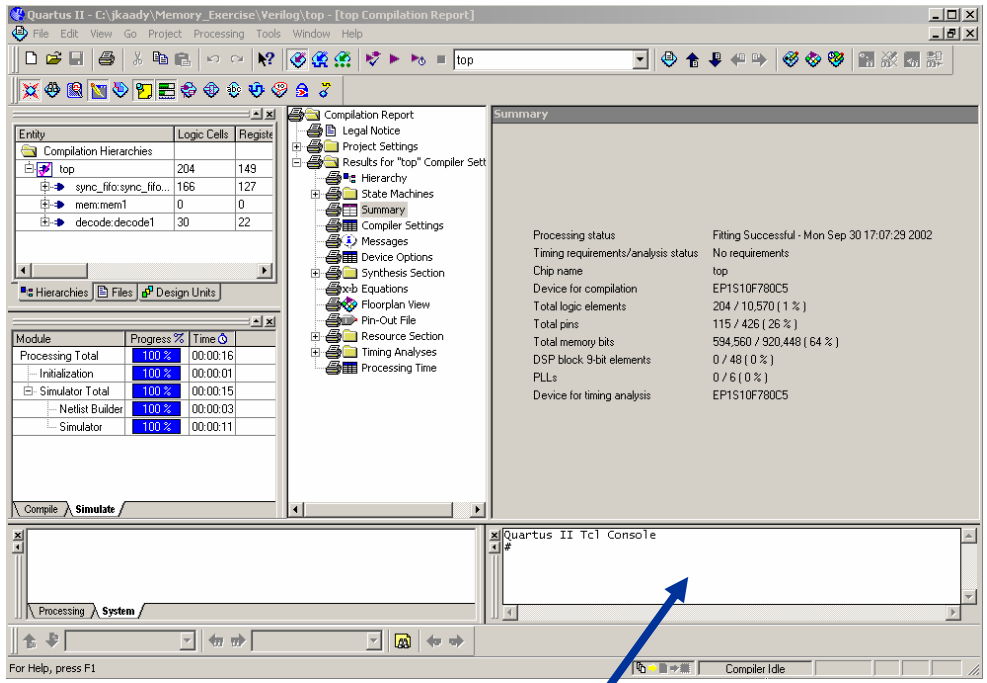
1) Tcl Console Window
2) Tcl Scripts Dialog Box
3) Command Line Arguments

Copyright © 2004 Altera Corporation

Tcl/TK Console Window

- Console can be invoked using
  - View ⇒Auxiliary Windows ⇒ Tcl/TK Console
  - Pressing the button
- Use Source Command to Run Script
  - **`source my_file.tcl`**
- Messages will appear in the Message Window under the System tab

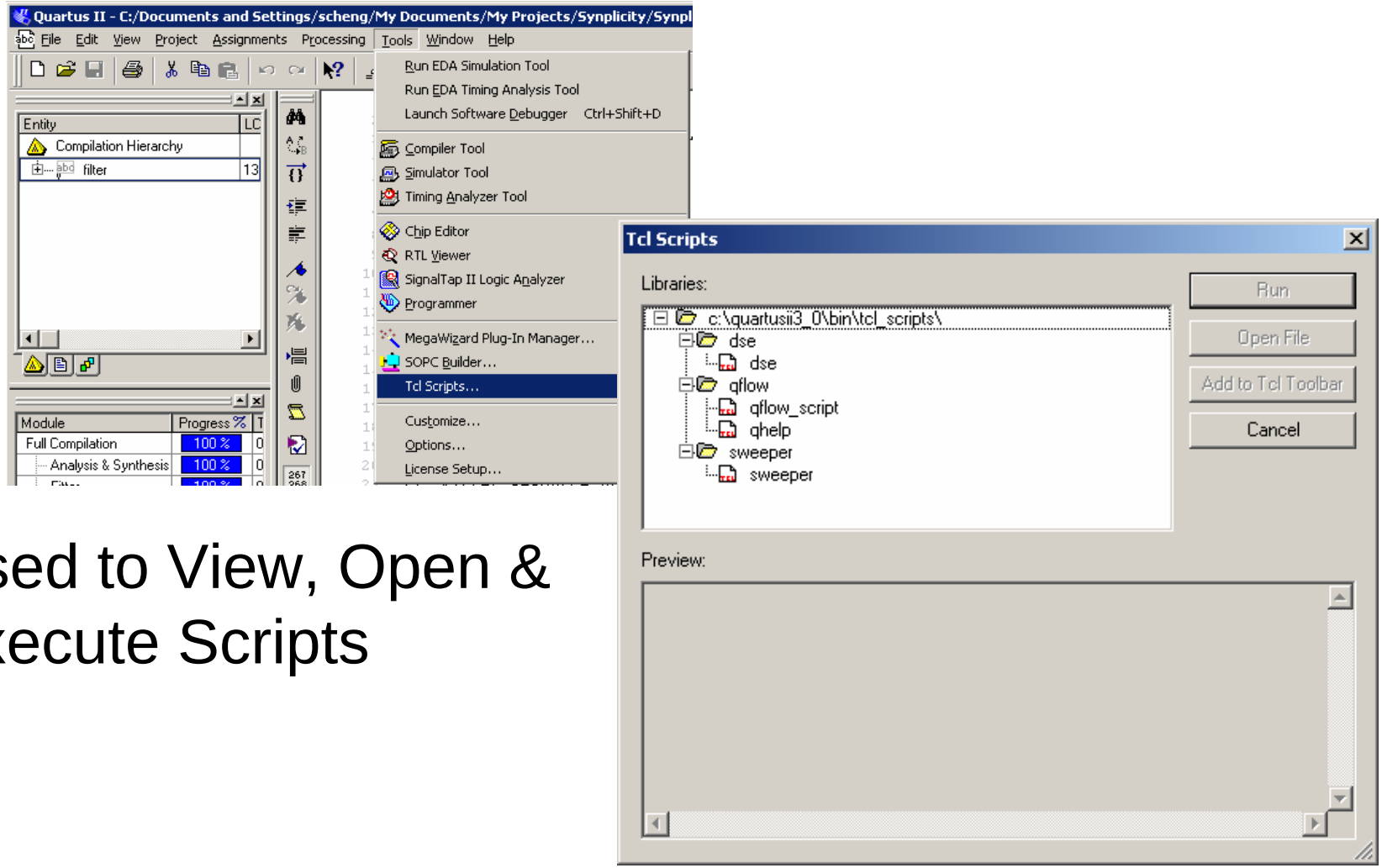




Tcl Console Window

Copyright © 2004 Altera Corporation

Tcl Scripts Dialog Box



⑤ Used to View, Open & Execute Scripts

Copyright © 2004 Altera Corporation

Old & New API Comparison

- Old Tcl API Syntax
  - cmp add_assignment “” “” “” FAMILY apex20ke
- New Tcl API Syntax
  - set_global_assignment -name FAMILY apex20ke
- New Tcl API
  - Don’t Need Knowledge of Constraint File Type
  - If User Gives Incorrect Arguments, Message Displayed Explaining Correct Usage & What Arguments are Supported

Copyright © 2004 Altera Corporation

Old & New API Comparison

- 4.0 API not Compatible with Old API
- Old API Supported by quartus_cmd
- Recommend New Scripts Target New API
  - Existing Scripts Should Be Rewritten
  - No Development on Old API
  - No Conversion Utility

Copyright © 2004 Altera Corporation

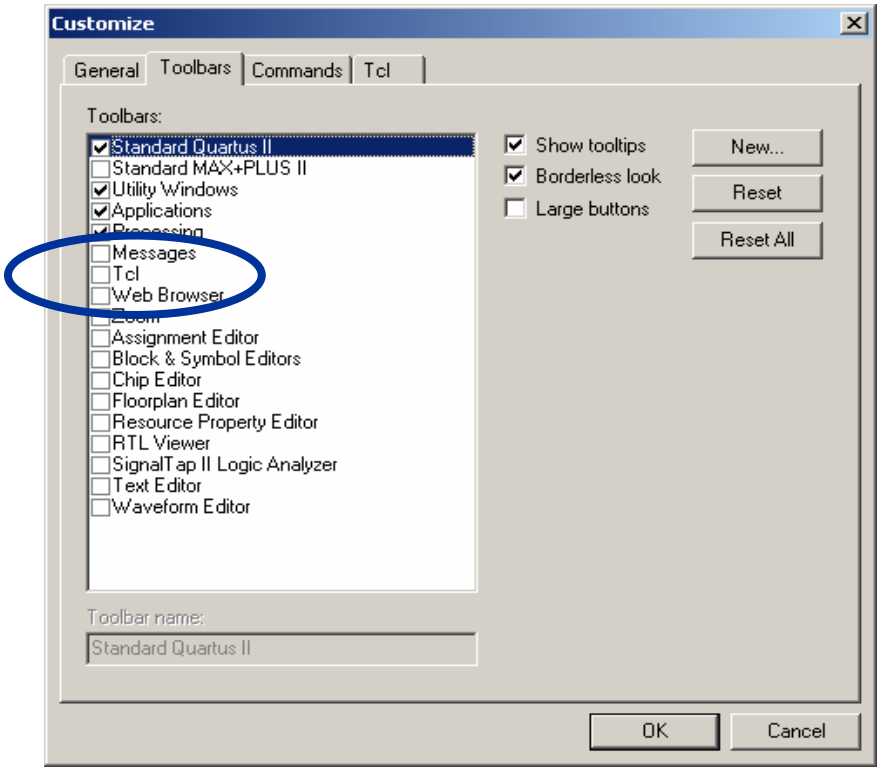
Sample Tcl Script

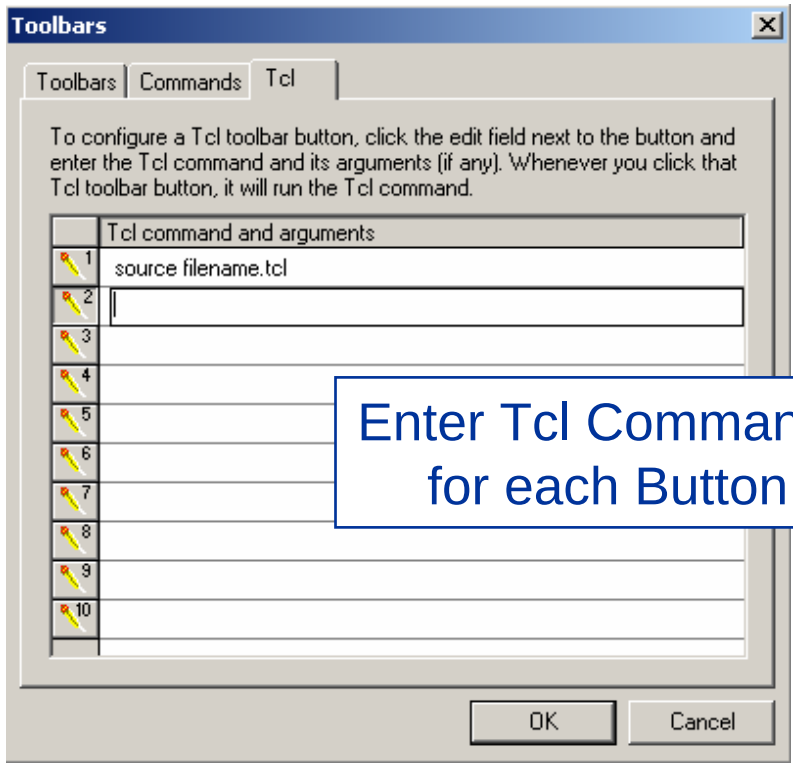
```
# Set the project name to filtref
set project_name filtref

# Open the Project. If it does not already
#   exist, create it
if [catch {project_open $project_name}]
    {project_new $project_name}

# Set Family
set_global_assignment -name family Stratix

# Set Device
set_global_assignment -name device
    ep1s10f484c5

# Optimize for speed
set_global_assignment -name
    optimization_technique speed

# Turn-on Fastfit
set_global_assignment -name
    fast_fit_compilation on

# Generate a NC-Sim Verilog simulation
#   Netlist
set_global_assignment -name
    eda_simulation_tool ncsim_verilog
```

```
# Generate a NC-Sim Verilog simulation
#   Netlist
set_global_assignment -name
    eda_simulation_tool ncsim_verilog

# Create an FMAX=50MHz assignment called
#   clk1 to pin clk
create_base_clock -fmax 50MHz -target clk
    clk1

# Create a pin assignment on pin clk
set_location_assignment -to clk Pin_AA14

# One way
# Always write the assignments to the
#   constraint files before doing a system
#   call. Else, stand-alone files will not
#   pick up the assignments
export_assignments
qexec quartus_map
qexec quartus_fit

# Better way -  FLOW will export_assignments
#   automatically. FLOW will also take care
#   of Smart Recompile
#   execute_flow -compile

# Close Project
project_close
```

