Quartus® II Software Design Series: Foundation

Objectives

- Create a new Quartus® II project
- Choose supported design entry methods
- Compile a design into an FPGA
- Locate resulting compilation information
- Create design constraints (assignments & settings)
- Manage I/O assignments
- Perform timing analysis & obtain results

Class Agenda

- Intro to Altera & Devices
- Quartus II Feature Overview
- Design Methodology
- Projects
  - Exercise 1
- Design Entry
  - Exercise 2
- Compilation
  - Exercise 3
- Settings & Assignments
  - Exercise 4
- I/O Planning
  - Exercise 5
- Timing Analysis
  - Exercise 6

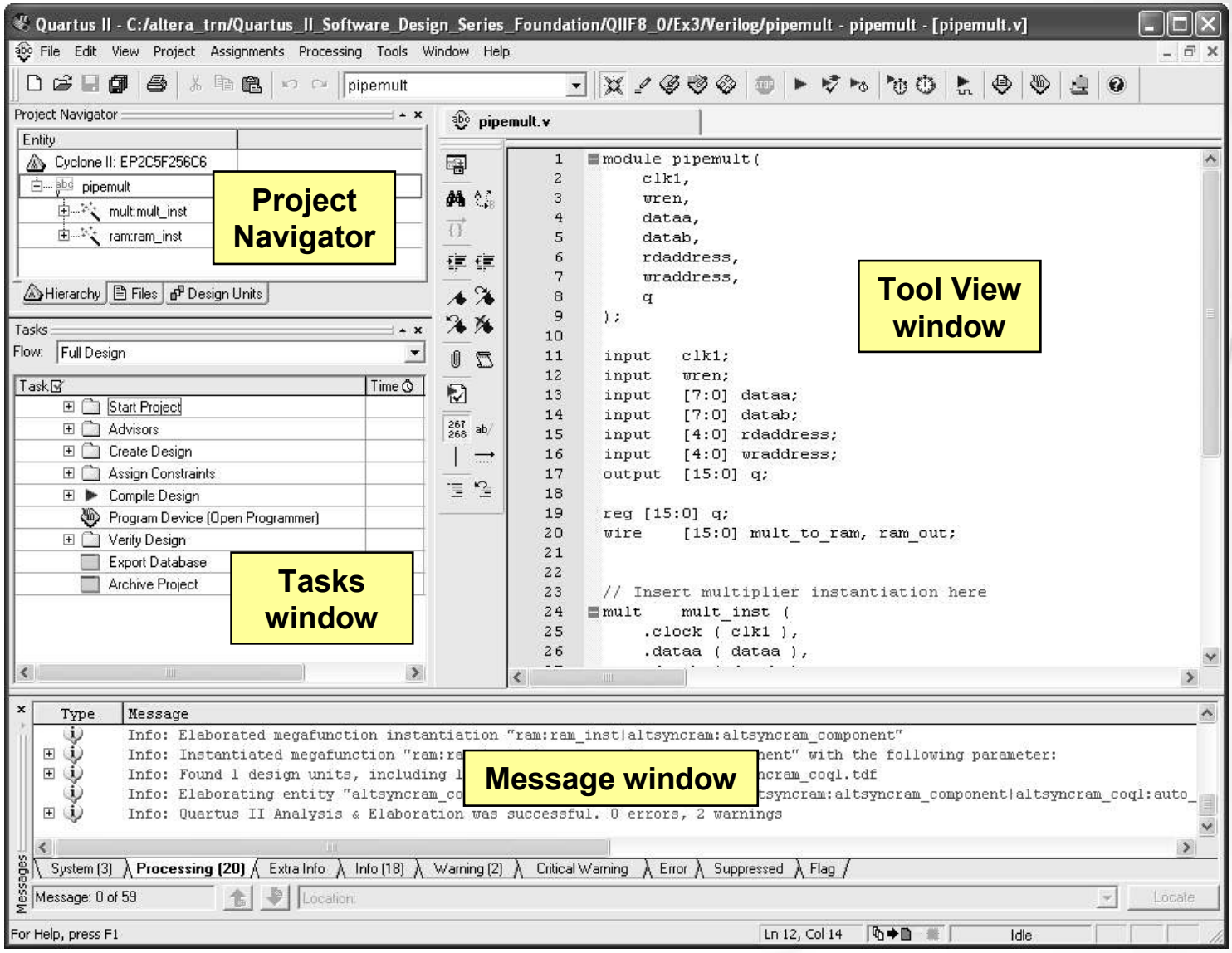
Quartus II Software Design Series: Foundation

Quartus II Design Software Feature Overview

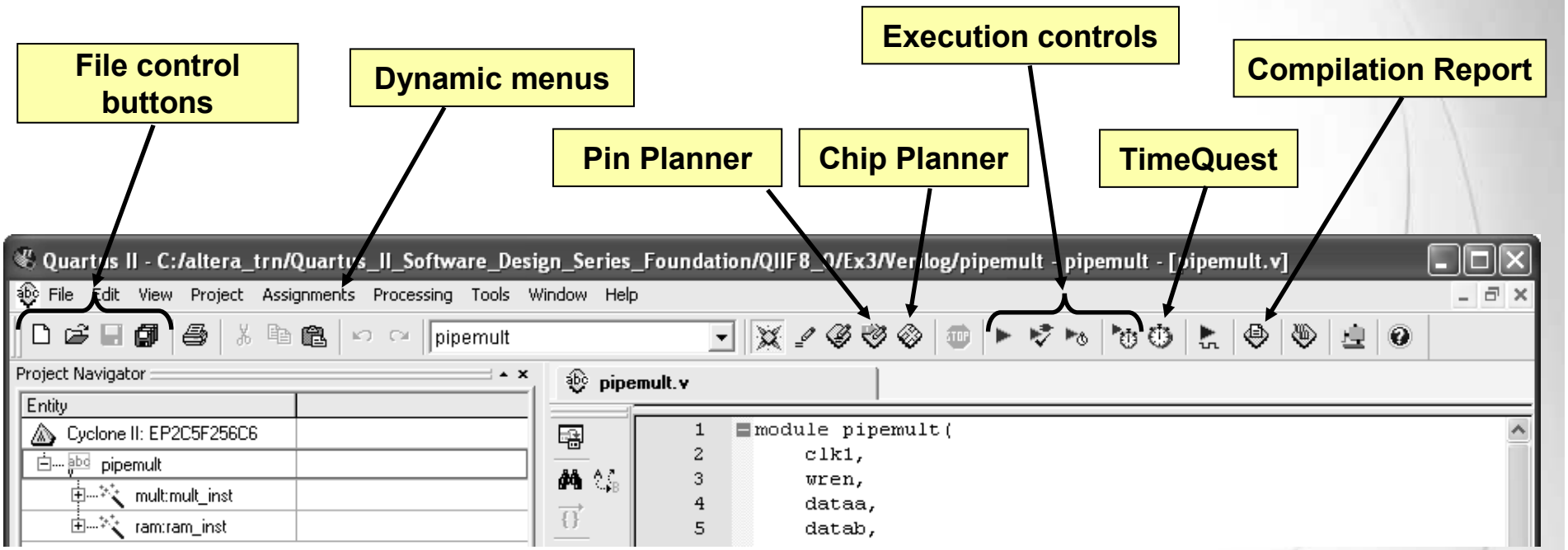
Quartus II Design Software

- Fully-integrated development tool
  - Multiple design entry methods
  - Logic synthesis
  - Place & route
  - Simulation
  - Timing & power analysis
  - Device programming

Quartus II Default Operating Environment



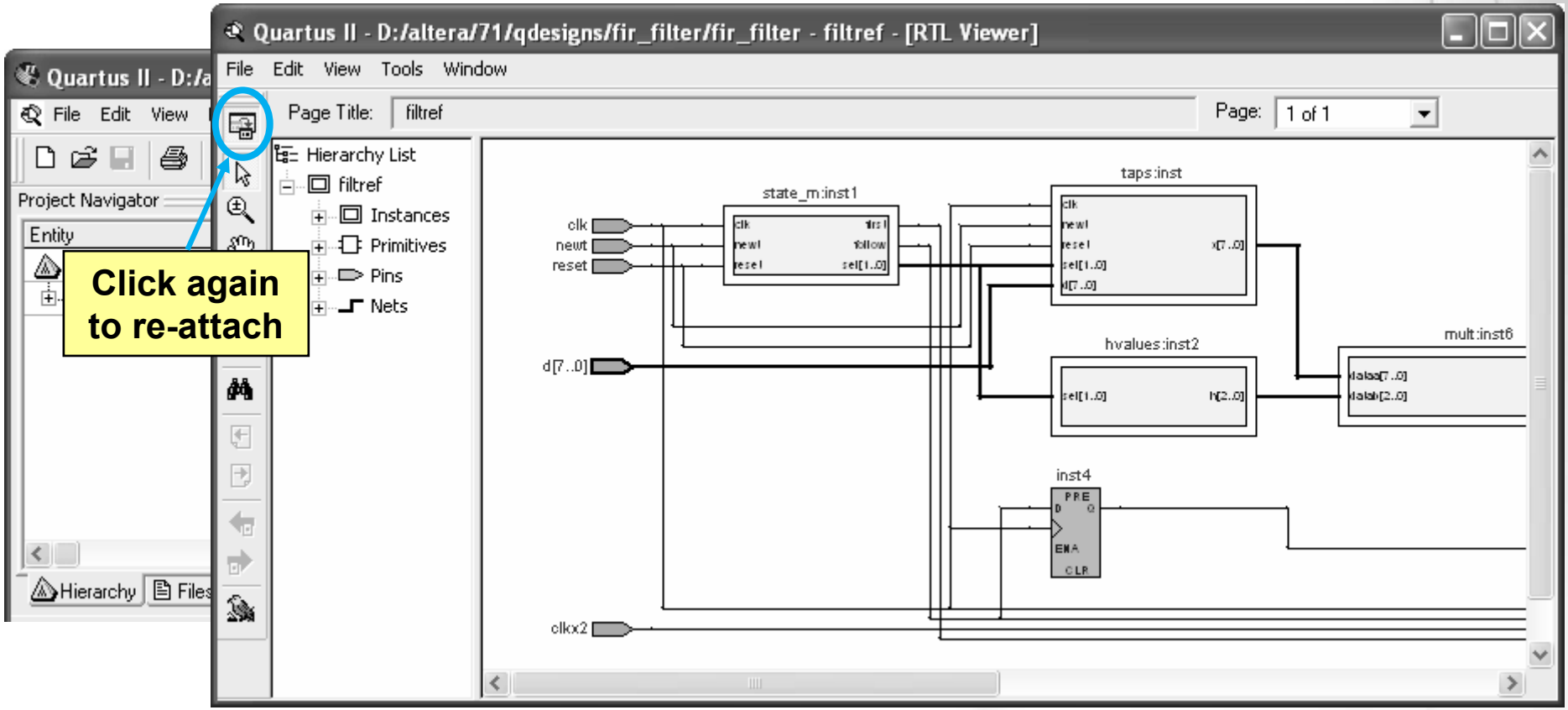
Main Toolbar



To reset views:
1. **Tools ⇒ Customize ⇒ Toolbars ⇒ Reset All**
2. **Restart Quartus II**