Copyright © 2004 Altera Corporation

Tcl Toolbar





Enter Tcl Commands for each Button

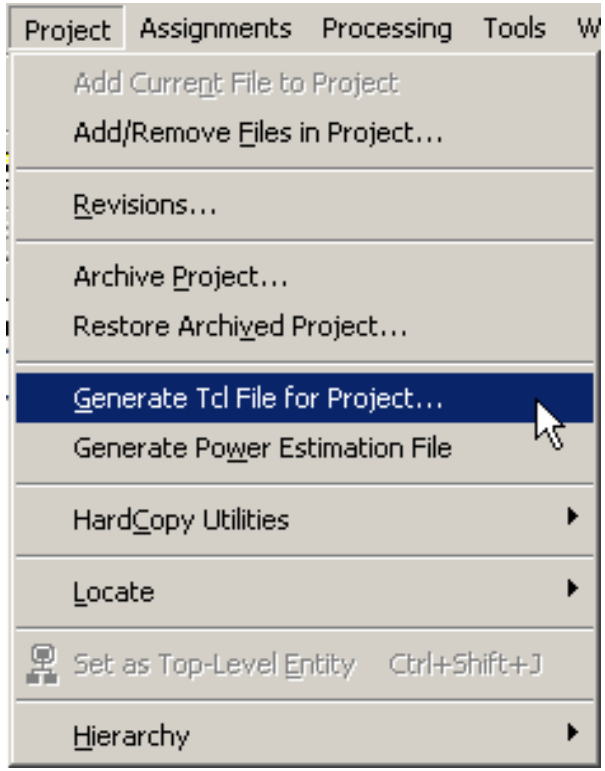
Copyright © 2004 Altera Corporation

Generate TCL File

⑤ Generate a TCL Script File for a Project

- GUI
  - ③ Project ⇒Generate TCLFile
- TCL command

```
generate_tcl_file <name>
```



Copyright © 2004 Altera Corporation

Quartus II API

⑤ Quartus II Help Lists the Tcl Commands

Quartus II Help Version 4.0

Back Forward Stop Refresh Home Print

Contents Index Search

Type in the keyword to find:

Tcl API functions for quartus_cmd

Tcl API functions
Tcl API functions for quartus_cmd
Tcl command-line mode
overview topics
running Tcl scripts in
Tcl commands
API syntax
inserting templates into Tcl Script File (.tcl)
list of API functions
overview topics
running
Tcl Console command
Tcl Console window (see Quartus II Tcl Cor
Tcl Script File (.tcl)
(see also Tcl scripts)
adding to Tcl toolbar
general description
generating
inserting templates
opening
overview topics
running
Tcl scripting, overview topics
Tcl scripts
(see also Tcl Script File (.tcl))
help on objects visible in the window
image of
overview topics
running
Tcl Scripts command
Tcl tab (Customize command)
Tcl test benches, general description
Tcl toolbar buttons, configuring
tcl, general description
tco
assignments for
general description
listing timing paths for

Display

API Functions for Tcl for quartus_cmd

See Also

The following functions are part of the Quartus® II software Application Programming Interface (API) for **quartus_cmd**. You can use these functions as Tcl commands. Click the function name to see the syntax for and a description of the function.

Tcl API functions for **quartus_cmd** have been retained in the Quartus II software for backwards compatibility only. Altera® recommends that you use the Tcl API functions for **quartus_sh**, **quartus_tan**, **quartus_cdb**, and **quartus_sim** for any new Tcl scripts that you want to create.

This list contains the following sections:

- Project Control Functions
  - Project (project) Functions
  - Compiler (cmp) Functions
    - Timing Analysis (cmp) Functions
    - LogicLock™ (cmp) Functions
  - Simulator (sim) Functions
  - Software Build (swb) Functions
- System-level Debugging (sld) Functions
- Device Database (device) Functions
- Source Code Control (scc) Functions
- Global Functions
- Synopsys® Design Constraints Functions

Project Control Functions

Project (project) Functions

Copyright © 2004 Altera Corporation

Getting Help

- Help Is Integrated in the Tcl Shell
  - Description, Commands, Examples
- Get Help in Tcl Shell by Typing
  - help -pkg <package>
  - <command> -h (short help)
  - <command> -long_help *or* help -cmd <command>
  - quartus_sh --qhelp

Copyright © 2004 Altera Corporation

Tcl API Short Help

⑤ Shows Usage & Brief Description About an Available Tcl Command

Command Prompt - quartus_sh -s

```
tcl> get_part_list -h
------------------------------------------------------------------------------

Usage: get_part_list [-h | -help] [-long_help] [-family <value>] [-device <value>] [-package <value>] [-pin_count
] [-speed_grade <value>]

        -h | -help: Short help
        -long_help: Long help with examples and possible return values

        -family <value>: Match family name
        -device <value>: Match device name
        -package <value>: Match package name
        -pin_count <value>: Match pin count
        -speed_grade <value>: Match speed grade

------------
Tcl Package:
------------

Belongs to ::quartus::device package.

------------
Description:
------------

Filters through the specified arguments and displays the resultant
list of available part(s).
Examples:
        - Return a list of all supported parts
                get_part_list
        - Return a list of all supported parts for Cyclone
                get_part_list -family Cyclone
        - Return a list of all supported parts with the FBGA package and 780 pins
                get_part_list -package fbga -pin_count 780

------------------------------------------------------------------------------

tcl> _
```

Copyright © 2004 Altera Corporation

Tcl API Long Help

- Shows Usage & Brief Description About an Available Tcl Command
- Shows Examples & List of Possible Return Values
  - <Tcl Command> -long_help *or* help -cmd <Tcl Command>

Command Prompt - quartus_sh -s

```
tcl> get_part_list -long_help
--------------------------------------------------------------------------------

Usage: get_part_list [-h | -help] [-long_help] [-family <value>] [-device <value>] [-package <value>] [-pin_count
] [-speed_grade <value>]

        -h | -help: Short help
        -long_help: Long help with examples and possible return values

        -family <value>: Match family name
        -device <value>: Match device name
        -package <value>: Match package name
        -pin_count <value>: Match pin count
        -speed_grade <value>: Match speed grade

-------------
Tcl Package:
-------------

Belongs to ::quartus::device package.

-------------
Description:
-------------

Filters through the specified arguments and displays the resultant
list of available part(s).
Examples:
        - Return a list of all supported parts
                get_part_list
        - Return a list of all supported parts for Cyclone
                get_part_list -family Cyclone
        - Return a list of all supported parts with the FBGA package and 780 pins
                get_part_list -package fbga -pin_count 780

---------------
Example Usage:
---------------

get_part_list -family Stratix

--------------
Return Value:
--------------

Code Name       Code    String Return
---------       ----    -------------
TCL_OK          0       INFO: Operation successful

--------------------------------------------------------------------------------

tcl> _
```

Copyright © 2004 Altera Corporation

Command-Line & Tcl Scripting

⑤Use Command-Line Scripting for

- Compiling Existing Projects
- Common Global Assignments

⑤Use Tcl Scripting for

- Custom Timing Analysis Reports
- Any Assignments or Settings
  - ③Pin Assignments, for Example
- Power Users

Copyright © 2004 Altera Corporation

Command-Line & Tcl Scripting

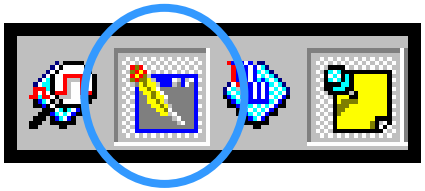
- Mix-and-Match for Complete Solution
- Example
  - Synthesize Design Files with quartus_map
  - Assign Pins with a Tcl Script
  - Place & Route with quartus_fit
  - Analyze Timing with quartus_tan
  - Create Custom Timing Report with a Tcl Script

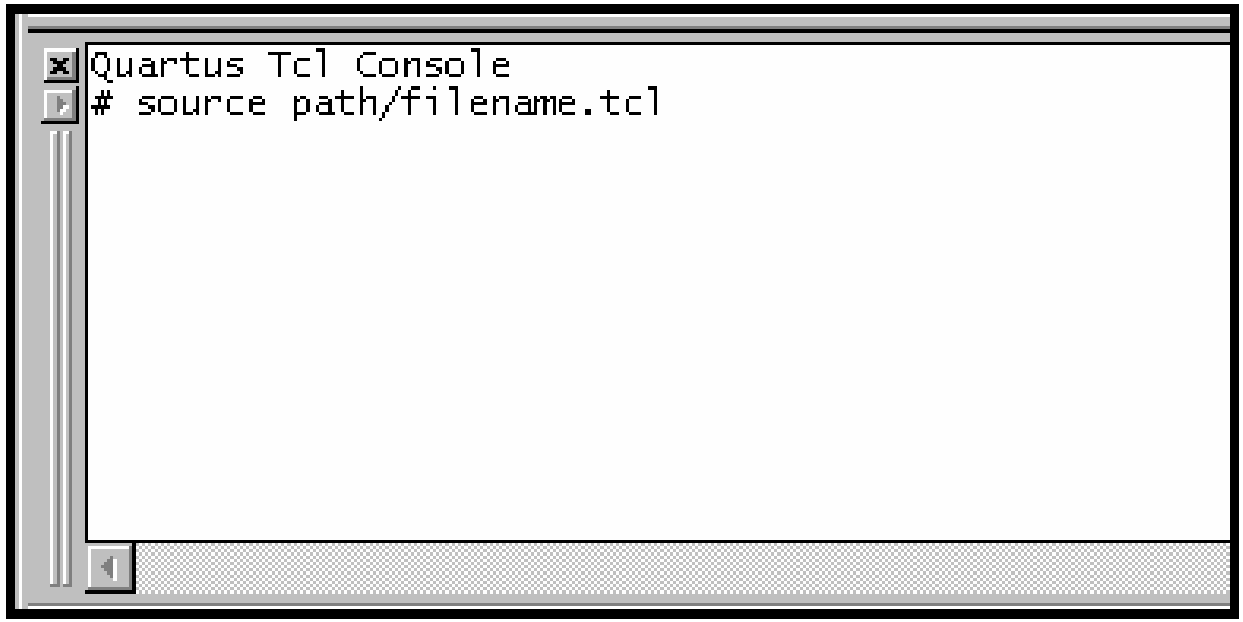
Copyright © 2004 Altera Corporation

Running Script Files in Quartus

⑤Quartus

– Tcl console source path/filename.tcl

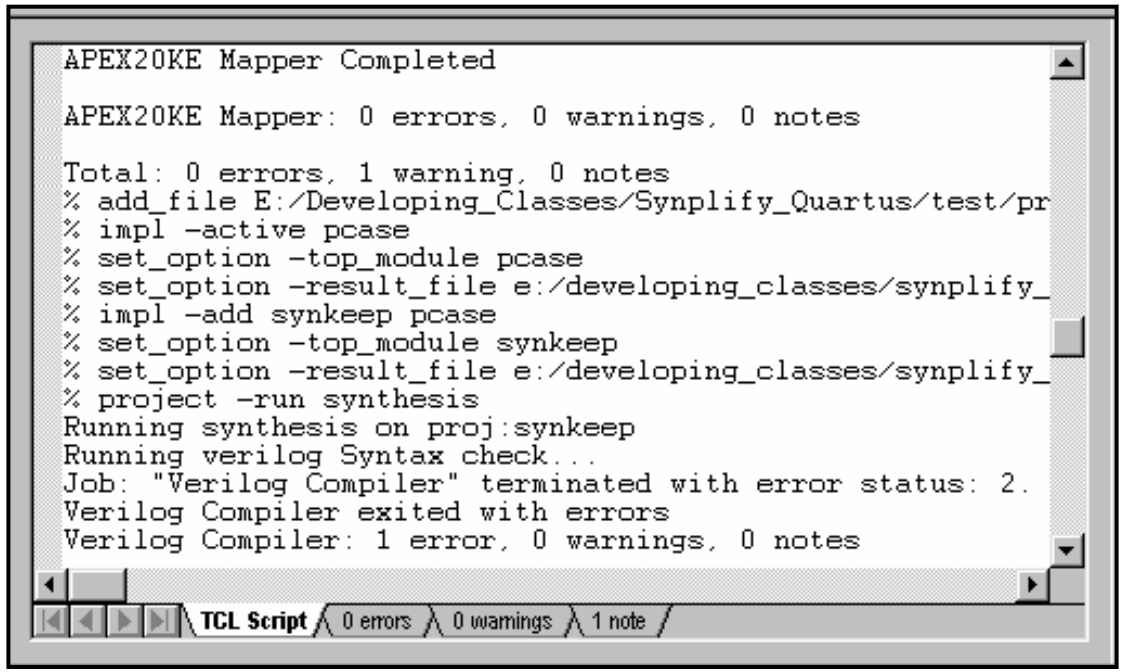




Copyright © 2004 Altera Corporation

Scripting in Synplify Pro

While performing functions like creating a project, setting synthesis options, & executing commands in Synplify Pro, the TCL Script Window is Recording the Tcl Equivalent Commands. These can be copied into a Tcl file (<filename.tcl>) for future automation.