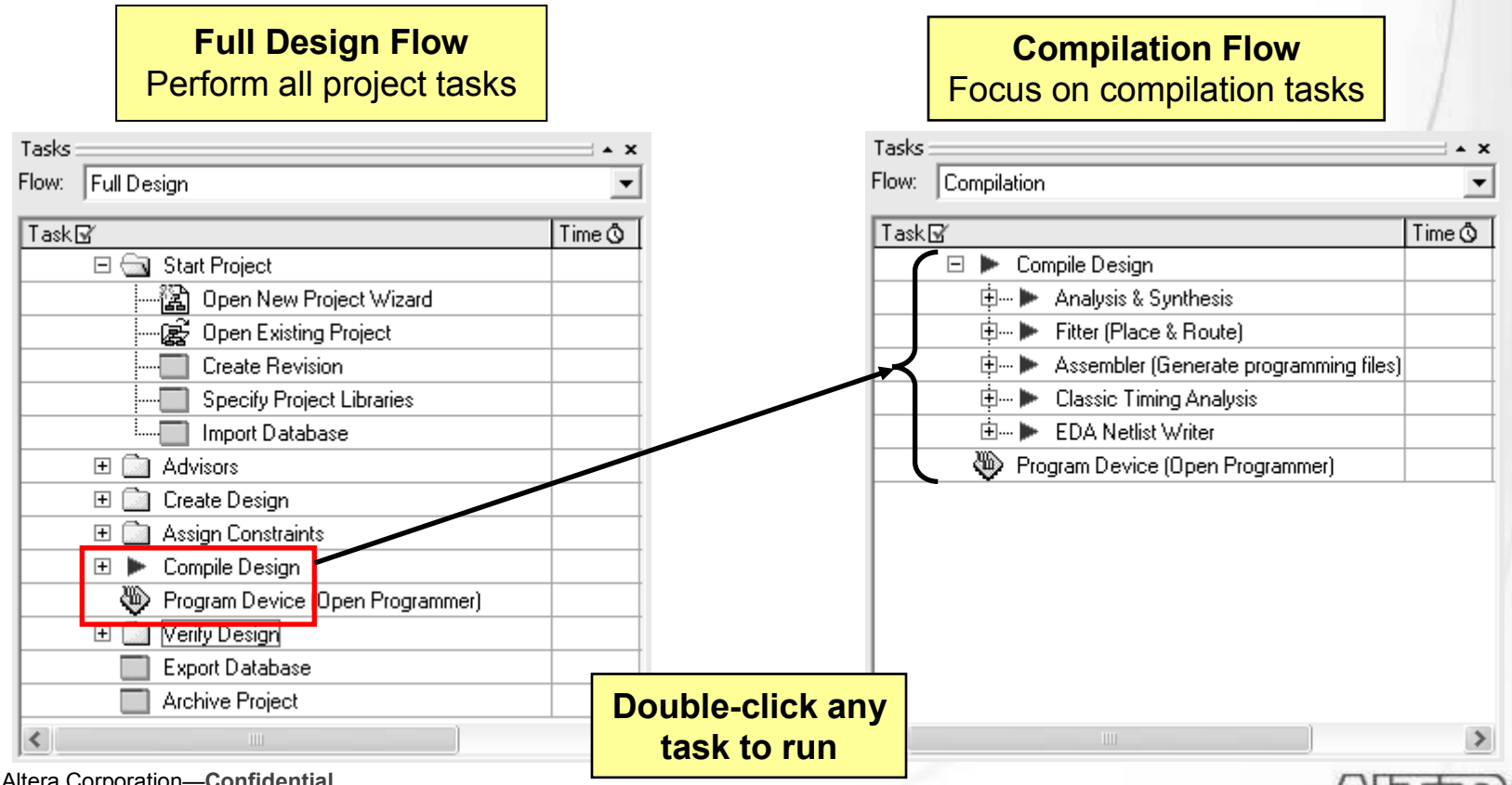
Detachable Windows

- Separate child windows from the Quartus II GUI frame (**Window** menu ⇒ **Detach/Attach Window**)



Tasks Window

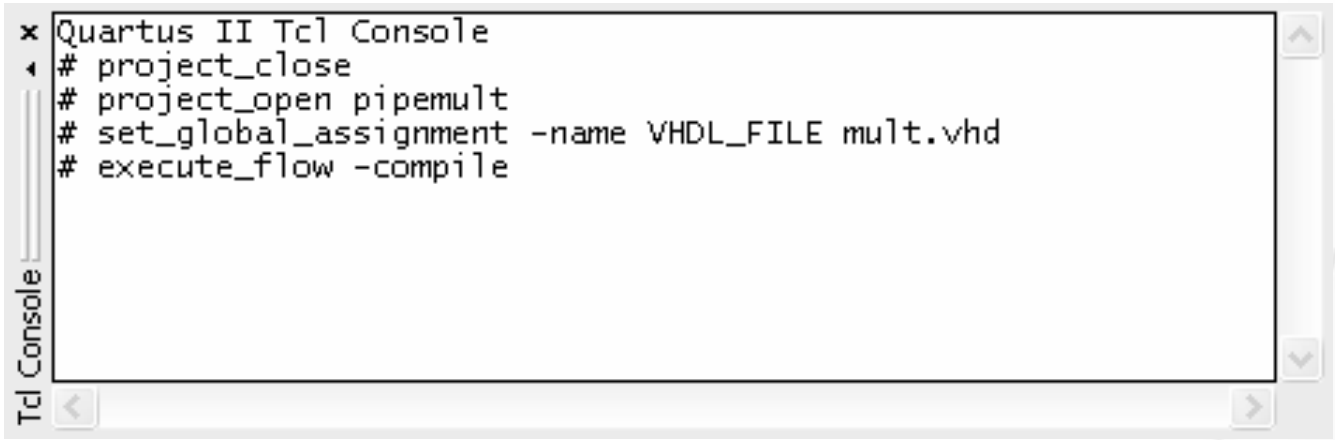
- Easy access to most Quartus II functions
- Organized into related tasks within two task flows



Tcl Console Window

- Enter and execute Tcl commands directly in the GUI

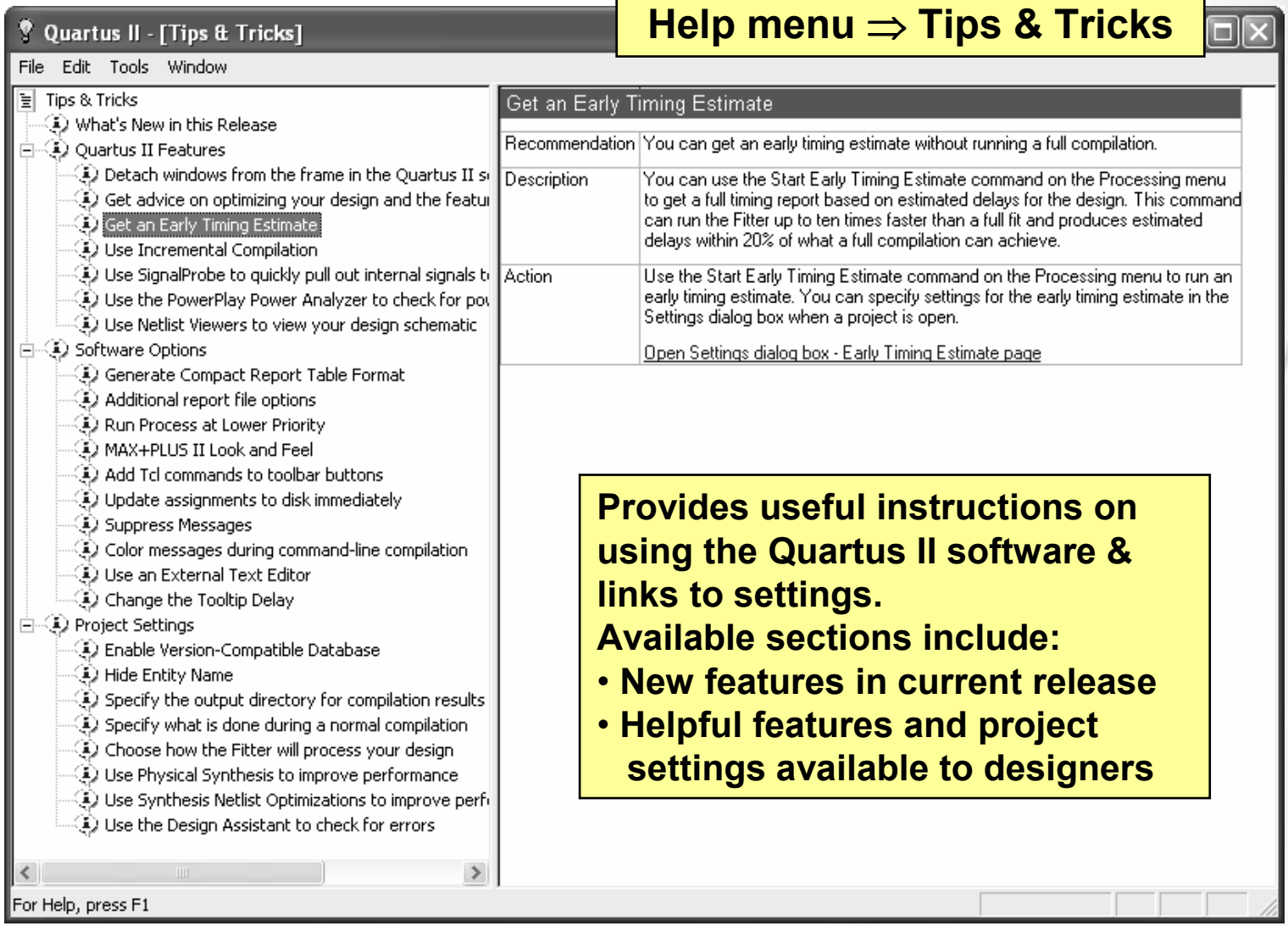
View menu ⇒ Utility Windows ⇒ Tcl Console



- Execute from command-line using Tcl shell
  - `quartus_sh --shell`

Tips & Tricks Advisor

Help menu ⇒ Tips & Tricks

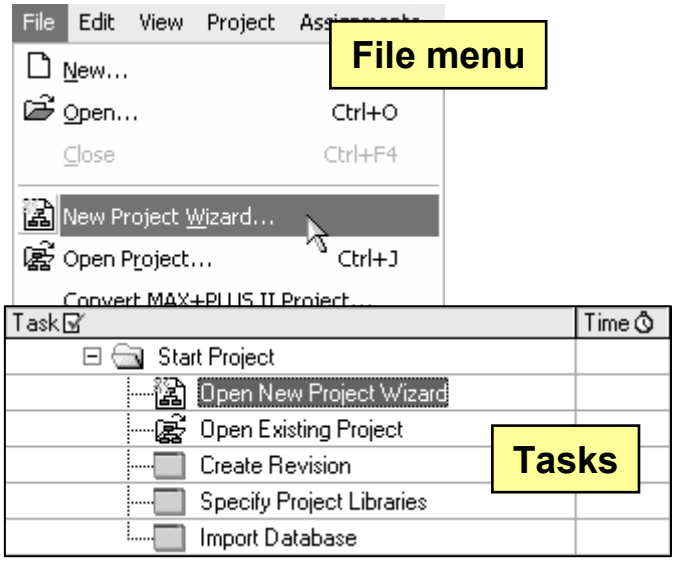


Provides useful instructions on using the Quartus II software & links to settings.
Available sections include:
- **New features in current release**
- **Helpful features and project settings available to designers**

Quartus II Software Design Series: Foundation

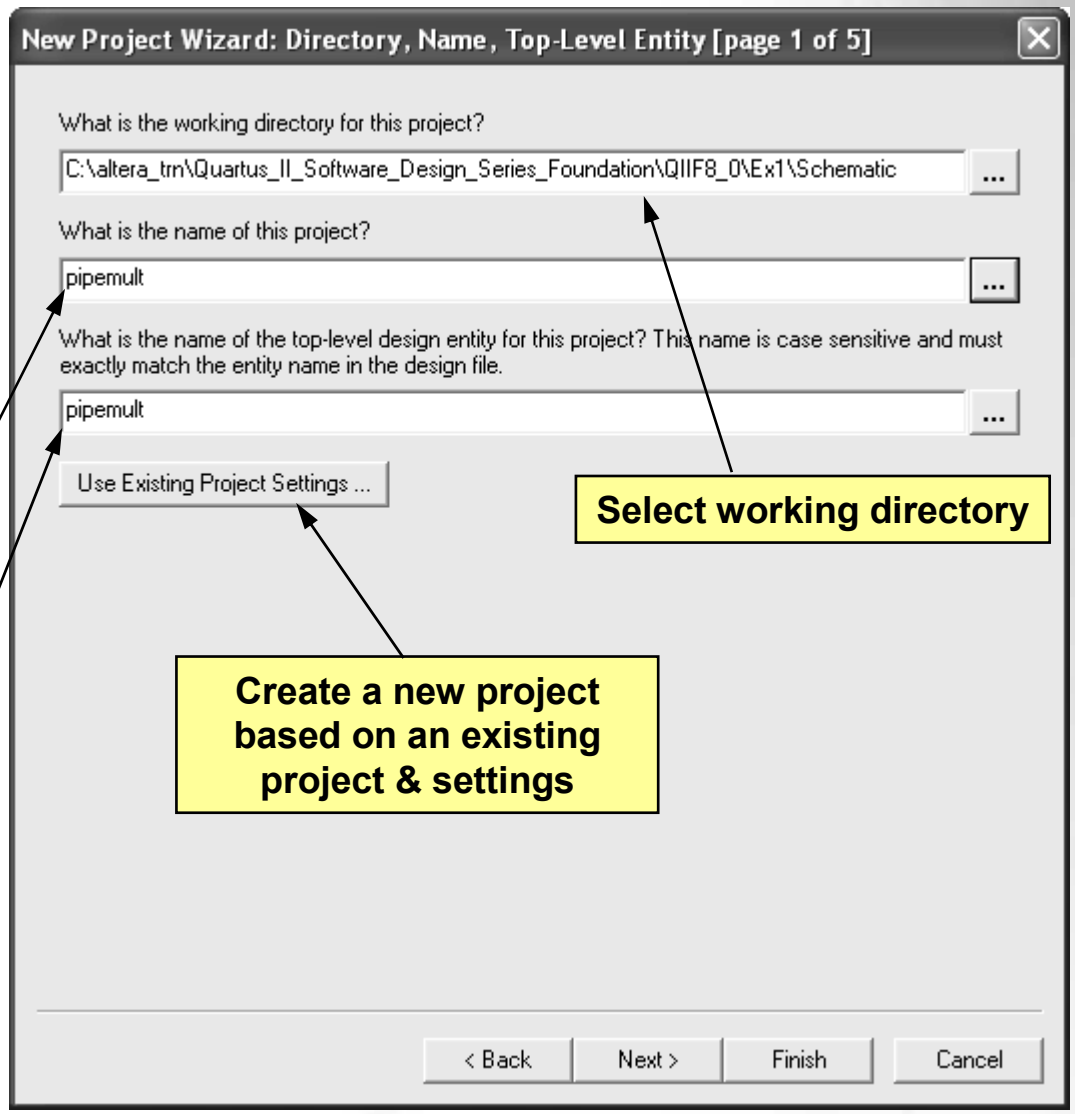
Quartus II Projects

New Project Wizard



File menu

Tasks



Select working directory

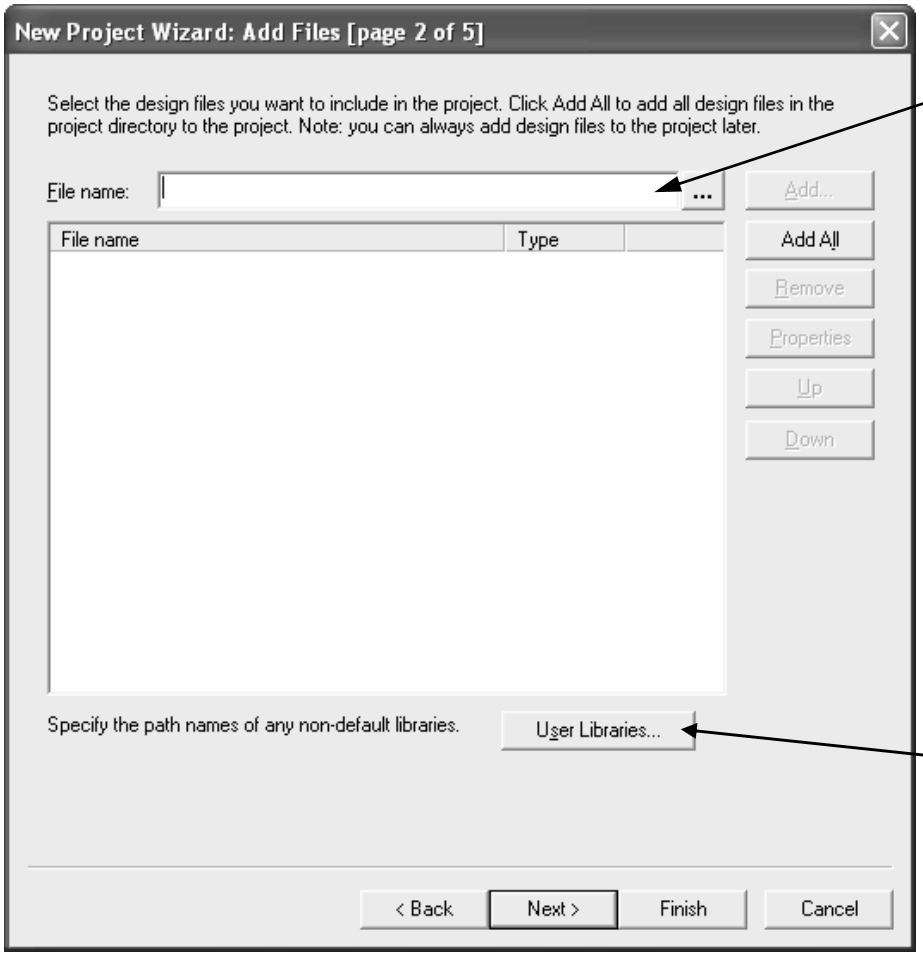
Name of project can be any name; recommend using top-level file name

Create a new project based on an existing project & settings

Top-level entity does not need to be the same name as top-level file name

Tcl: project_new <project_name>

Add Files



Add design files

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

Notes:

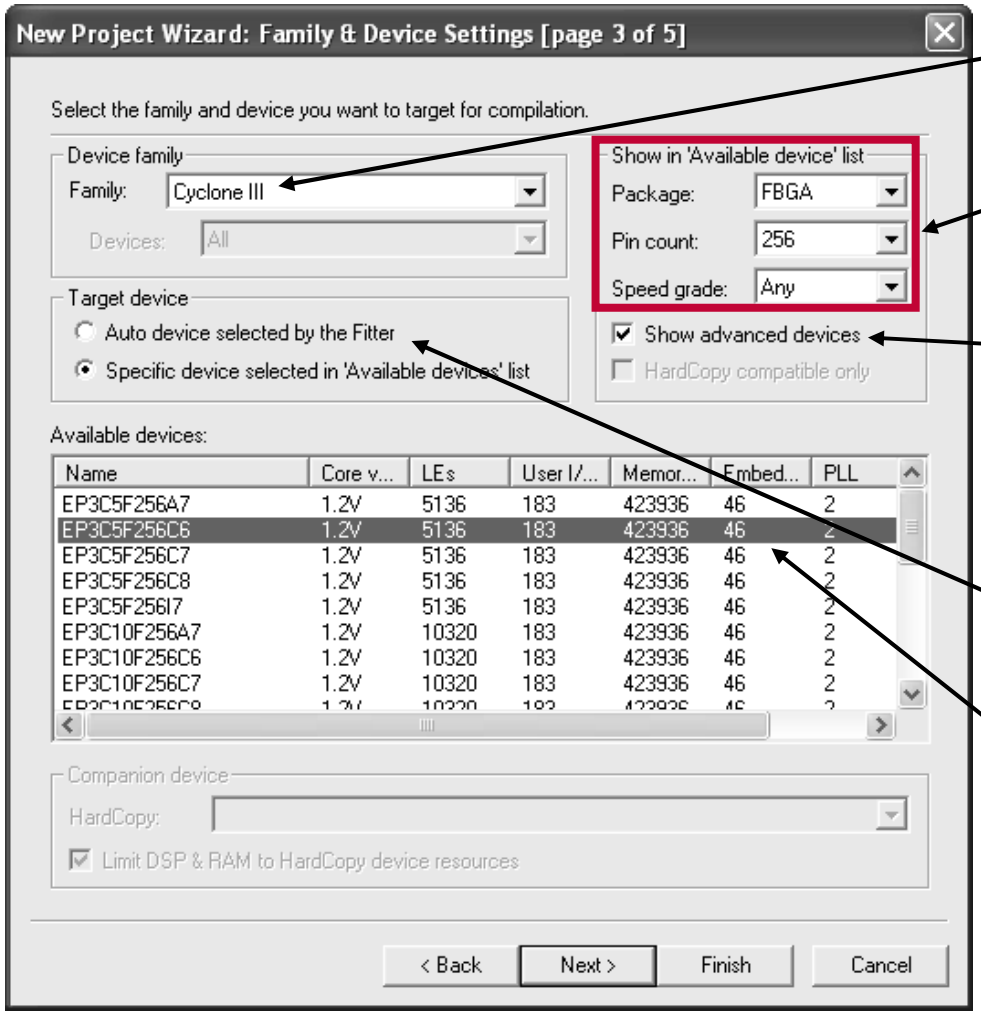
- *Files in project directory do not need to be added*
- *Add top-level file if filename & entity name are not the same*
- *Absolute & relative paths are supported*

Add user library pathnames

- User libraries (any directory containing files)
- MegaCore®/AMPP℠ libraries
- Pre-compiled VHDL packages

Tcl: set_global_assignment –name VHDL_FILE* <filename.vhd>
Tcl: set_global_assignment –name USER_LIBRARIES <library_path_name>
**** Replace with VERILOG_FILE, EDIF_FILE, AHDL_FILE or BDF_FILE***

Device Selection



Choose device family and filter results

Advanced information on future devices

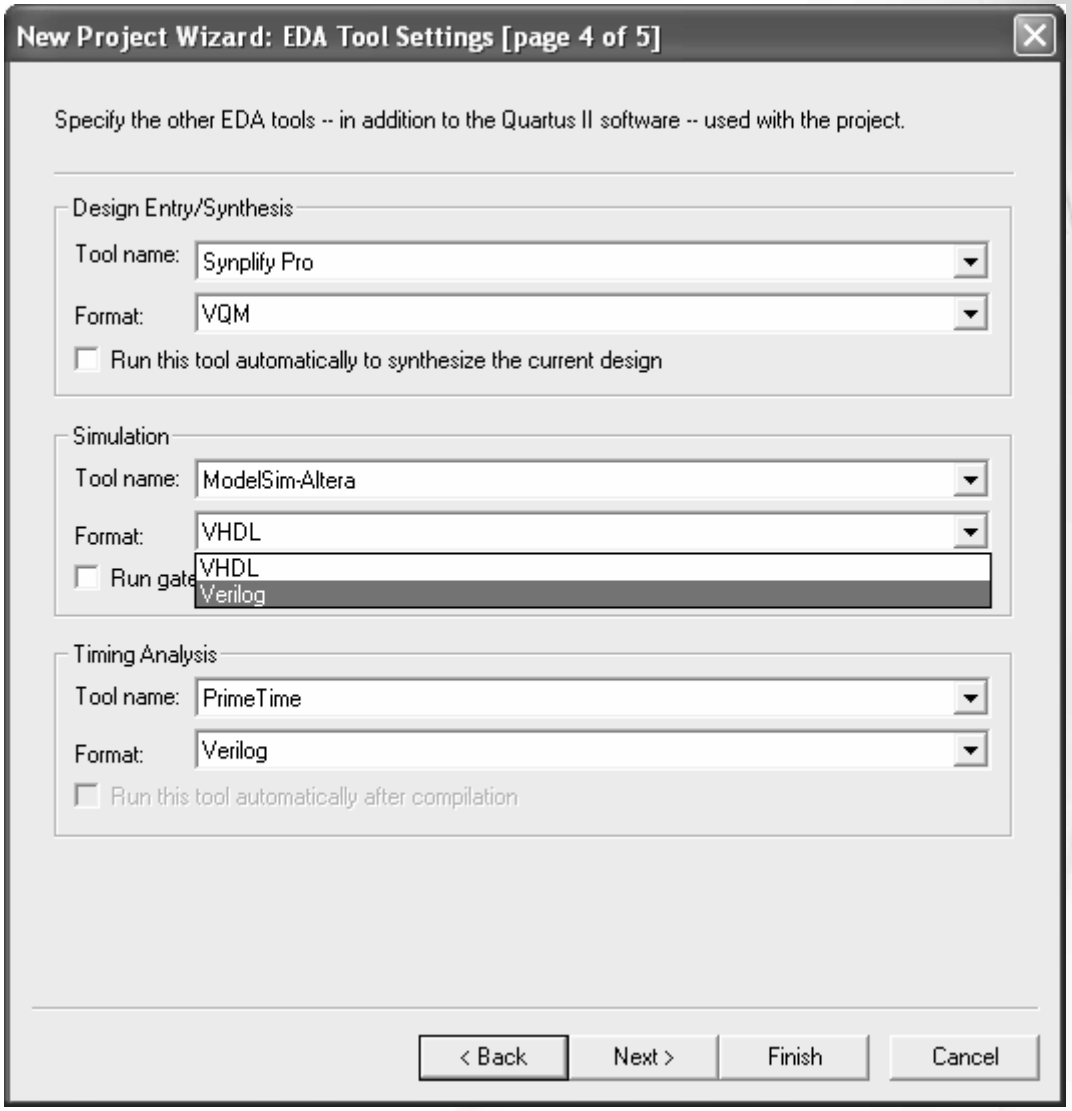
Choose specific part number from list or let Fitter choose smallest, fastest device based on filter criteria

Tcl: set_global_assignment –name FAMILY "device family name"
Tcl: set_global_assignment –name DEVICE <part_number>