Copyright © 2004 Altera Corporation

Recording Tcl Commands

- Records Tcl Commands into a File as You Work in Synplify Pro
- *recording*
  - -on 0 | 1
    - Turns Recording Session On or Off
  - -file <logfilename>.tcl
    - Returns the Name of the Target Log File (If No Log File is Specified)
    - Sets the Name of the Target Log File
  - -save <logfilename>.tcl
    - Updates the Current History Log
    - Save History to Log File if File Name is Specified

Copyright © 2004 Altera Corporation

Creating a New Tcl Synthesis Script

1. Open a Text File
2. Create or Open a Project
   - project –new <directory_path\project_name>
   - project –load <directory_path\project_name>
3. Add Files
   - add_file –vhdl <filename>.vhd
   - add_file –verilog <filename.v
   - add_file –constraint <filename>.sdc

Copyright © 2004 Altera Corporation

Creating a New Tcl Synthesis Script

4. Set Design Synthesis Options & Output
   - set_option
     - Ex. Set option –technology Mercury
   - project –result_file <filename>
   - project –log_file <filename>
5. Set the File & Run Options
   - project –save
   - project –run synthesis

Copyright © 2004 Altera Corporation

Sample TCL File

```
# 2) Open New Project
project -new

# 3) Add Files
add_file -vhdl file1.vhd
add_file -vhdl file2.vhd
add_file -vhdl top_level.vhd
add_file -constraint top_level.sdc

# 4)  Set Design Syntheis Options
#          And Output
set_option -technology APEX20KC
set_option -part EP20K100C
set_option -speed_grade -7
set_option -frequency 50.0
```

```
# 4) [cont.]
set_option -result_file
top_level_imp1.vqm
set_option -log_file top_level_imp1.srr

# 5) Set File & Run Options
project -save top_level.prj
project -run synthesis
```

Copyright © 2004 Altera Corporation

Sample SDC File

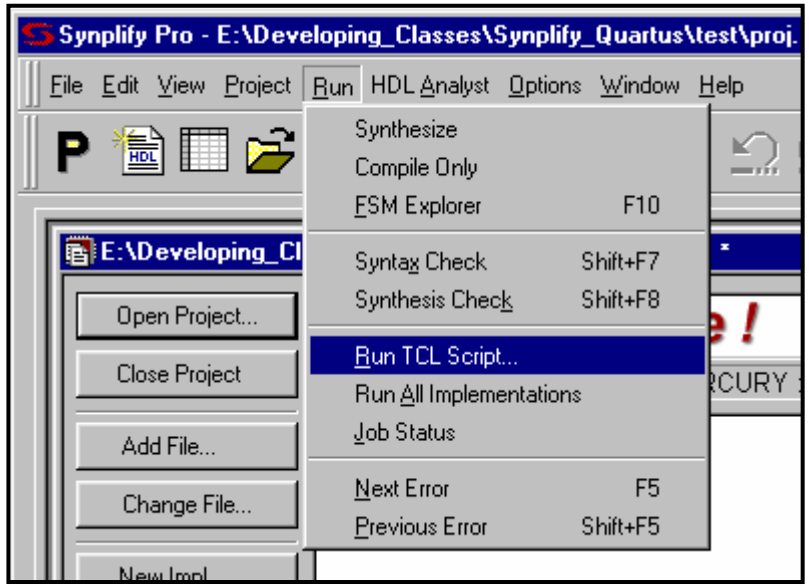
```
# Add Timing Constraints
define_input_delay -default 100
define_output_delay -default 100
define_input_delay {DATA0 [7:0]} 10

# Put a scalar signal on a pin location 159
define_attribute SEL altera_chip_pin_lc "163"

# Assign pin locations to all bits of a bus
define_attribute {DATA0[7:0]} altera_chip_pin_lc “40, 41, 42, 43, 46, 47, 48, 49"

# Assign pin locations to 2 bits of an 8 bit bus
define_attribute {DATA1[3]} altera_chip_pin_lc "113"
define_attribute {DATA1[7]} altera_chip_pin_lc "119"
```

Copyright © 2004 Altera Corporation

Running Script Files

⑤Synplify Pro

– Run -> Run TCL Script



Copyright © 2004 Altera Corporation

Summary

- Introduction to Altera & Altera Devices
- PLD Design Flow
- Quartus II / Synplify Pro Overview
- Design & Synthesis with Synplify Pro
- Quartus II Project Management
- Quartus II Compilation
- NativeLink
- Optimizing Designs with Synplify Pro & Quartus II
- TCL Scripting

Copyright © 2004 Altera Corporation

Altera Technical Support

- Reference Quartus II On-Line Help
- Consult Altera Applications (Factory Applications Engineers)
  - Hotline: (800) 800-EPLD (7:00 a.m. - 5:00 p.m. PST)
  - MySupport: http://www.altera.com/mysupport
- Field Applications Engineers: Contact Your Local Altera Sales Office
- Receive Literature by Mail: (888) 3-ALTERA
- FTP: ftp.altera.com
- World-Wide Web: http://www.altera.com
  - Use Solutions to Search for Answers to Technical Problems
  - View Design Examples

Copyright © 2004 Altera Corporation

Learn More Through Technical Training

Instructor-Led Training	On-Line Training
With Altera's instructor-led training courses, you can: *Listen to a lecture from an Altera technical training engineer (instructor) *Complete hands-on exercises with guidance from an Altera instructor *Ask questions and receive real-time answers from an Altera instructor *Each instructor-led class is one day in length (8 working hours).	With Altera's on-line training courses, you can: *Take a course at any time that is convenient for you *Take a course from the comfort of your home or office (no need to travel as with instructor-led courses) *Each on-line course will take approximately 2-3 hours to complete.

www.altera.com/training

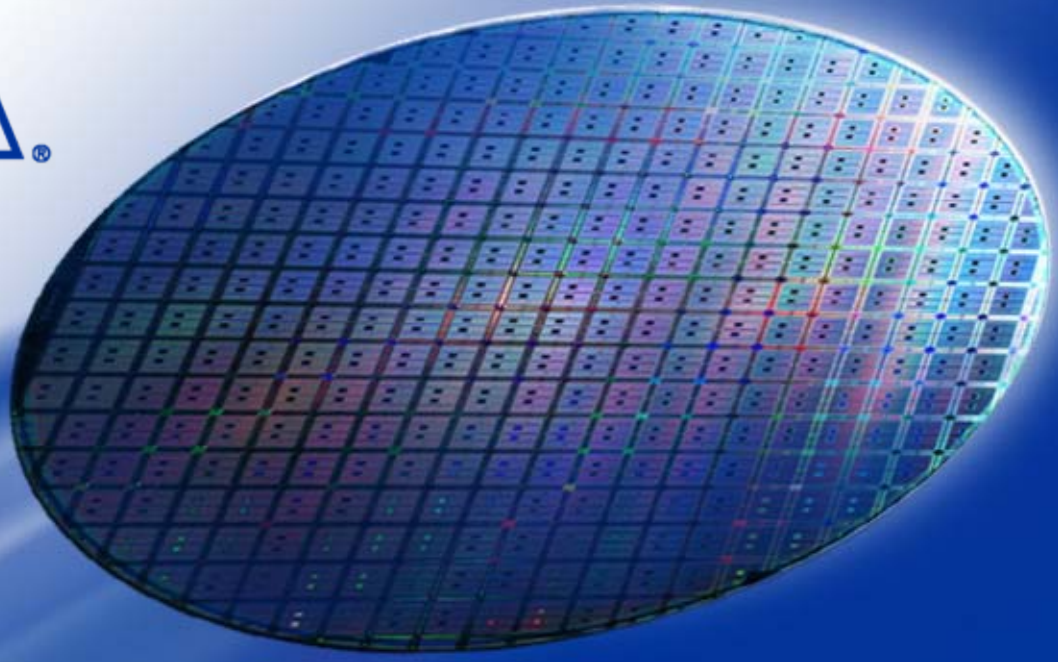
View Training Class Schedule & Register for a Class

Copyright © 2004 Altera Corporation

Synplicity Technical Support

- ⑤ Synplify Online Support[SOS] and Synplify Newsgroup
  - ③http://www.synplicity.com/support/logon_news/log_news.html
  - ③news://news.synplicity.com/Synplicity.Synplify
- ⑤ Synplify First Level Support
  - ③Solutions to Common Designer Issues & Questions
  - ③Can be accessed from S.O.S
- ⑤ Send email to *support@synplicity.com*
- ⑤ Call the Technical Support Hotline at (408) 215-6000

Copyright © 2004 Altera Corporation



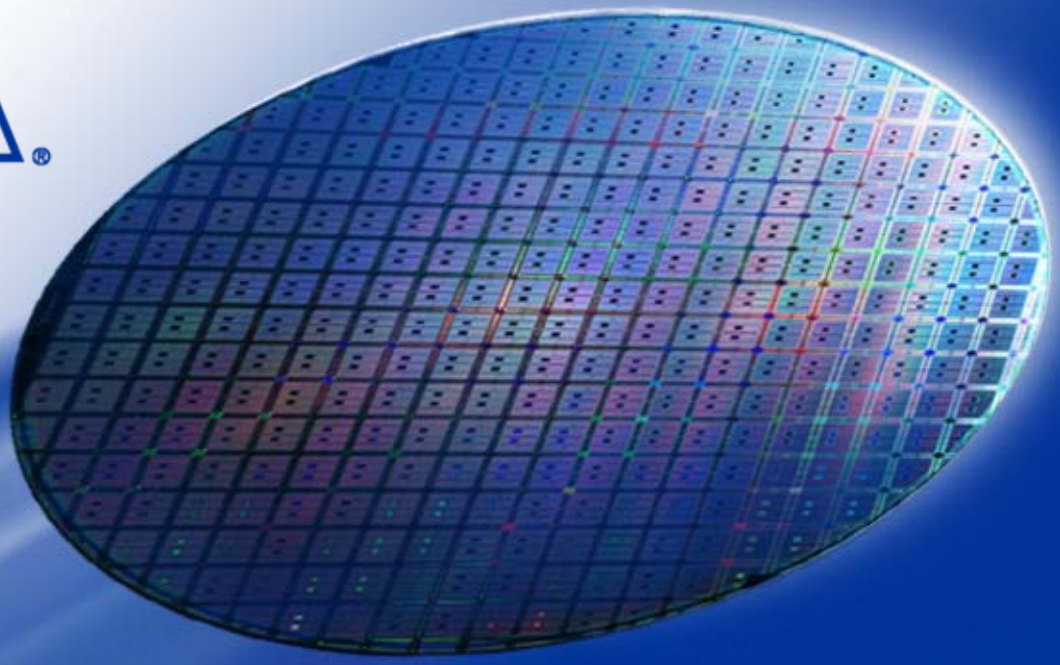
Appendix

Copyright © 2004 Altera Corporation

Appendix

- ⑤Quartus Memory Editor
- ⑤More Attributes & Directives
- ⑤DSP MAC Support for Stratix

Copyright © 2004 Altera Corporation



Memory Editor

Copyright © 2004 Altera Corporation

Memory Editor

- Create or Edit Memory Initialization Files in Intel Hex (.HEX) or Altera-Specific (.MIF) Format

- Design Entry
  - Use to Initialize Your Memory Block (Ex. RAM, ROM) during Power-Up