EDA Tool Settings

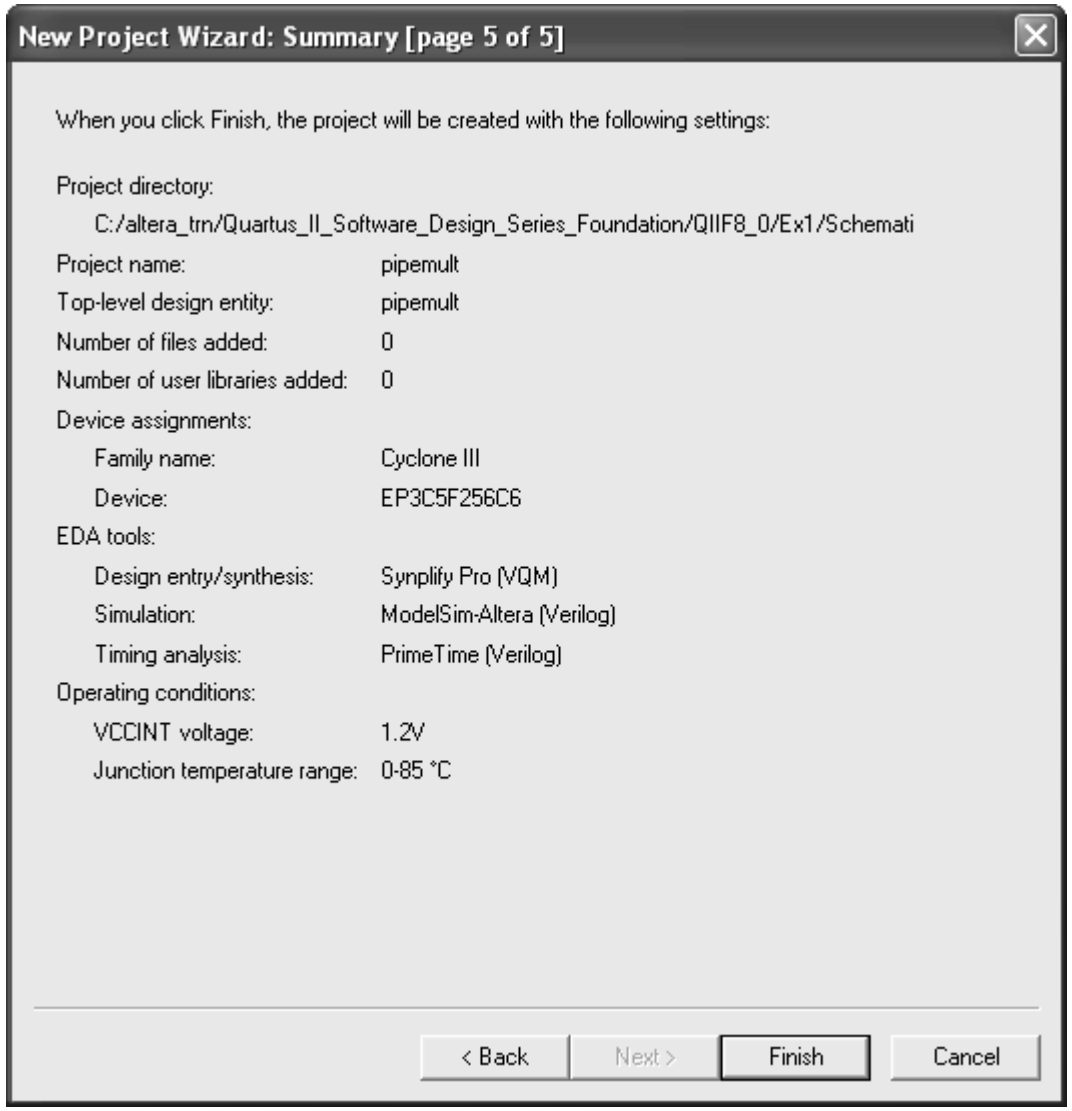
Choose EDA tools & file formats

Add or change settings later



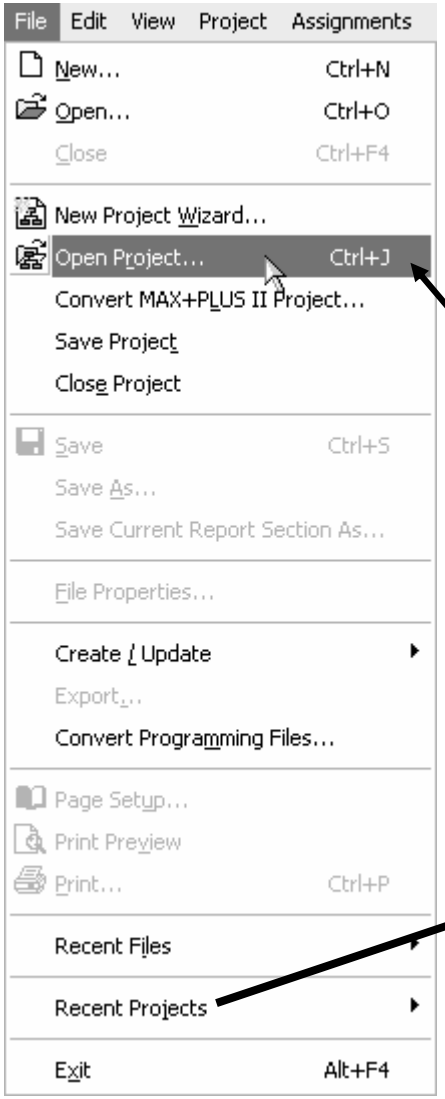
See handbook for Tcl command format

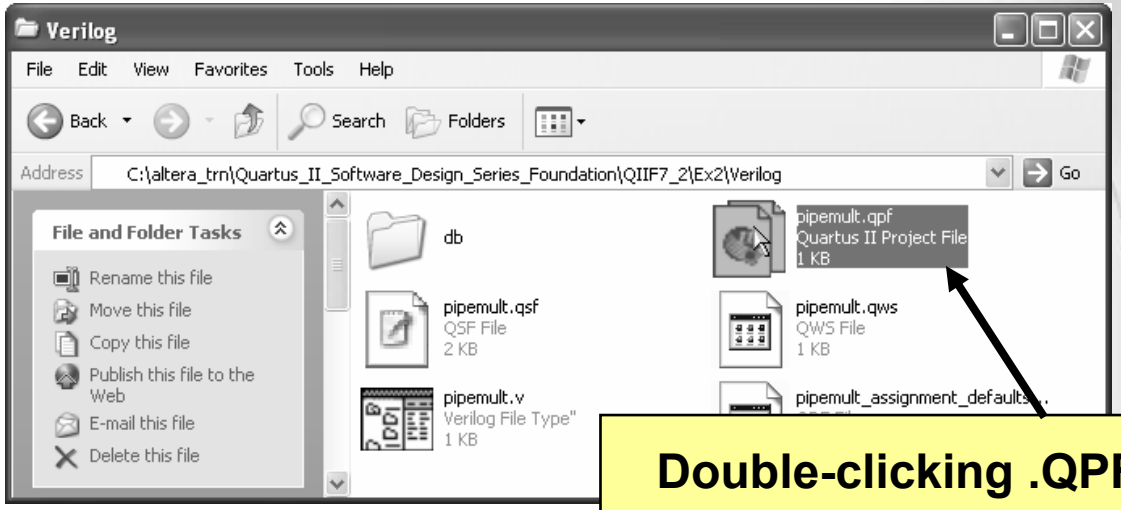
Done!



Review results & click Finish

Opening an Existing Project



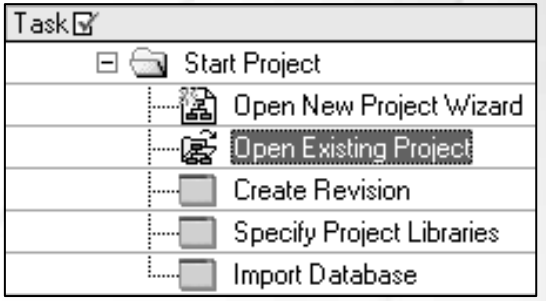


Double-clicking .QPF file auto launches Quartus II software

File ⇒ Open Project… OR

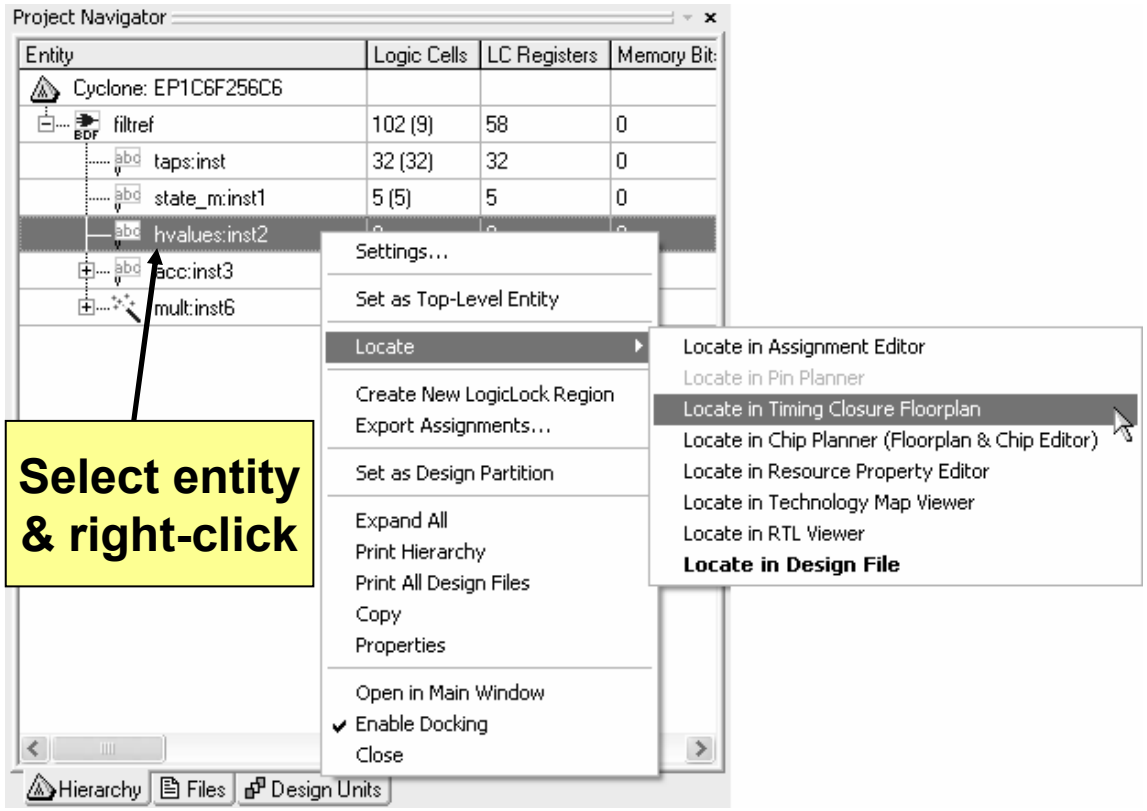
Select from most recent projects list OR Tasks window

1 D:\altera\71\qdesigns\QIIF7_1\Ex7\couner_dsp\top_counter.qpf
2 D:\altera\71\qdesigns\QIIF7_1\Ex7\counter_nios\top_counter.qpf
3 D:\altera\71\qdesigns\fir_filter\fir_filter.qpf
4 D:\altera\71\qdesigns\my_new_project\my_project.qpf
5 D:\altera\71\qdesigns\QIIF7_1\Ex2\VHDL\pipemult.qpf



Tcl: project_open <project_name>

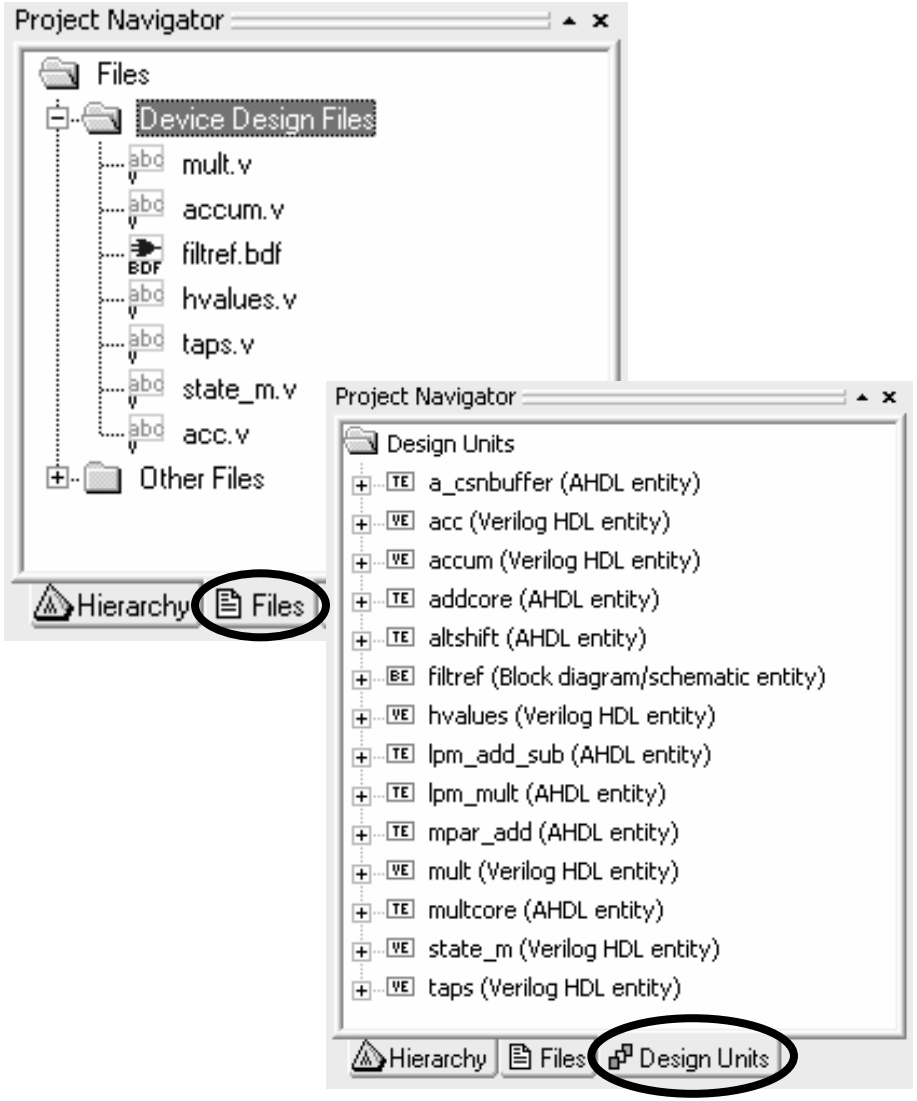
Project Navigator – Hierarchy Tab



- Displays project hierarchy after project is analyzed
- Uses
  - Set top-level entity
  - Set incremental design partition
  - Make entity-level assignments
  - Locate in design file or viewers/floorplans
  - View resource usage