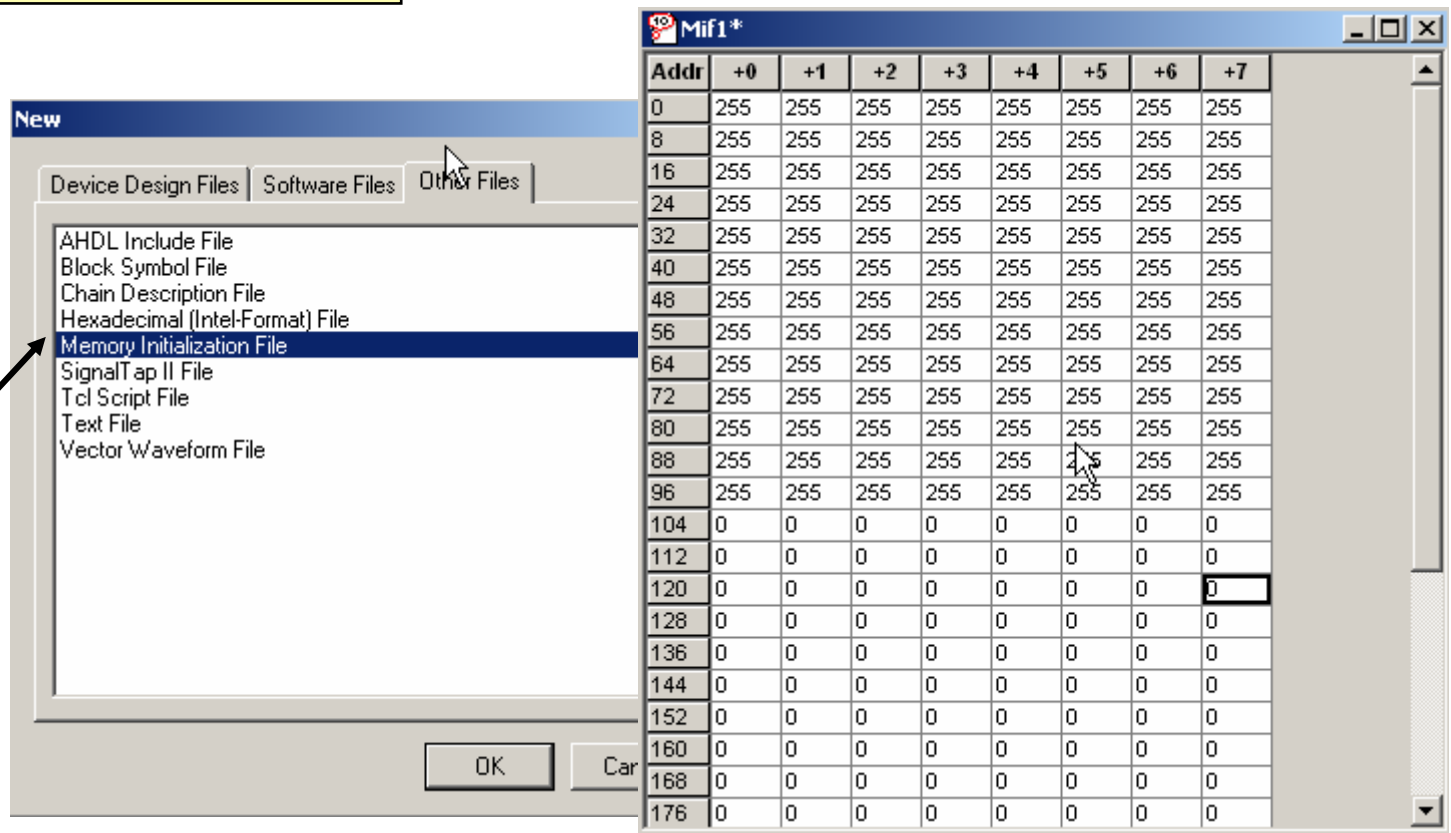
- Simulation
  - Use to Initialize Memory Blocks before Simulation

Copyright © 2004 Altera Corporation

Create Memory Initialization File

File ⇒New ⇒Other Files tab

HEX format or MIF format



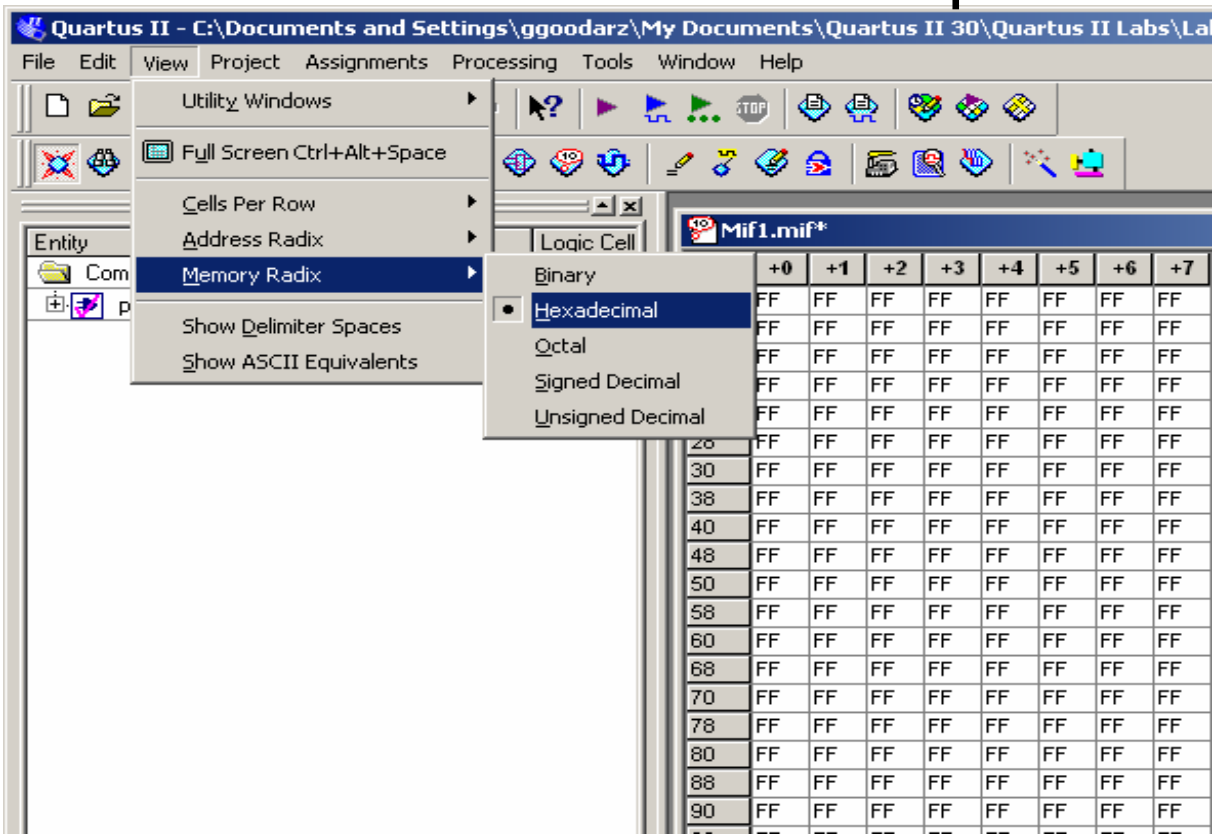
Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	255	255	255	255	255	255	255	255
8	255	255	255	255	255	255	255	255
16	255	255	255	255	255	255	255	255
24	255	255	255	255	255	255	255	255
32	255	255	255	255	255	255	255	255
40	255	255	255	255	255	255	255	255
48	255	255	255	255	255	255	255	255
56	255	255	255	255	255	255	255	255
64	255	255	255	255	255	255	255	255
72	255	255	255	255	255	255	255	255
80	255	255	255	255	255	255	255	255
88	255	255	255	255	255	255	255	255
96	255	255	255	255	255	255	255	255
104	0	0	0	0	0	0	0	0
112	0	0	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0
128	0	0	0	0	0	0	0	0
136	0	0	0	0	0	0	0	0
144	0	0	0	0	0	0	0	0
152	0	0	0	0	0	0	0	0
160	0	0	0	0	0	0	0	0
168	0	0	0	0	0	0	0	0
176	0	0	0	0	0	0	0	0

Copyright © 2004 Altera Corporation

Change Options

⑤View Options of Memory Editor

– View ⇒Select from Available Options

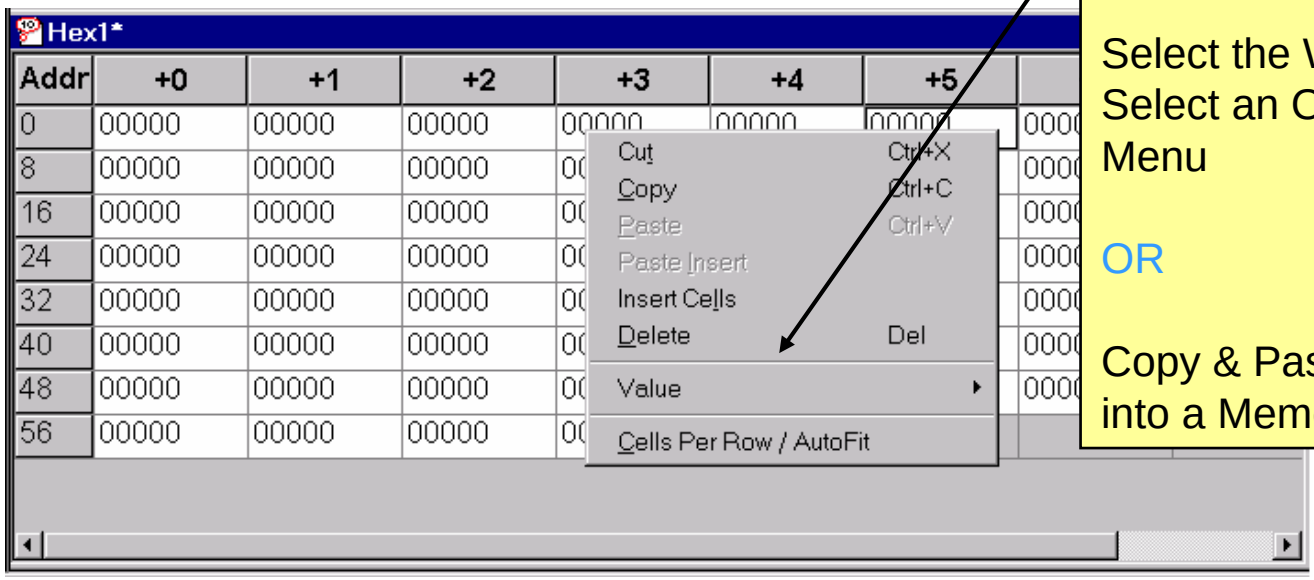


Copyright © 2004 Altera Corporation

Edit Contents

⑤Edit Contents of the Memory File

⑤Save the Memory File as .HEX or .MIF File



Select the Word & Type in a Value

OR

Select the Word & Right Click to Select an Option from the Pop-Up Menu

OR

Copy & Paste from a Spreadsheet into a Memory File

Copyright © 2004 Altera Corporation

Memory Size Wizard

Need to Edit Size of Memory File?

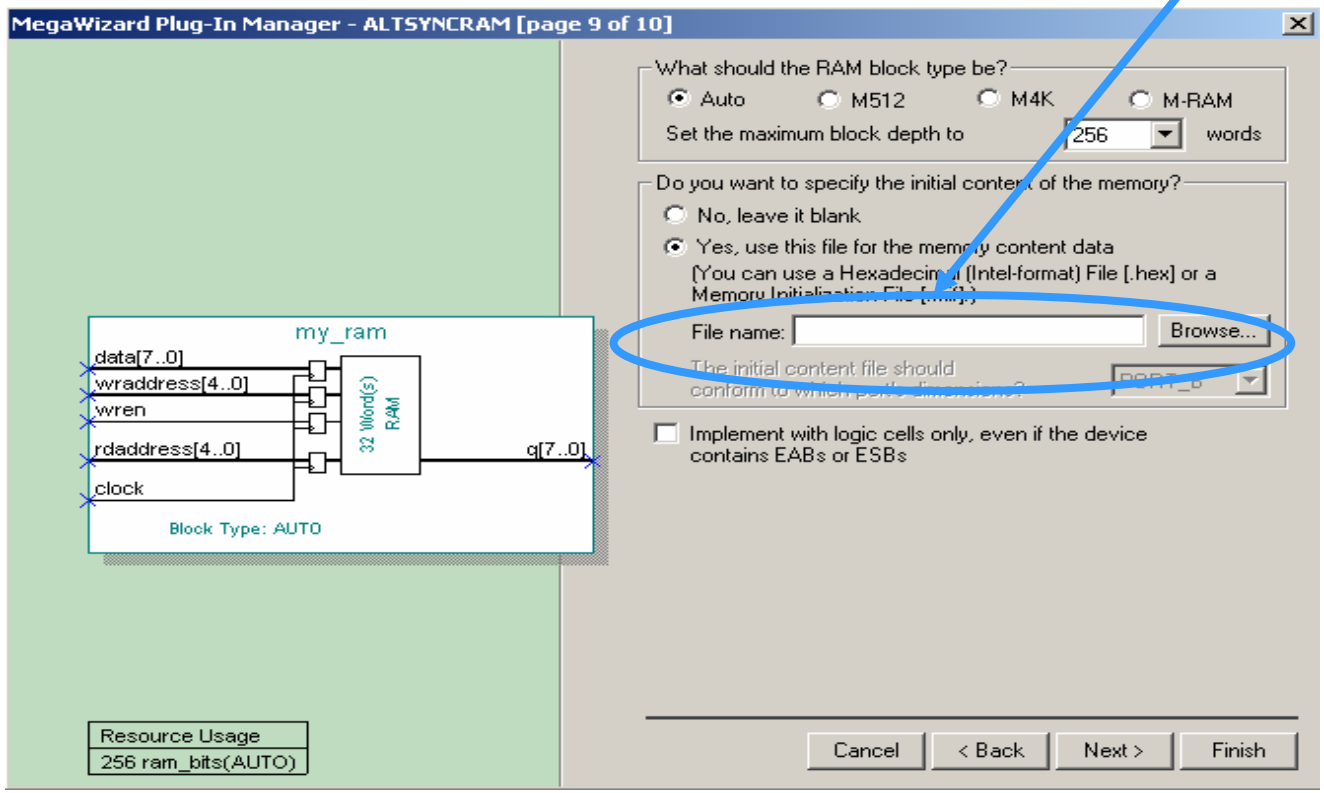
⑤Use the Memory Size Wizard (Edit Menu)