Full compilation or
Processing menu ⇒ Start ⇒ Start Analysis & Elaboration

Files & Design Units Tabs



- Files tab
  - Shows files explicitly added to project
  - Uses
    - Open files
    - Remove files from project
    - Set new top-level entity
    - Specify VHDL library
    - Select file-specific synthesis tool
  - Can also use **Project ⇒ Add/Remove Files in Project…**
- Design Units tab
  - Displays design unit & type
    - VHDL entity
    - VHDL architecture
    - Verilog module
    - AHDL subdesign
    - Block diagram filename
  - Expanded unit displays file which instantiates design unit

Quartus II Project Files & Folders

- Quartus II Project File (.QPF)
- Quartus II Defaults File (.QDF)
- Quartus II Settings File (.QSF)
- db folder
  - Contains compiled design information
- Synopsys Design Constraints (.SDC)
  - Holds timing constraints
  - Discussed later

Project & Default Files

- Quartus II Project File (QPF)
  - Quartus II version
  - Time stamp
  - Active revision(s)

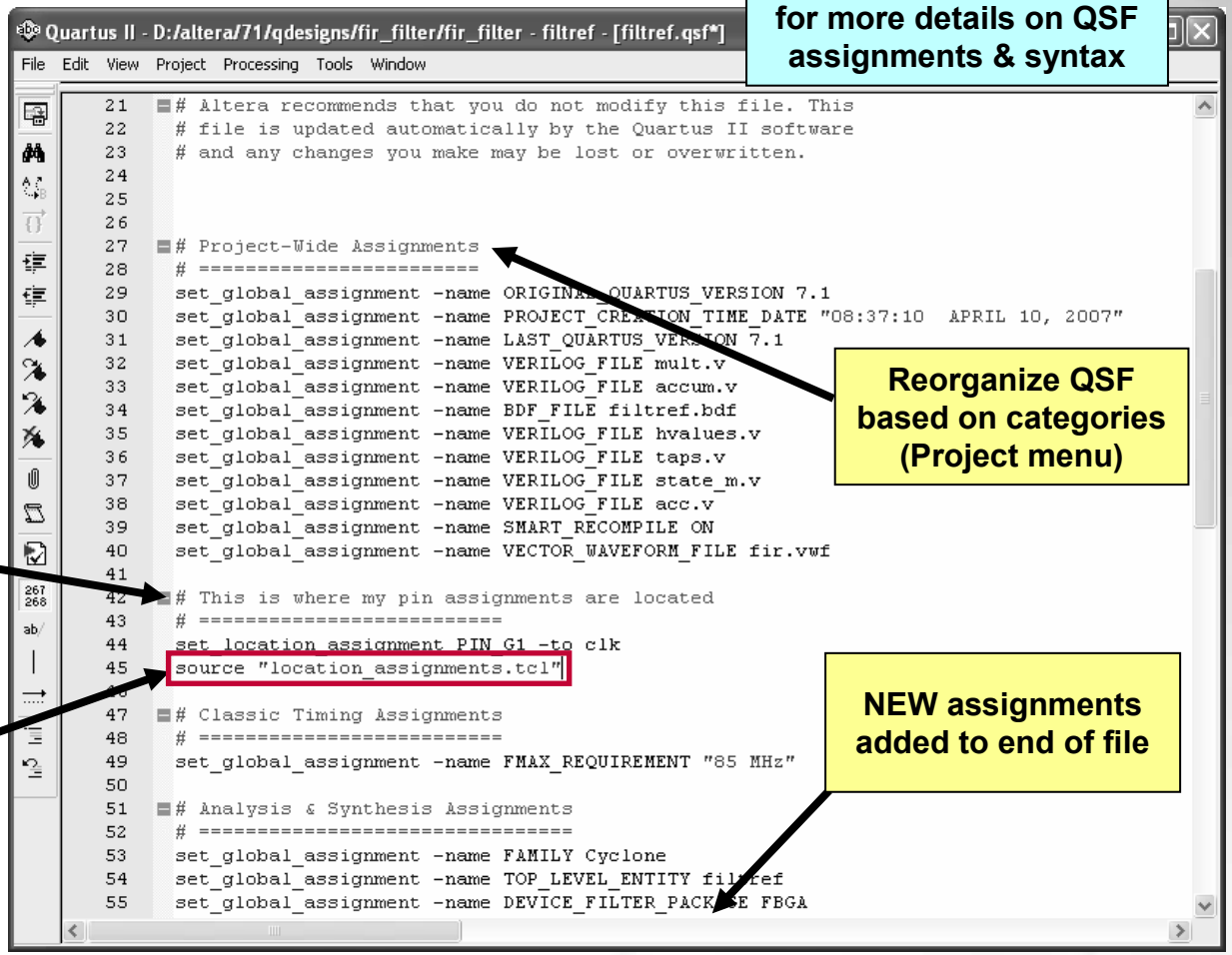
fir_filter.QPF

```
QUARTUS_VERSION = "8.0"
DATE = "08:37:10  June 19, 2008"


# Active Revisions

PROJECT_REVISION = "filtref"
PROJECT_REVISION = "filtref_new"
```

- Quartus II Defaults Files (QDF)
  - Stores Quartus II project setting & assignment defaults
  - Example names: *assignment_defaults.qdf* or *<revision_name>_assignment_defaults.qdf*
  - Found in local project or *altera\<version>\quartus\bin* directory
    - Copy stored in local project directory read before original version in bin

Quartus II Settings File (QSF)

- Stores all settings & assignments (constraints)
- Uses Tcl syntax
- Can be edited manually by user



Note: See Appendix for more notes on using QSF file.

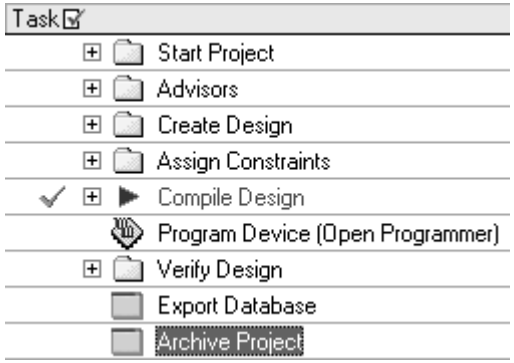
Project Archive

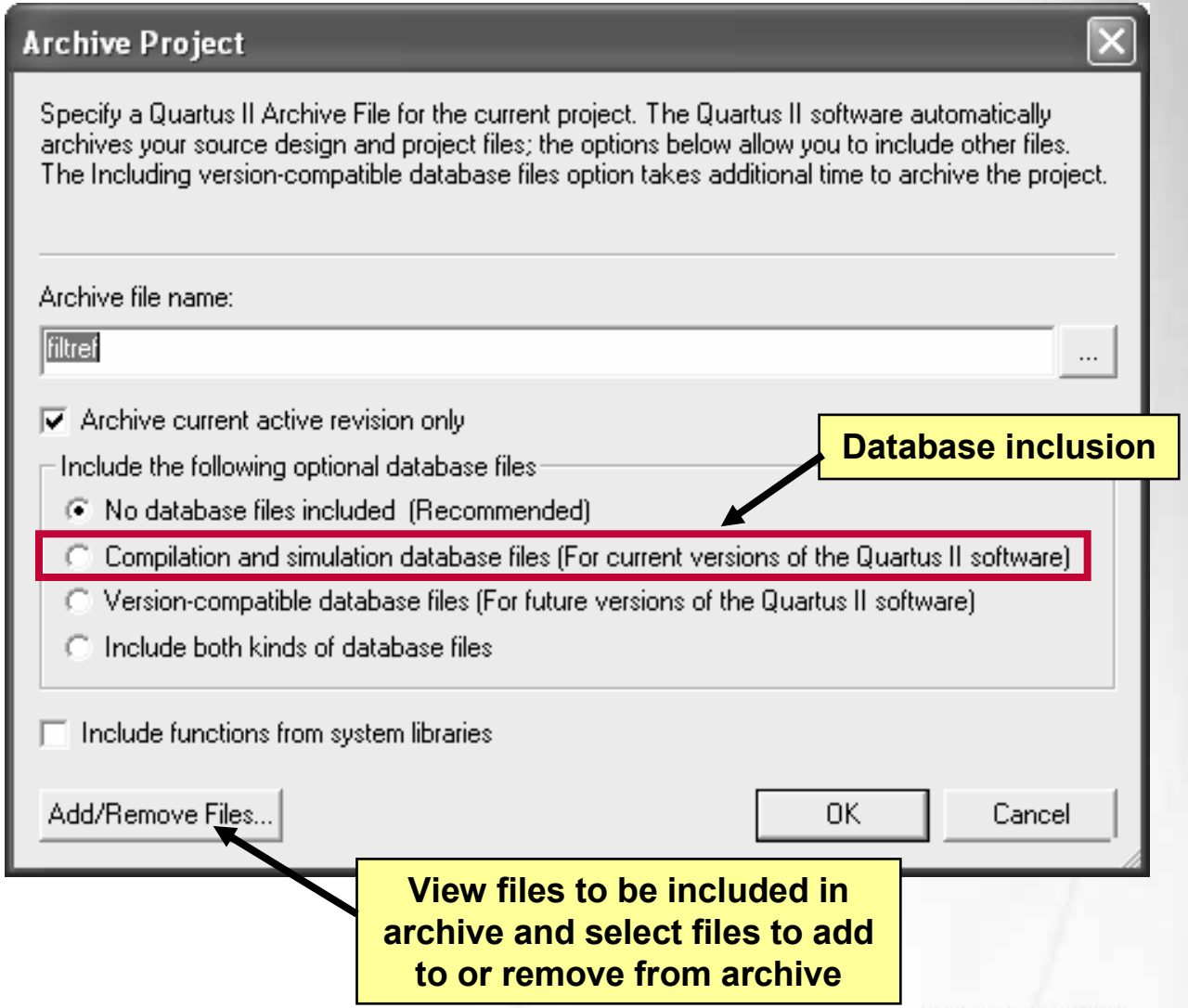
- Creates 2 files
  - Compressed Quartus II Archive File (.QAR)
    - Includes design files, QPF file, & QSF file(s)
    - Option to include databases (db folder in project directory)
      - Recompile necessary if databases not included
    - Creates local QDF file for archive
  - Archive activity log (.QARLOG)
- Example uses
  - File storage (e.g. version control)
  - Project handoff
    - Useful for sending to Altera support
- Design files referenced from user libraries are included in archive

Tcl: project_archive <project_name>

Project Archive (cont.)

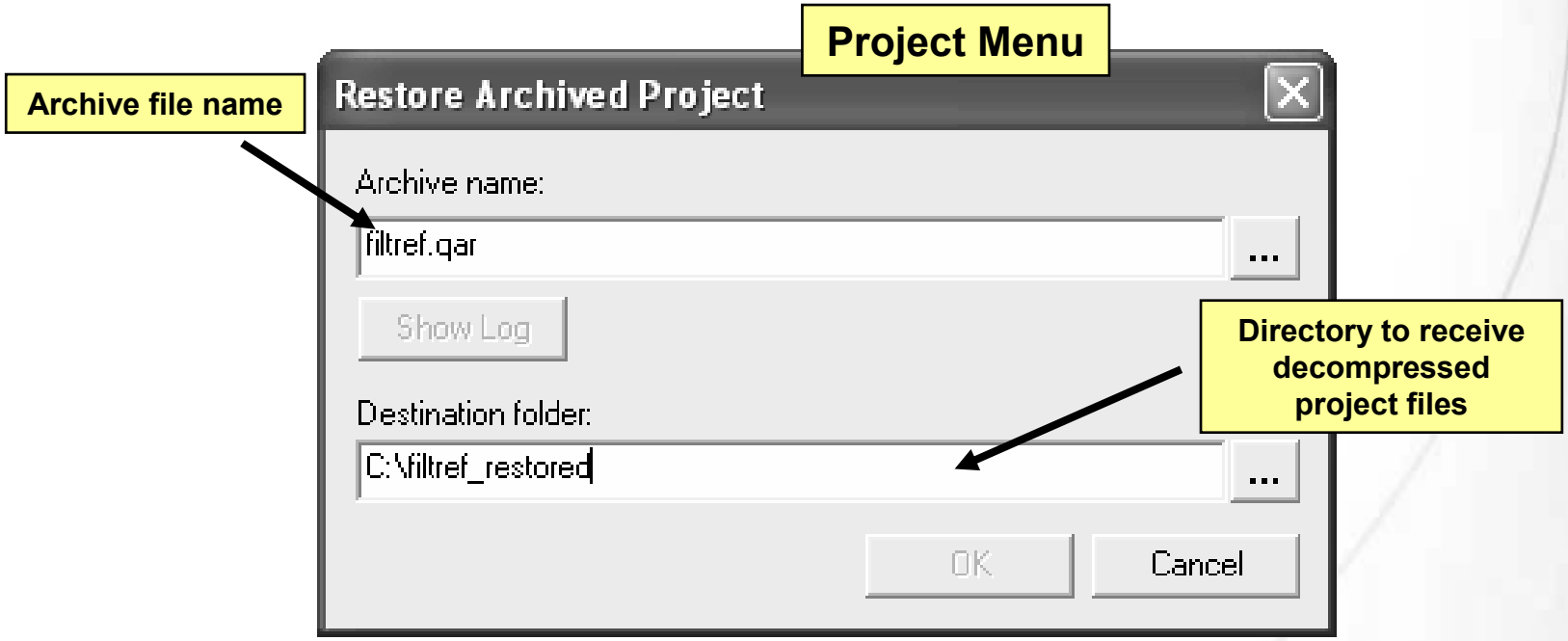
Project Menu or Tasks window





Project Restore

- Decompresses .QAR into specified directory



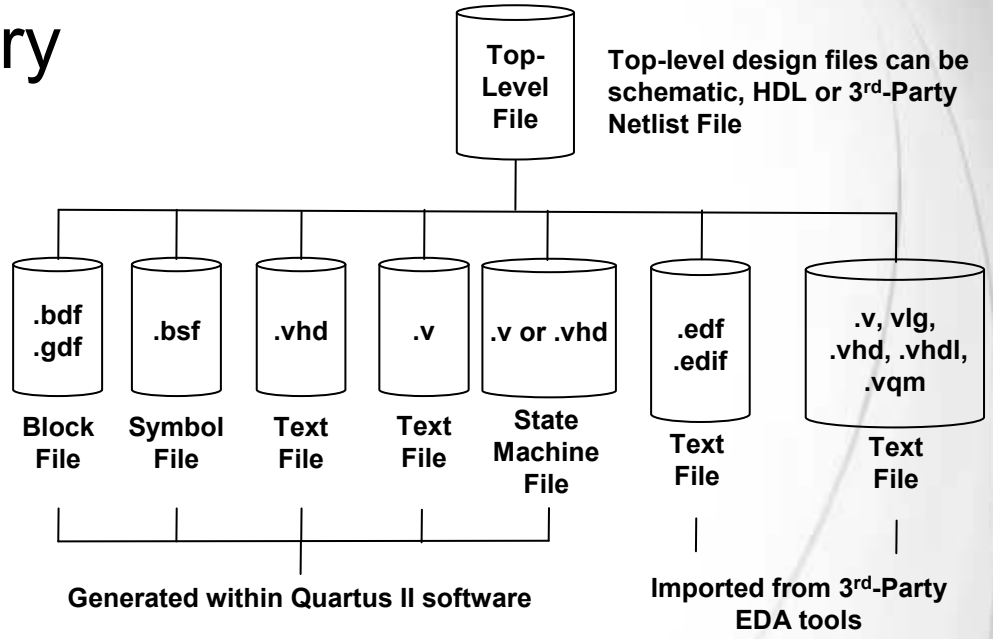
Tcl: project_restore <archive_file>

Quartus II Software Design Series: Foundation

Design Entry

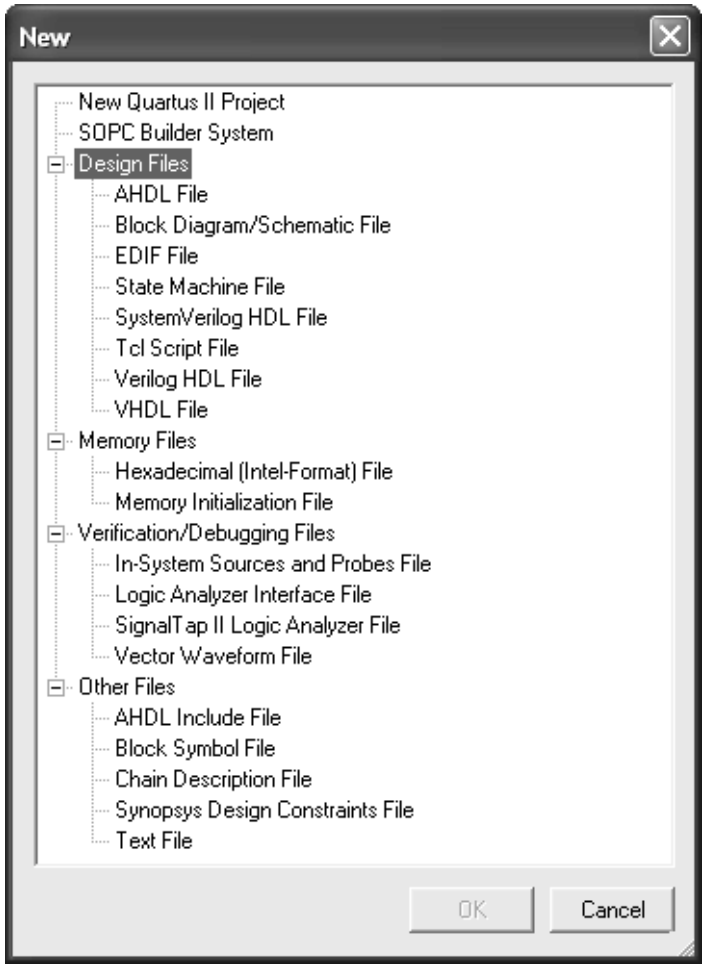
Design Entry Methods

- Quartus II design entry
  - Text editor
    - VHDL
    - Verilog
  - Schematic editor
    - Block Diagram File
    - Graphic Design File
  - State machine editor
    - HDL from state machine file
  - Memory editor
    - HEX
    - MIF
- 3rd-party EDA tools
  - EDIF 2 0 0
  - Verilog Quartus Mapping (.VQM)
- Mixing & matching design files allowed

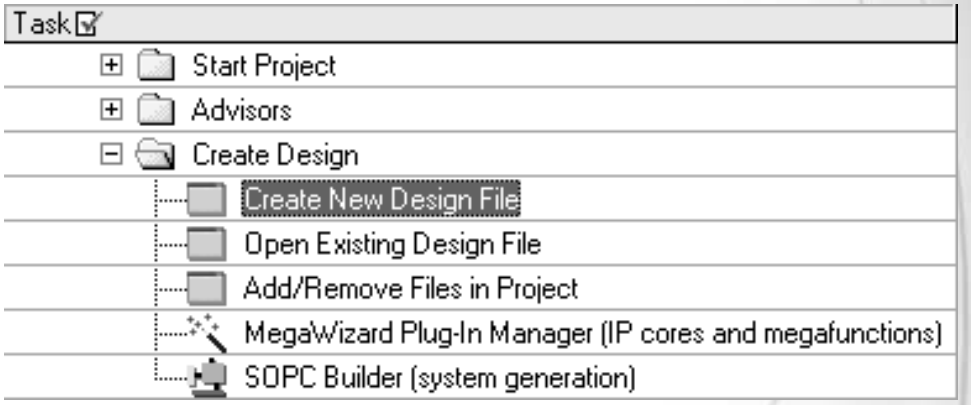


Creating New Design Files (& Others)

File ⇒ New or in Toolbar



Tasks window



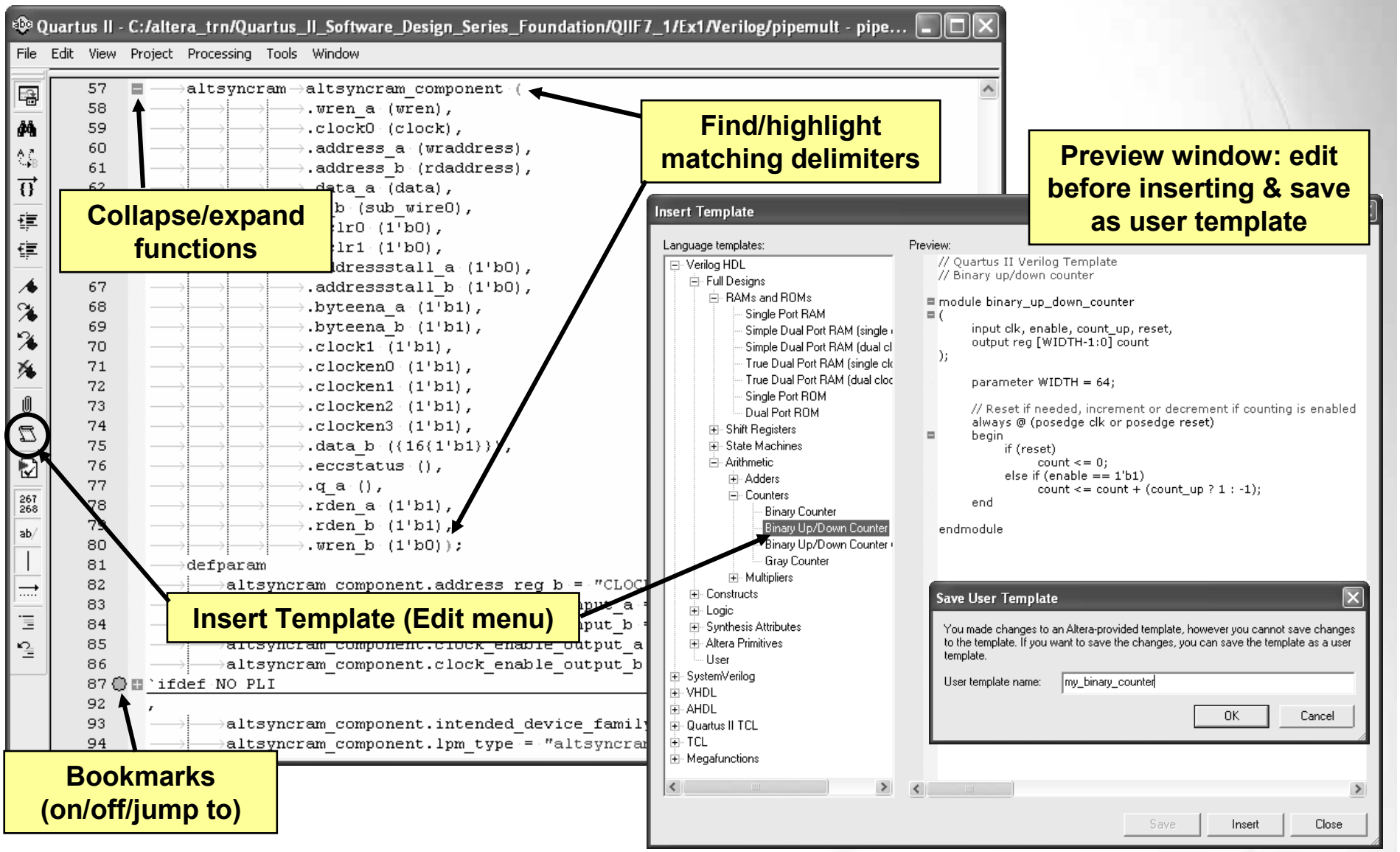
Create new files or start New Project Wizard

Verilog & VHDL

- **VHDL**- VHSIC hardware description language
  - IEEE Std 1076 (1987 & 1993) supported
  - IEEE Std 1076.3 (1997) synthesis packages supported
- **Verilog**
  - IEEE Std 1364 (1995 & 2001) & 1800 (SystemVerilog) supported
- Create in the Quartus II editor or any standard text editor
  - Select different text editor app with **Tools** ⇒ **Options for 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