- Edit Word Size
- Edit Number of Words
- Specify How to Handle Word Size Change
  - ③Increasing Word Size
    - Pad Words
    - Combine Words
  - ③Decreasing Word Size
    - Truncate Words from Left
    - Truncate Words from Right

Copyright © 2004 Altera Corporation

Using Memory File in Design

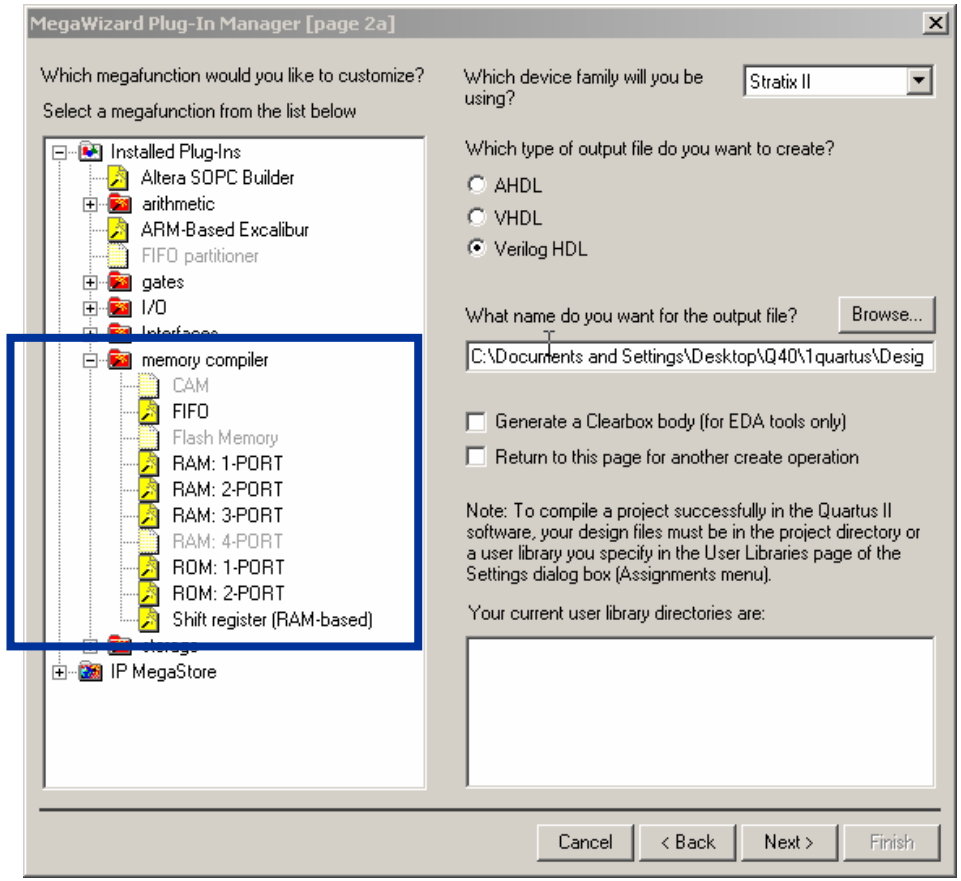
Specify MIF or HEX file in MegaWizard Interface



Copyright © 2004 Altera Corporation

Memory Compiler

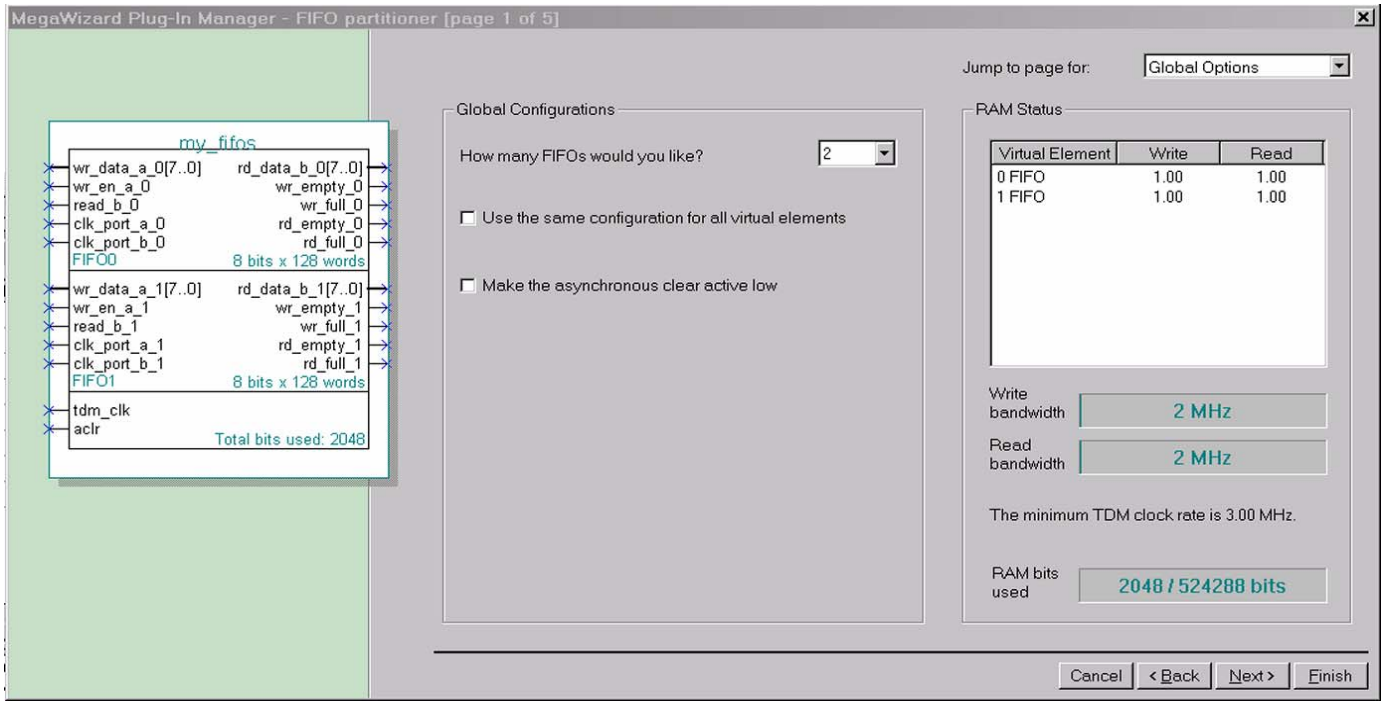
- ⑤ Memory Functions are Organized by Type
- ⑤ MegaWizard is Family-Aware
  - Select Memory Type & Parameters
  - Appropriate Megafunction Selected Automatically



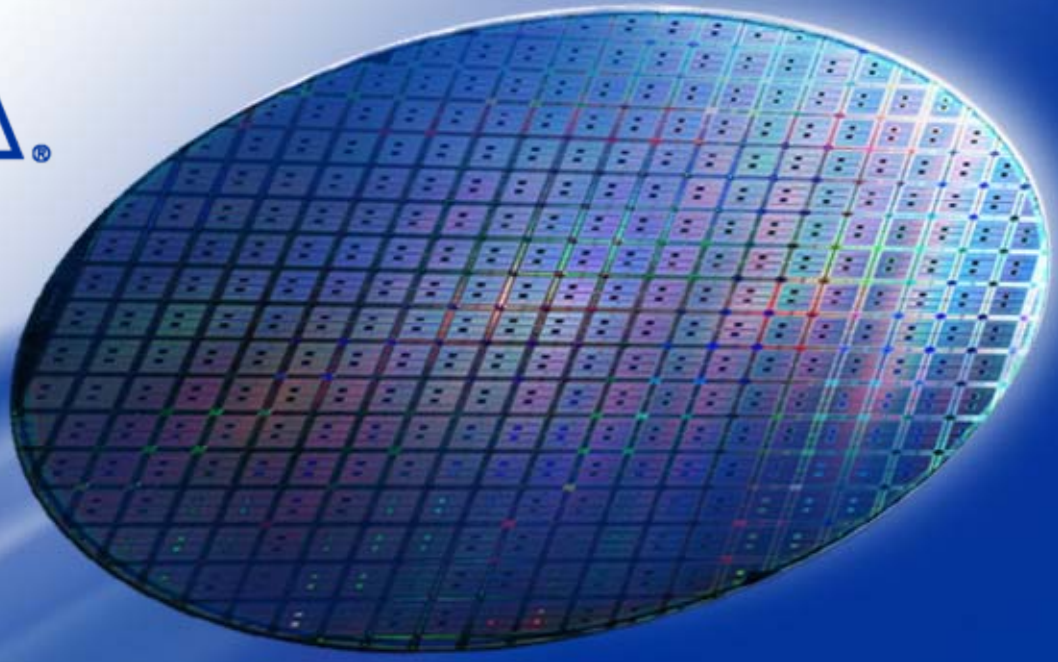
Copyright © 2004 Altera Corporation

FIFO Partitioner

- Map Multiple FIFOs into a Single Physical Memory Block
- Partition M-RAM Blocks into Multiple FIFOs



Copyright © 2004 Altera Corporation



More Attributes & Directives

Copyright © 2004 Altera Corporation

More Directives & Attributes

Attributes

- *syn_state_machine*
- *syn_sharing*
- *parallel_case / full_case*

Directives

- *altera_chip_pin_lc*
- *altera_implement_in_eab/esb*
- *altera_io_opendrain*

Copyright © 2004 Altera Corporation

Directive - *syn_state_machine*

- Enables/Disables FSM Compiler for an Individual State Machine
- Apply to State Register Declaration
- Settings
  - *1* or *0* (*TRUE* or *FALSE*)

Verilog

```
reg [7:0] current_state  /*  synthesis syn_state_machine = 1  */  ;
```

VHDL

```
SIGNAL current_state  :  std_logic_vector (7 DOWNTO 0);
ATTRIBUTE syn_state_machine :  BOOLEAN;
ATTRIBUTE syn_state_machine OF current_state : SIGNAL IS TRUE;
```

Copyright © 2004 Altera Corporation

Directive – *syn_sharing*

- Enables/Disables Resource Sharing on a Module/Entity
- Apply to Modules/Entities Individually or Globally
- Overrides Project Setting
- Settings
  - *ON* or *OFF*

Verilog

```
Module alu (out, opcode, a, b)  /*  synthesis syn_sharing = "off"  */  ;
```

VHDL

```
ARCHITECTURE rtl OF alu IS
ATTRIBUTE syn_sharing :  STRING;
ATTRIBUTE syn_sharing OF rtl : ARCHITECTURE IS "off";
```

Copyright © 2004 Altera Corporation

Directive – *parallel_case* (Verilog Only)

- Forces Parallel Muliplexers Versus Priority Encoding
  - User Knows Design Will Only Allow One Match at Any Time
- Apply Directly to Case, Casex, or Casez Statements