Text Editor Features



Schematic Design Entry

- Full-featured schematic design capability
- Schematic Editor uses
  - Create simple test designs to understand the functionality of an Altera megafunction
    - PLL, LVDS I/O, memory, etc…
  - Create top-level schematic for easy viewing & connection
    - Convert Block Diagram File (.BDF) to HDL file (VHDL/Verilog) or image file (.JPG or .BMP) (**File ⇒ Create/Update**)
    - Convert HDL file to schematic block symbol file (.BSF; black box) (**File ⇒ Create/Update**)

Note: ***Please see the Appendix for a more detailed discussion of the Block Diagram Editor and schematic entry.***
Online training available: *Using the Quartus II Software: Schematic Design*

Altera Megafunctions

- Pre-made design blocks
- Benefits
  - Configurable settings add flexibility
  - “Drop-in” support to accelerate design entry
  - Pre-optimized for Altera architecture
- Two versions
  - Quartus II megafunctions
  - Intellectual Property (IP) megafunctions

Quartus II Megafunctions

- Free & installed with Quartus II software
  - Non-encrypted functions written in AHDL
  - HDL simulation models installed in Quartus II libraries
- Two types
  - Altera-specific megafunctions (begin with “ALT”)
  - Library of parameterized modules (LPMs)
    - Industry standard logic functions
    - See www.edif.org/lpmweb (EDIF.org archive) for more info
- Examples
  - Multiply-accumulate (ALTMULT_ACCUM)
  - On-chip RAM/ROM (ALTSYNCRAM)
  - PLL (ALTPLL)
  - DDR/QDR memory interface (ALTMEMPHY)
  - Counter (LPM_COUNTER)
  - Comparator (LPM_COMPARE)

IP Megafunctions

- Must purchase license to use in finished design
  - Logic for IP function is encrypted
- Two types
  - MegaCore® IP
    - Developed by Altera
    - Install with Quartus II software or download/install individually from www.altera.com
  - Altera Megafunctions Partner Program (AMPP℠) IP
    - Developed by 3rd-Party IP vendors & certified by Altera
    - Contact vendor for evaluating and licensing function
- All MegaCore functions & some AMPP functions support OpenCore® Plus feature
  - Develop design using free version of core
  - HDL simulation models provided with IP
  - Generate time-limited configuration/programming files
  - See *AN320: OpenCore Plus Evaluation of Megafunctions*

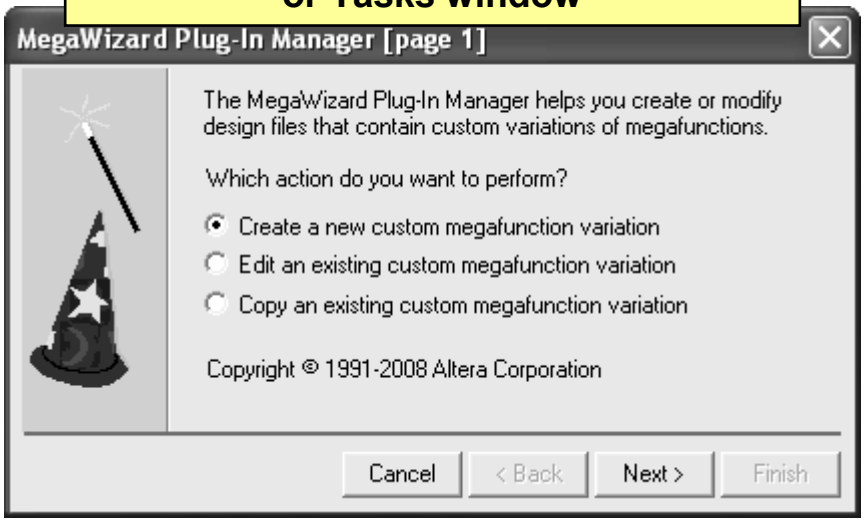
Example MegaCore IP

- Triple-Speed Ethernet MAC
- FIR Compiler
- Fast Fourier Transform
- DDR/DDR2 High Performance Memory Controller
- CRC Compiler
- PCI Compiler

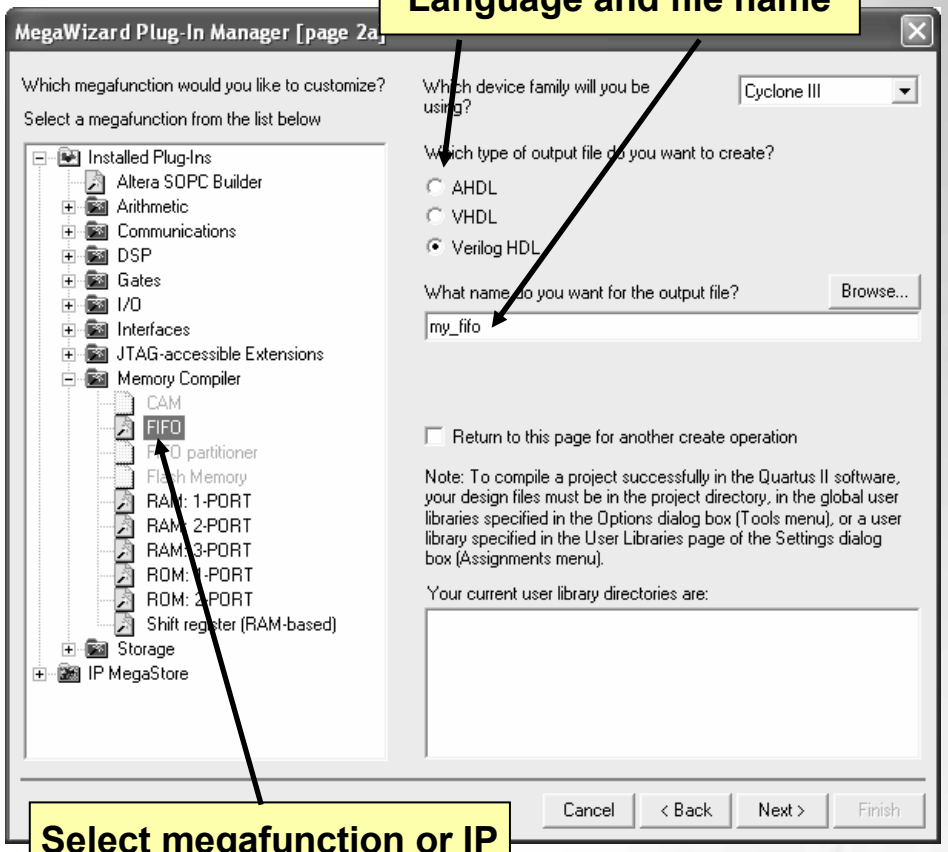
MegaWizard Plug-in Manager

- Eases implementation and configuration of megafunctions & IP
- GUI, command line or both

Tools ⇒ MegaWizard Plug-In Manager or Tasks window



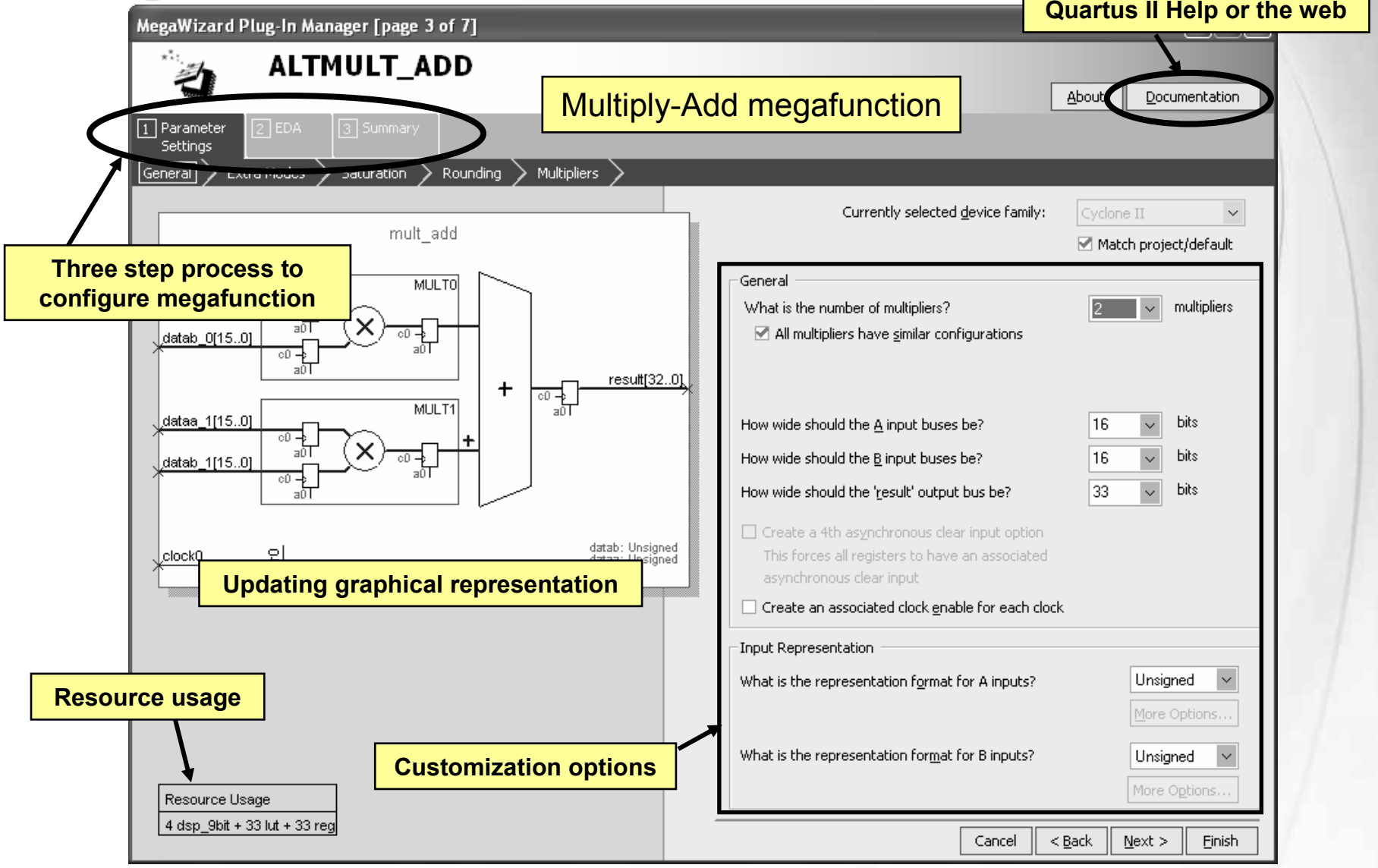
Language and file name



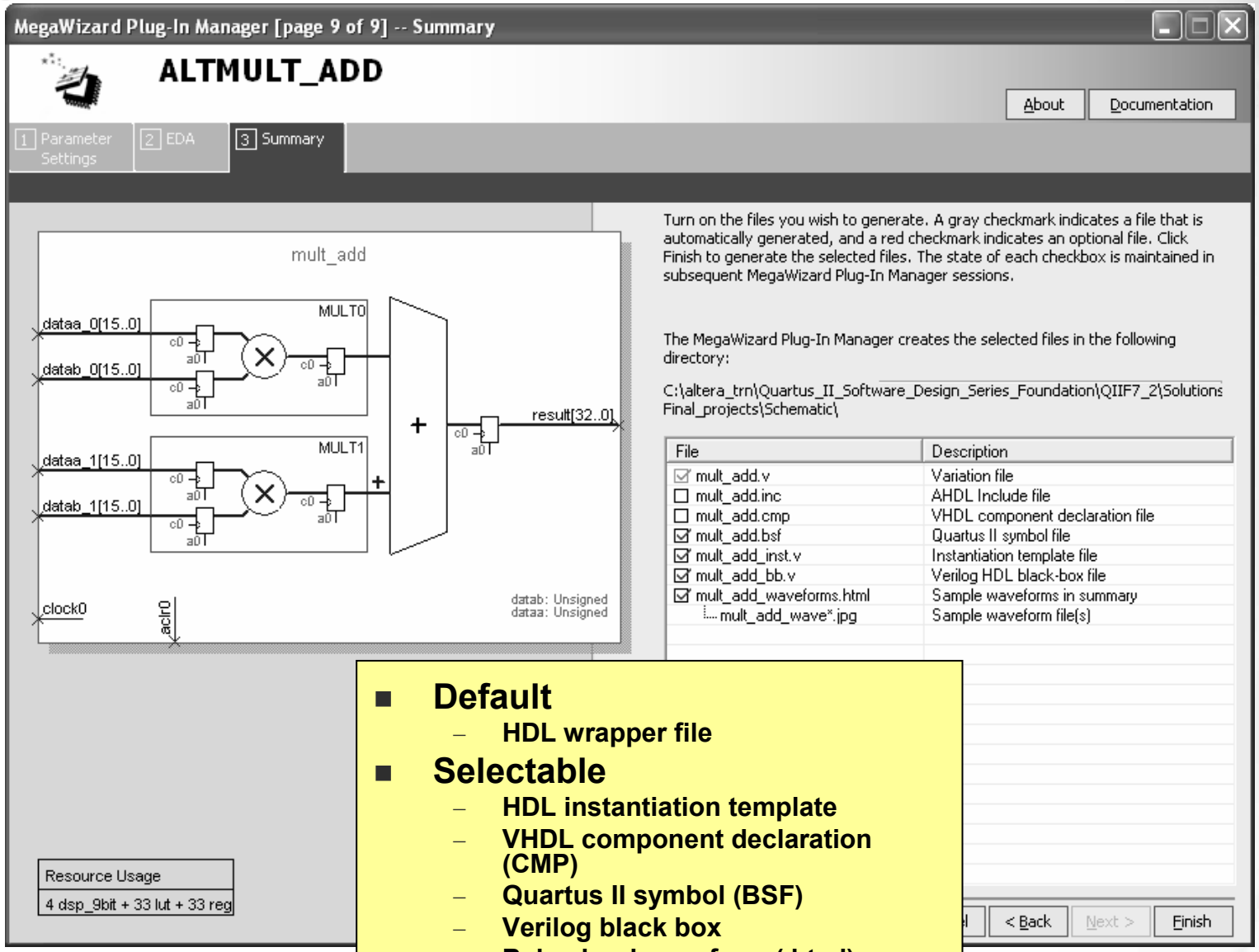
Select megafunction or IP

Command line: qmegawiz <-silent> <module | wizard>=<mf_name> <ports & parameters options> file_name