Verilog

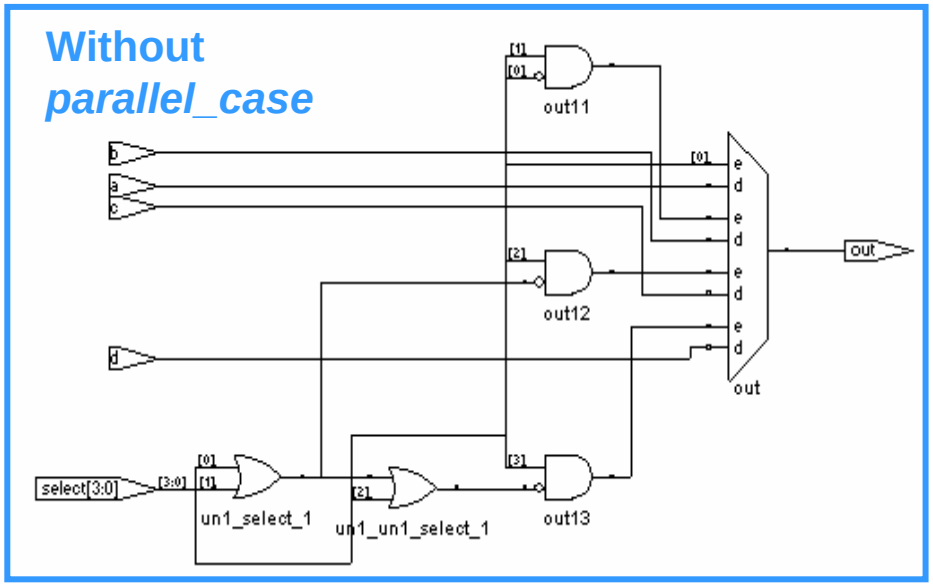
```
Casez (select)  // synthesis  parallel_case
   4'b???1:   out = a;
   4'b??1?:   out = b;
   4'b?1??:   out = c;
   4'b1???:   out = d;
   default:    out = 'bx;
endcase
```

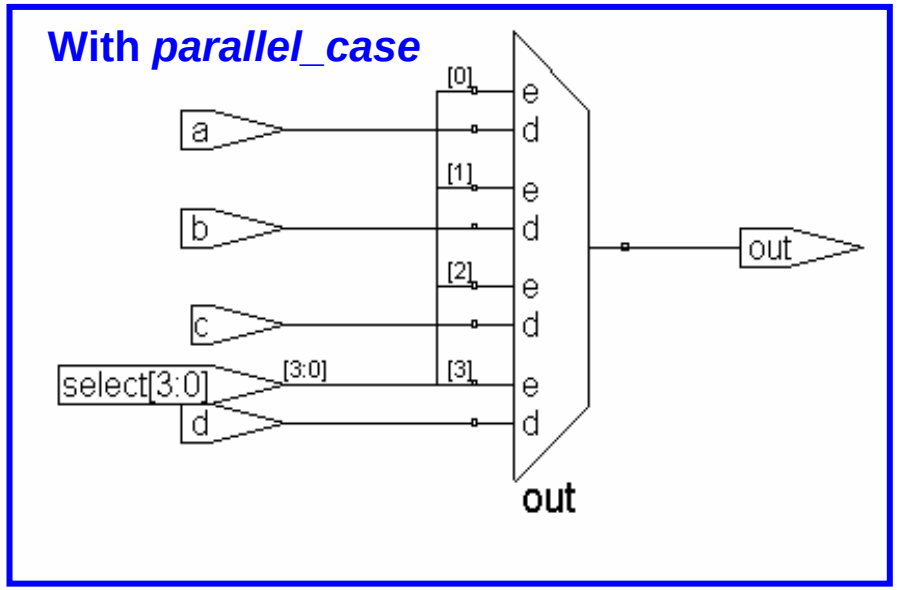
Note: *This Directive Is Unnecessary If the Input Logic Is Decoded Within the Same Module as the Case Statement as Synplify Pro Will Automatically Determine if the Input Conditions are Parallel*

Copyright © 2004 Altera Corporation

Example: *parallel_case*

```
always @(select or a or b or c or d)
 begin
   casez (select) /* synthesis parallel_case */
      4'b???1: out = a;
      4'b??1?: out = b;
      4'b?1??: out = c;
      4'b1???: out = d;
      default: out = 'bx;
   endcase
 end
```





Copyright © 2004 Altera Corporation

Directive – *full_case* (Verilog Only)

- Indicates All Possible Values Have Been Defined
  - No Additional Hardware Needs to Be Generated by Synplify Pro to Protect Design (Preserve Signal Values)
- Apply Directly to Case, Casex, or Casez Statements
- May Lead to Simulation Mismatches Since Simulators Do Not Recognize Directives
  - Use "default : 'bx" in a Casez as an Alternative

Verilog

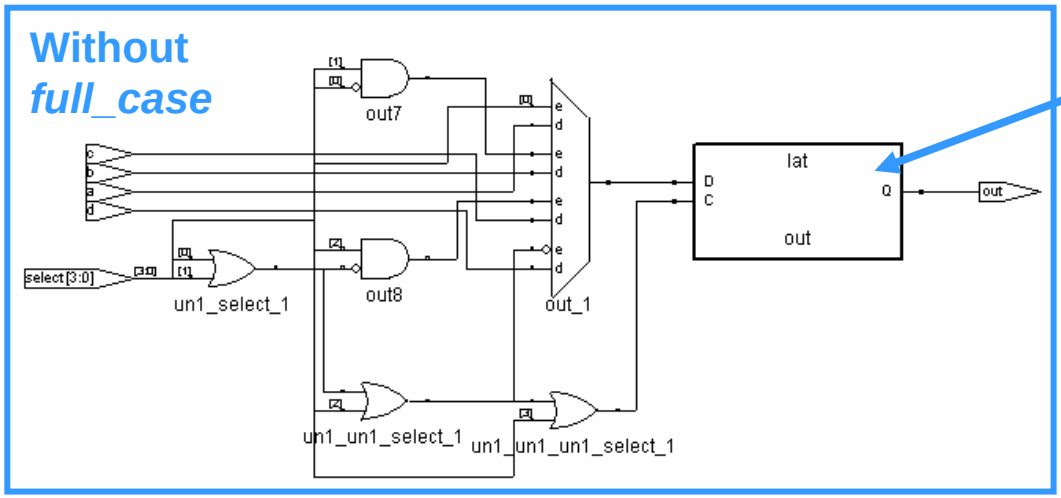
```
Casez (select)  // synthesis  full_case
   4'b???1:   out = a;
   4'b??1?:   out = b;
   4'b?1??:   out = c;
   4'b1???:   out = d;
endcase
```

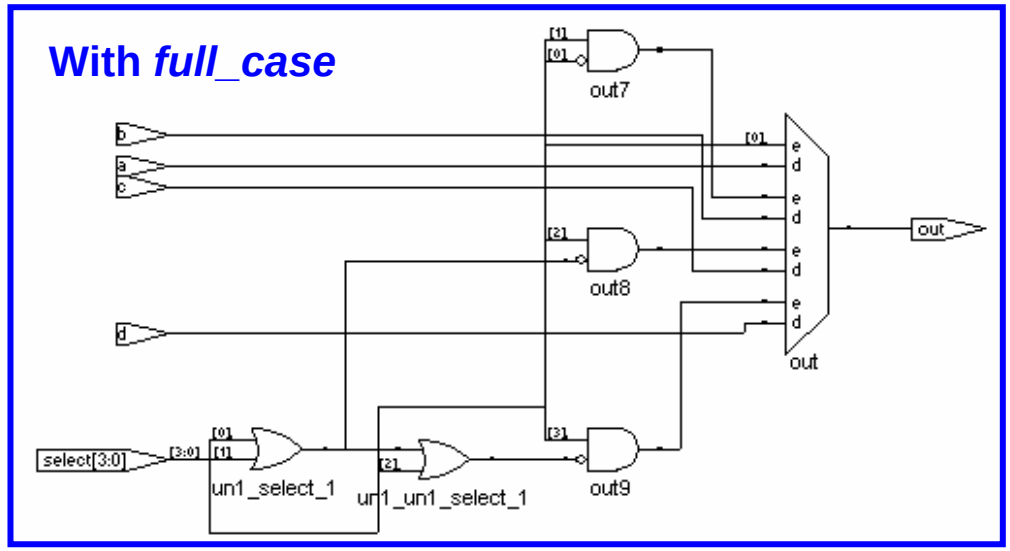
What Happens when Select = "0000"? Synplify Pro Will Generate a Latch to Preserve the Value of Out if Select = "0000"

Note: This Directive Is Unnecessary If the Input Logic Is Decoded Within the Same Module as the Case Statement since Synplify Pro Will Automatically Determine If All Cases Have Been Considered

Copyright © 2004 Altera Corporation

Example: *full_case*





Copyright © 2004 Altera Corporation

Attribute – *altera_implement_in_eab/esb*

- Enables/Disables Logic Implementation in Memory Blocks instead of Logic Cells
  - ESBs : All APEX, Mercury, Excalibur
  - EABs : All FLEX 10K
- Improves Area Utilization
- Is Applied to
  - Module Instantiation (Verilog)
  - Component Declaration (VHDL)
- Settings
  - 1 (TRUE) : Enables ESB/EAB Usage
  - 0 (FALSE)

Note: Please Review Device Datasheets for Rules as to when Logic Can Be Moved in This Way

Copyright © 2004 Altera Corporation

Attribute – *altera_implement_in_eab/esb*

Verilog

```
node_buf u1 (.d_in (d_in), .d_out (d_out))
          /*  synthesis altera_implement_in_esb = 1 /*  ;
```

VHDL

```
COMPONENT node_buf IS PORT (
          d : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
          q : OUT STD_LOGIC_VECTOR (3 DOWNTO 0));
END COMPONENT node_buf;
     ATTRIBUTE altera_implement_in_esb : BOOLEAN;
     ATTRIBUTE altera_implement_in_esb u1 : LABEL IS TRUE;
```

Instance Name (Label)

Copyright © 2004 Altera Corporation

Attribute – *altera_chip_pin_lc*

- Specifies Pin Locations
  - Assignments Passed via NativeLink (TCL File)
- Is Applied to Individual I/O Ports as They Are Declared
- Use @ Symbol for FLEX & ACEX Devices

Verilog (APEX, Excalibur, Mercury, Stratix)

```
module reg4 (clk, d, q);
input clk     /*  synthesis altera_chip_pin_lc = “20”  */ ;
input [3:0] d     /*  synthesis altera_chip_pin_lc = “60,62,63,64”  */ ;
output [3:0]  q     /*  synthesis altera_chip_pin_lc = “65,66,67,68”  */ ;
```

Verilog (ACEX or FLEX)

```
module reg4 (clk, d, q);
input clk     /*  synthesis altera_chip_pin_lc = “@20”  */ ;
input [3:0] d     /*  synthesis altera_chip_pin_lc = “@60,@62,@63,@64”  */ ;
output [3:0]  q     /*  synthesis altera_chip_pin_lc = “@65,@66,@67,@68”  */ ;
```

Copyright © 2004 Altera Corporation

Attribute – *altera_chip_pin_lc* (cont.)

VHDL (APEX, Excalibur, Mercury, Stratix)

```
ENTITY reg4 IS PORT (
        clk : IN STD_LOGIC;
        d : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
        q : OUT STD_LOGIC_VECTOR (3 DOWNTO 0) );
    ATTRIBUTE altera_chip_pin_lc : STRING;
    ATTRIBUTE altera_chip_pin_lc OF clk : SIGNAL IS “20”;
    ATTRIBUTE altera_chip_pin_lc OF d : SIGNAL IS “60,62,63,64”;
    ATTRIBUTE altera_chip_pin_lc OF q : SIGNAL IS “65,66, 67, 68”;
END ENTITY reg4;
```

VHDL (ACEX or FLEX) – Same as above but Add @

```
    ATTRIBUTE altera_chip_pin_lc : STRING;
    ATTRIBUTE altera_chip_pin_lc OF clk : SIGNAL IS “@20”;
    ATTRIBUTE altera_chip_pin_lc OF d : SIGNAL IS “@60,@62,@63,@64”;
    ATTRIBUTE altera_chip_pin_lc OF q : SIGNAL IS “@65,@66,@67,@68”;
```

Copyright © 2004 Altera Corporation

Attribute – *altera_io_opendrain*

- Enables Open Drain Mode on Device I/O
- Applied to Output & Bi-Directional Ports
- Settings
  - 1 (TRUE) : Enables Open-Drain Mode

Verilog

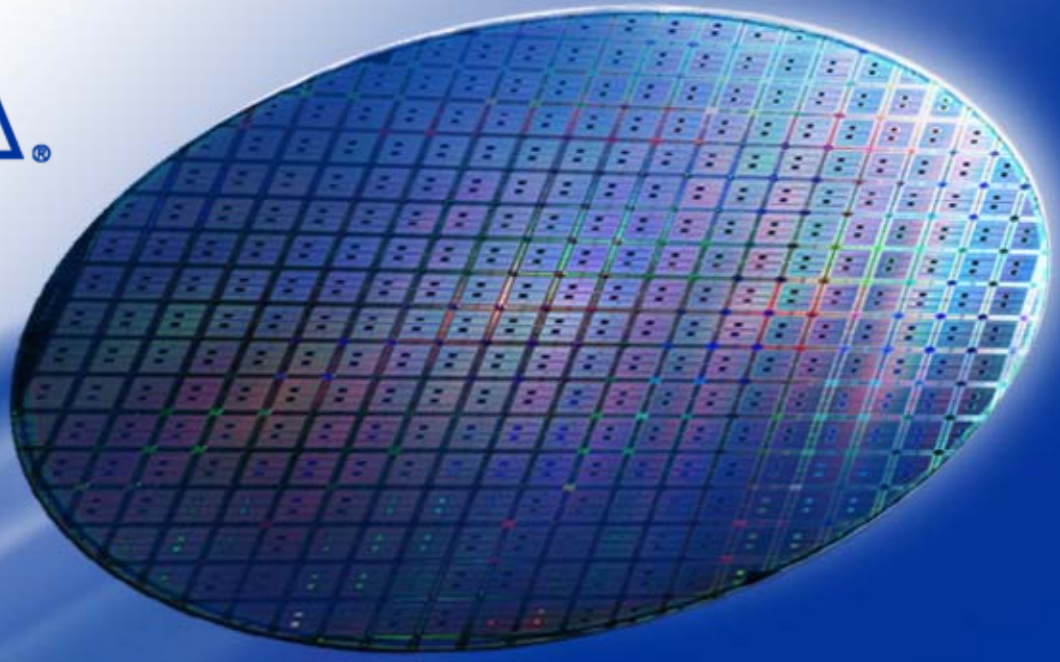
```
module node_buf (datain, dataout);
input [3:0] datain;
output [3:0]  dataout   /*  synthesis altera_io_opendrain = 1  */ ;
```

VHDL

```
ENTITY node_buf IS PORT (
          datain : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
          dataout : OUT STD_LOGIC_VECTOR (3 DOWNTO 0) );
     ATTRIBUTE altera_io_opendrain : BOOLEAN;
     ATTRIBUTE altera_io_opendrain OF dataout : BOOLEAN IS TRUE;
END ENTITY reg4;
```

Note: Please Review Device Datasheets to See if Device I/O Support This Functionality

Copyright © 2004 Altera Corporation



DSP MAC Support for Stratix

Copyright © 2004 Altera Corporation

DSP MAC Block Support

- Different configurations:
  - LPM_MULT: Plain Multipliers
    - Unregistered
    - Pipelined
  - ALTMULT_ADD: Multipliers followed by adder
    - Unregistered
    - Registered
  - ALTMULT_ACCUM: Multiplier followed by accumulator register
    - Unregistered
    - Registered
- Width of Multiplier > 4 bits
- To override inference, use ***syn_multstyle="logic"***

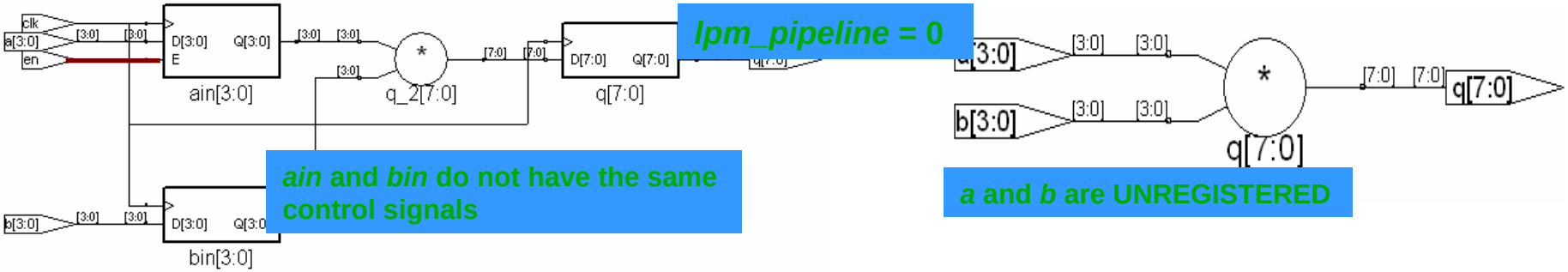
Copyright © 2004 Altera Corporation

Plain Multiplier

- Inferred to *LPM_MULT*
- Unregistered
  - Parameter *lpm_pipeline=0*
- Pipelined
  - Input / output regs must have same clock, aclr, & clken
    - DFF, DFFR, DFFRE, DFFE
  - Output register pipelined only if input registers pipelined
  - Values for parameter *lpm_pipeline*
    - 0: no pipeline
    - 1: input registers pipelined
    - 2: input and output registers pipelined

Copyright © 2004 Altera Corporation

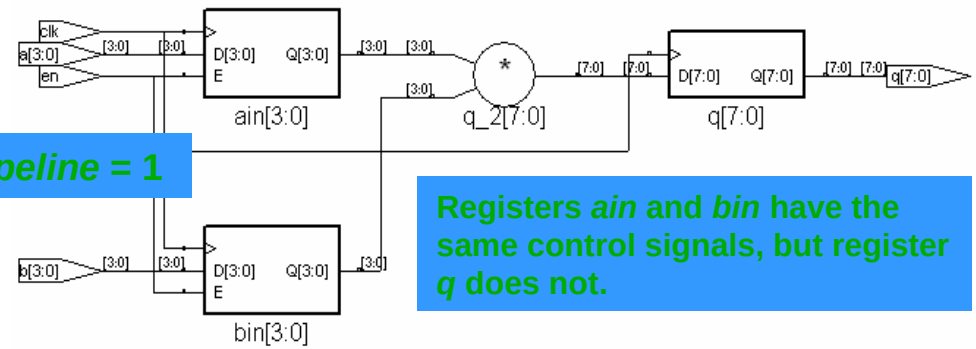
Plain Multiplier Examples



lpm_pipeline = 0

ain and *bin* do not have the same control signals

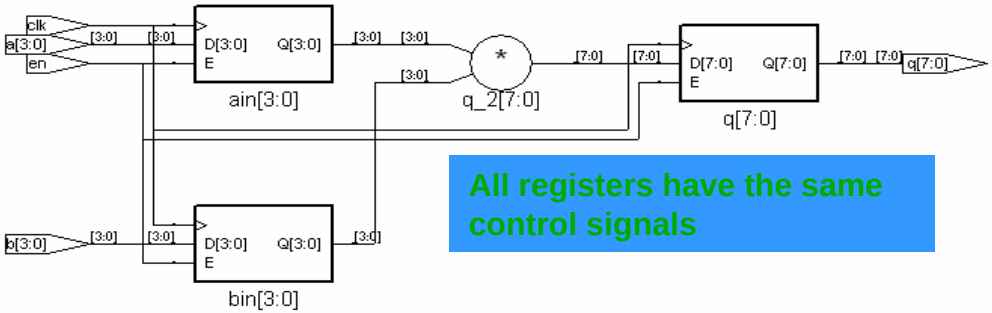
a and *b* are UNREGISTERED



lpm_pipeline = 1

Registers *ain* and *bin* have the same control signals, but register *q* does not.

lpm_pipeline = 2



All registers have the same control signals

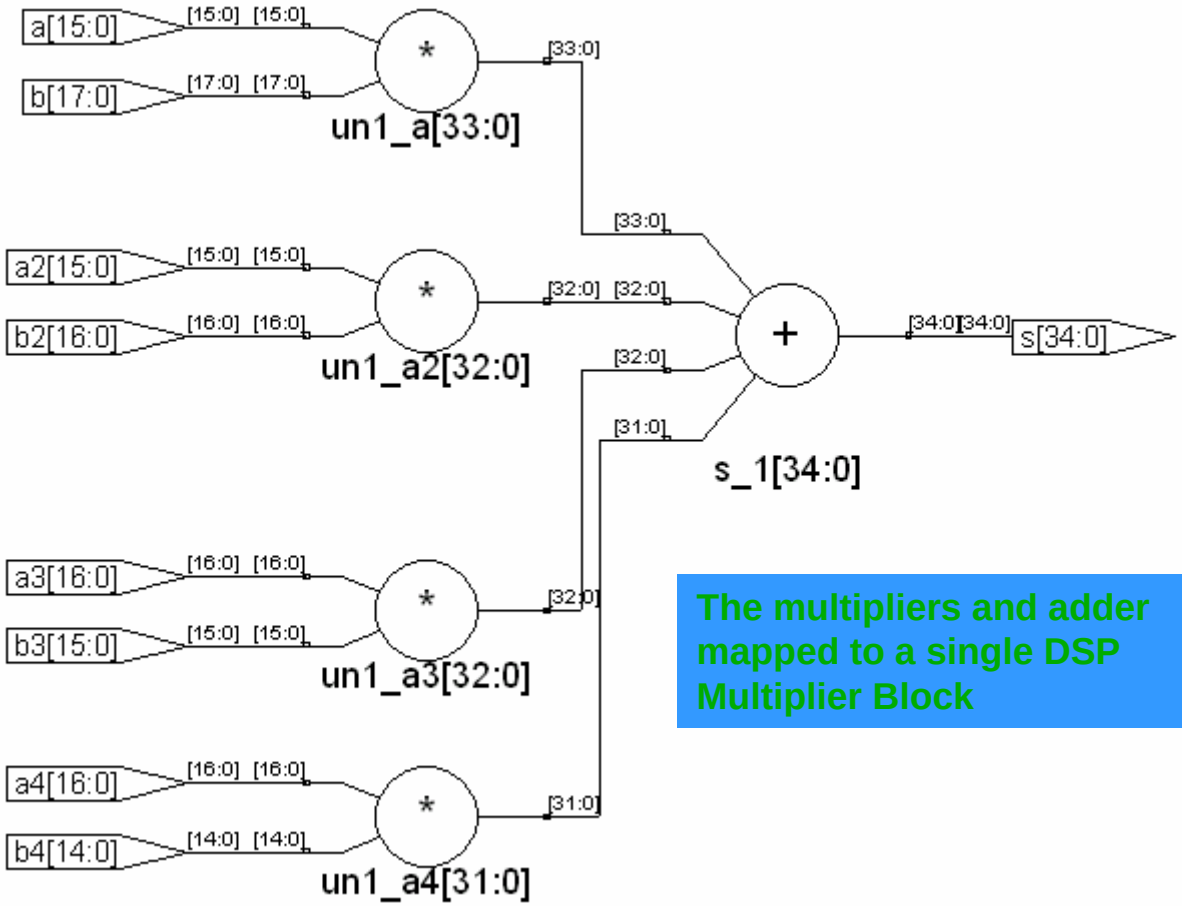
Multiply—Add Operation

- Inferred to *ALTMULT_ADD*
- Inferred when outputs of Multipliers added
  - q = a * b + c * d
  - q = (a * b + c * d) + (e * f + g * h)
  - q = (a * b + c * d) – (e * f + g * h)
- Up to 4 Multipliers of width <= 18, in one DSP Block
- Adder without Multipliers not packed into DSP Block
- Configurations
  - Unregistered
  - Registered

Copyright © 2004 Altera Corporation

Unregistered ALTMULT_ADD Example

- No registers on any of the inputs, multiplier outputs or output



The multipliers and adder mapped to a single DSP Multiplier Block

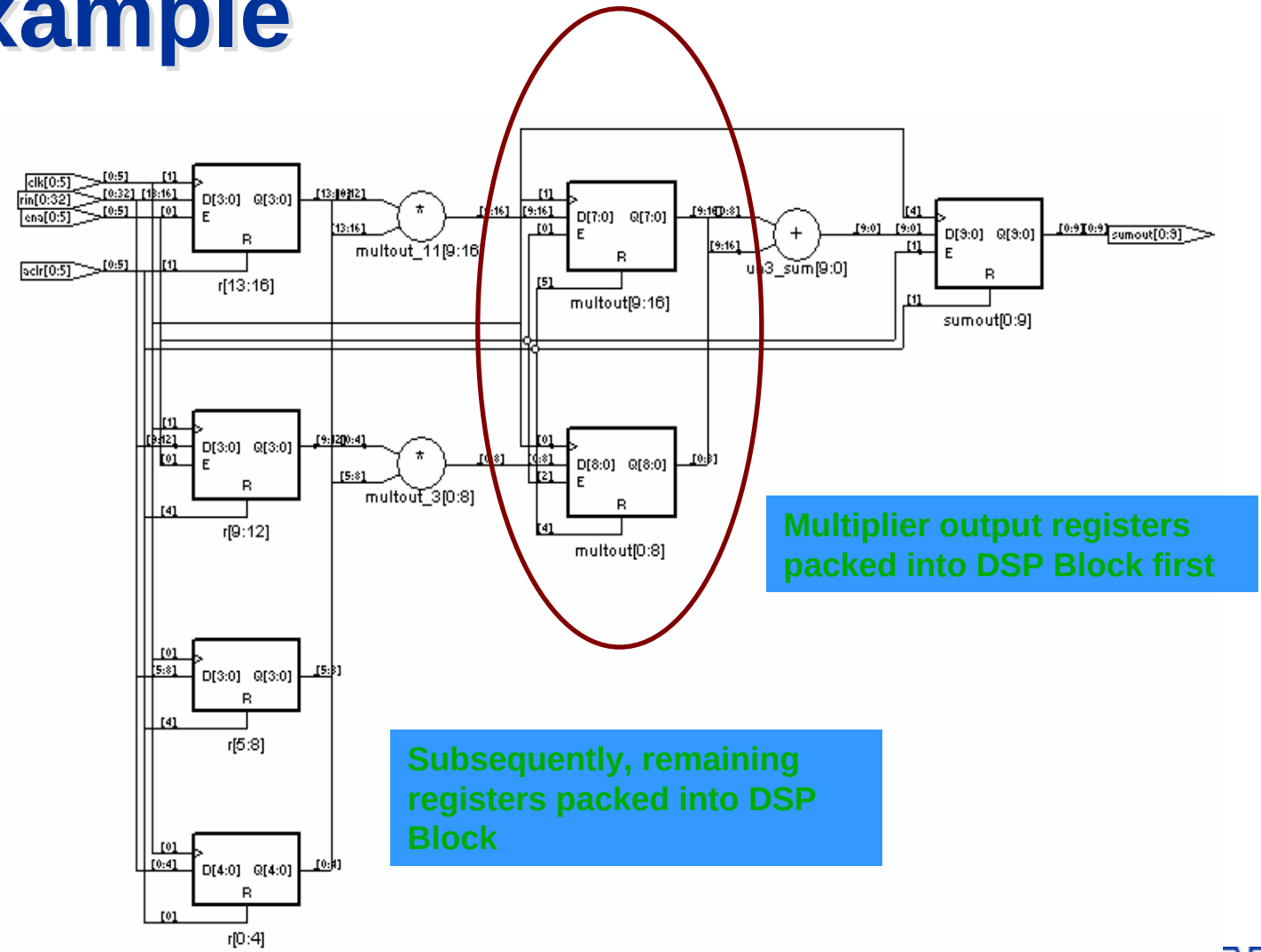
Copyright © 2004 Altera Corporation

Registered ALTMULT_ADD

- Configuration
  - Registers supported
    - dff, dffe, dffr, dffre
  - Up to 4 unique CLK / CLKEN pairs
    - Ena3 corresponds to clock3; ena2 corresponds to clock2, etc…
  - Up to 4 unique asynchronous resets
- Inference methodology
  - Registers on the output of multipliers sucked into the MAC first
  - Registers on input or output are sucked into the MAC next
  - What cannot be sucked into the MAC implemented as regular atoms

Copyright © 2004 Altera Corporation

Registered ALTMULT_ADD Example



Multiplier output registers packed into DSP Block first

Subsequently, remaining registers packed into DSP Block

Copyright © 2004 Altera Corporation

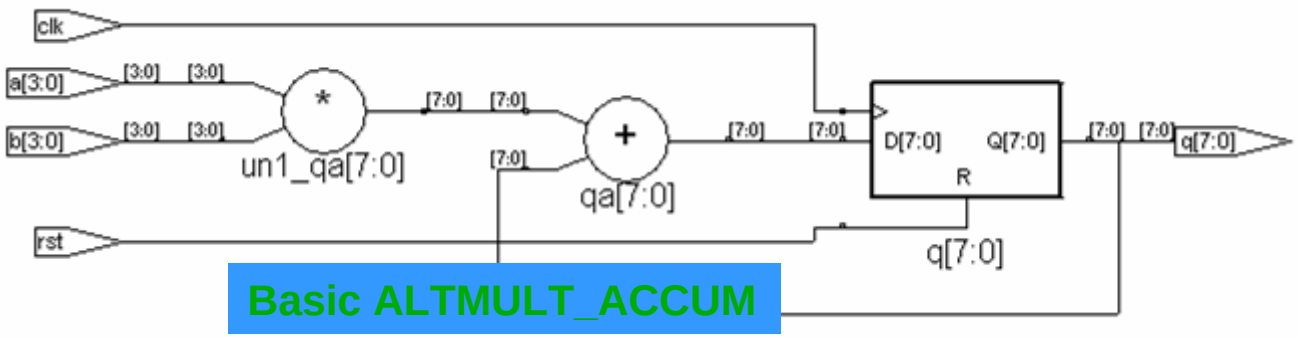
Other Multipliers

- *Constant Multiplier* implemented as logic
  - Better quality of results than mapping it to MAC
- Variable Multiplier Inputs of type (a*b+c) mapped to DSP MACs
  - Interpreted as (a*b+c*1)

Copyright © 2004 Altera Corporation

Multiply—Accumulate Operation

- Inferred to *ALTMULT_ACCUM*
- Inferred when Multiplier followed by accumulator register
  - q = q + a * b
  - q = q - a * b



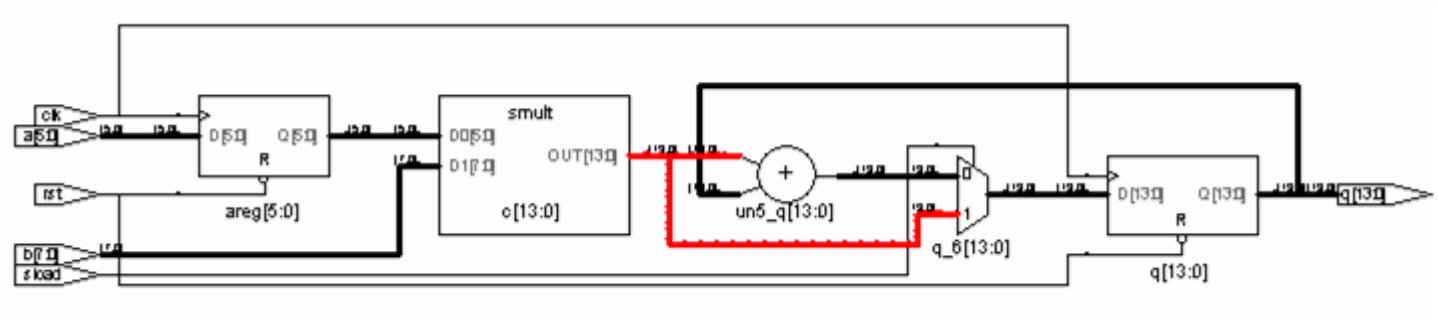
Basic ALTMULT_ACCUM

- Width of inputs <=
- Output width <= 52
  - LPM_MULT inferred if output width > 52 bits

Copyright © 2004 Altera Corporation

Sload_mux Pin Inferrence

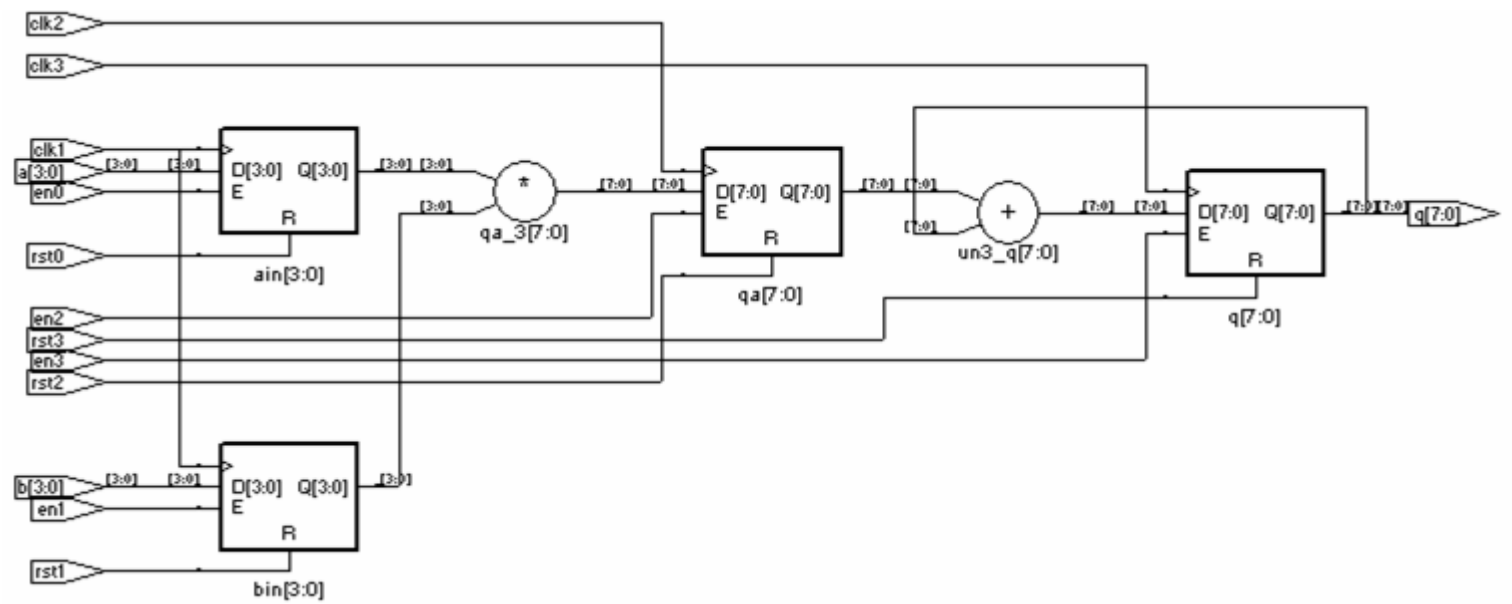
- ⑤ *sload_mux* pin of *altmult_accum* is inferred
- ⑤ Loads value at output of multiplier without having to reset register



Copyright © 2004 Altera Corporation

Registered ALTMULT_ACCUM

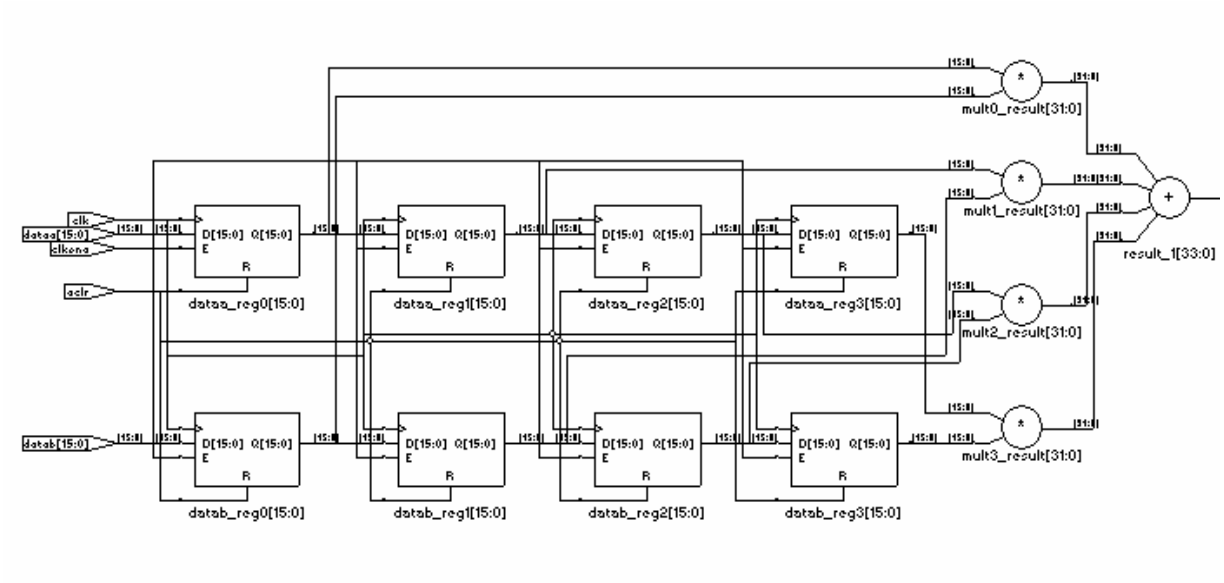
- Inputs & output of multiplier can be registered
  - dff, dffe, dffr, dffre
- Up to 4 unique CLK / CLKEN pairs
  - Ena3 corresponds to clock3; ena2 corresponds to clock2, etc…
- Up to 4 unique asynchronous resets



Copyright © 2004 Altera Corporation

Stratix Scan Chains In DSP MACs

- Multiplier inputs of multadd either registered or unregistered
- Full and partial scan chains inferred
  - When registered input A_{i+1} driven by the registered input A_i of the previous multiplier, register A_{i+1} said to be scan-chained
  - When one input of multiplier scan-chained, other external, said to be a partial scan chain



Copyright © 2004 Altera Corporation