MegaWizard Example



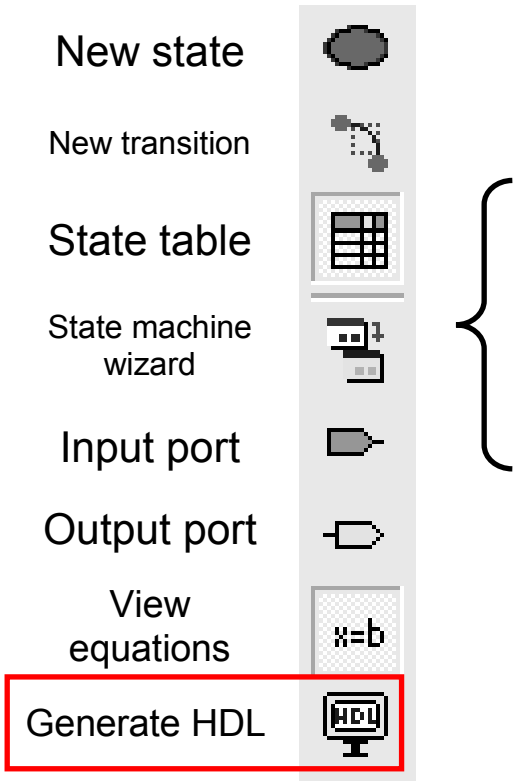
MegaWizard Output File Selection

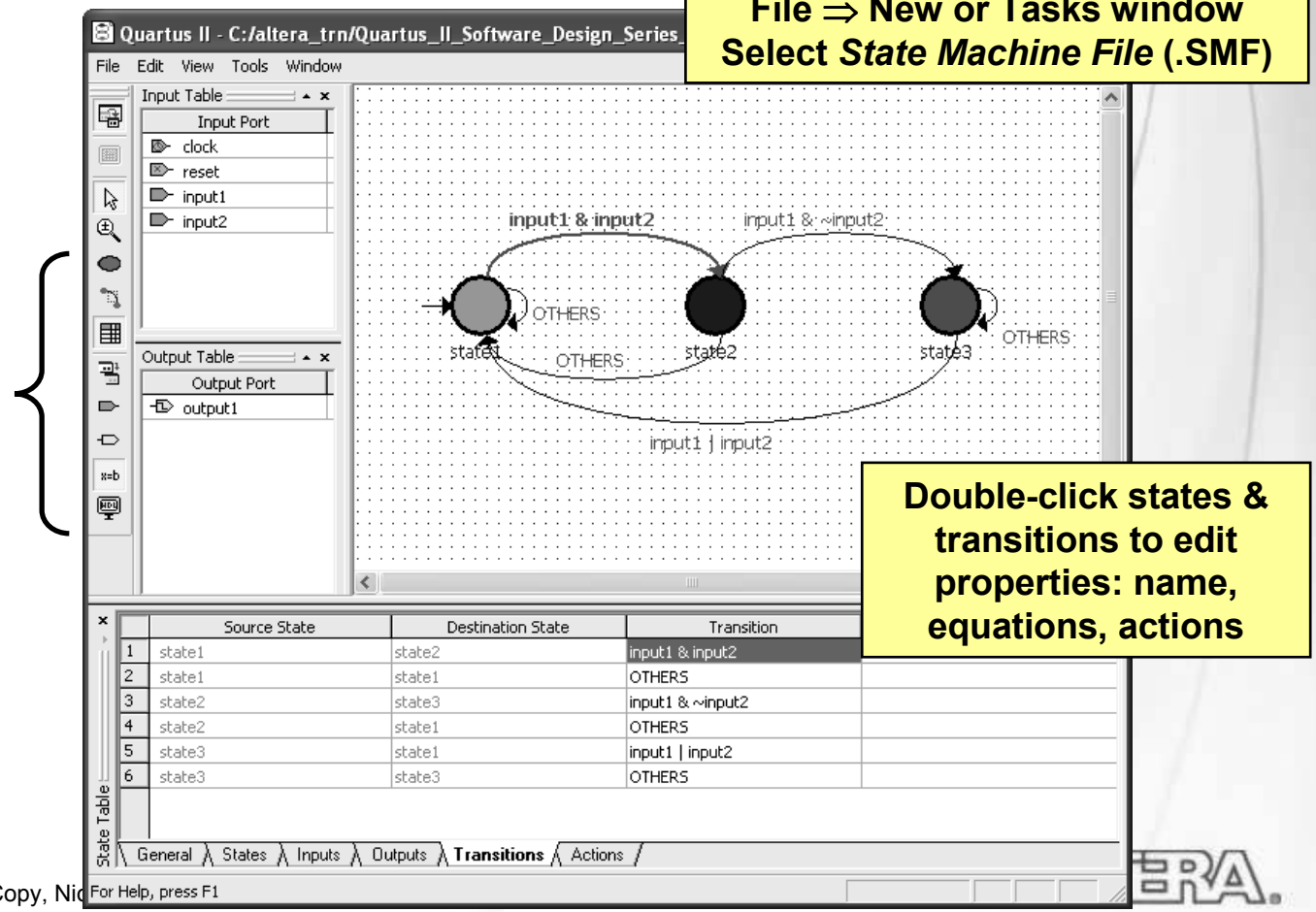


- **Default**
  - **HDL wrapper file**
- **Selectable**
  - **HDL instantiation template**
  - **VHDL component declaration (CMP)**
  - **Quartus II symbol (BSF)**
  - **Verilog black box**
  - **Behavioral waveform (.html)**

State Machine Editor

- Create state machines in GUI
  - Manually by adding individual states and transitions
  - Automatically with State Machine Wizard (**Tools** menu)
- Generate state machine HDL code (required)



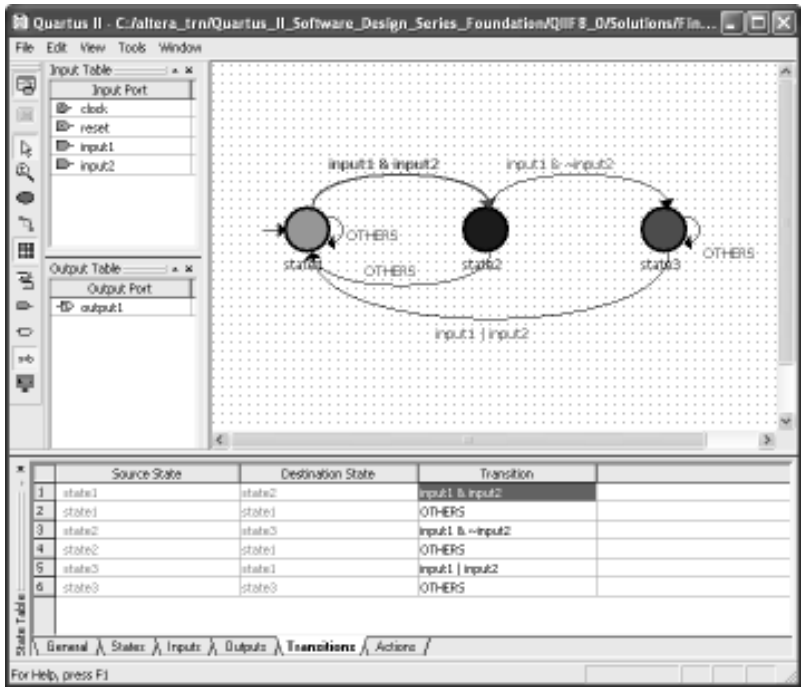


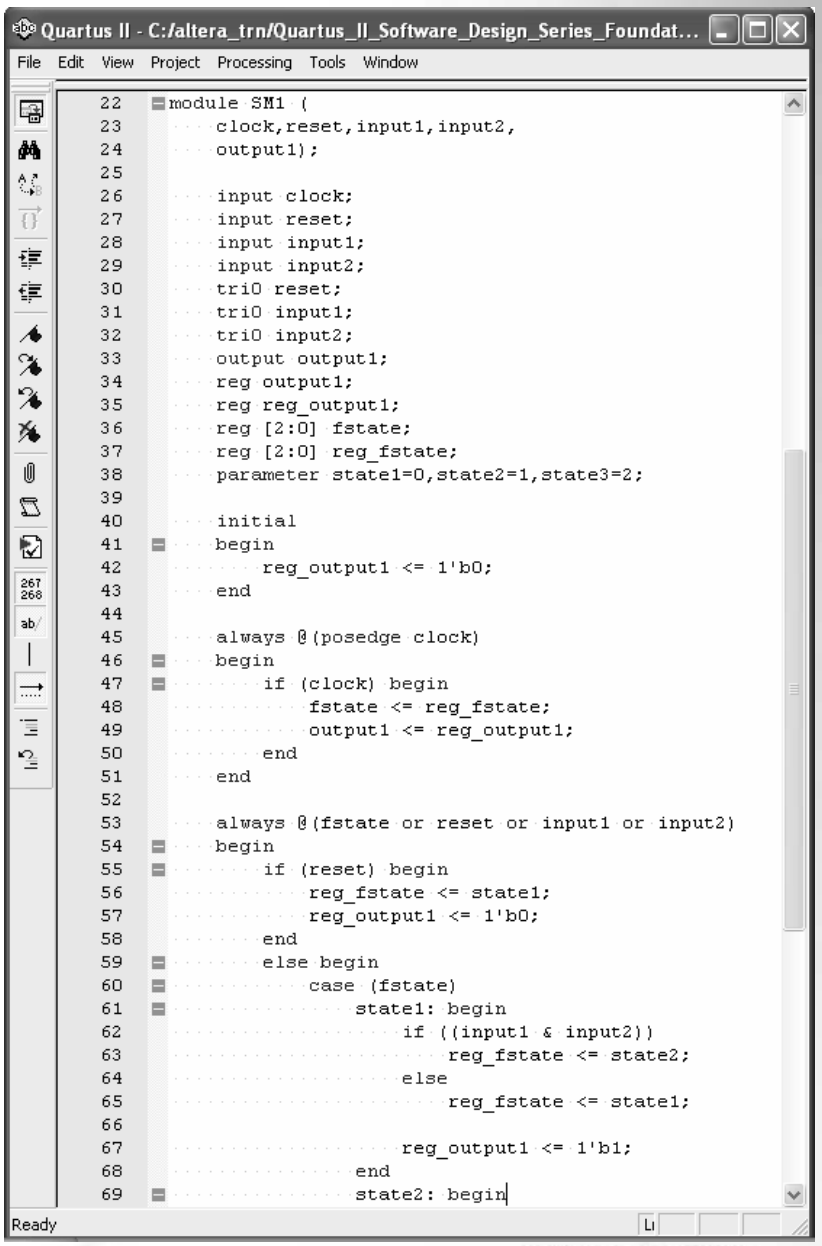
File ⇒ New or Tasks window
Select *State Machine File* (.SMF)

Double-click states & transitions to edit properties: name, equations, actions

From .SMF to HDL

- Generate optimized code (Verilog or VHDL)
- Add to project
- **Required for use**



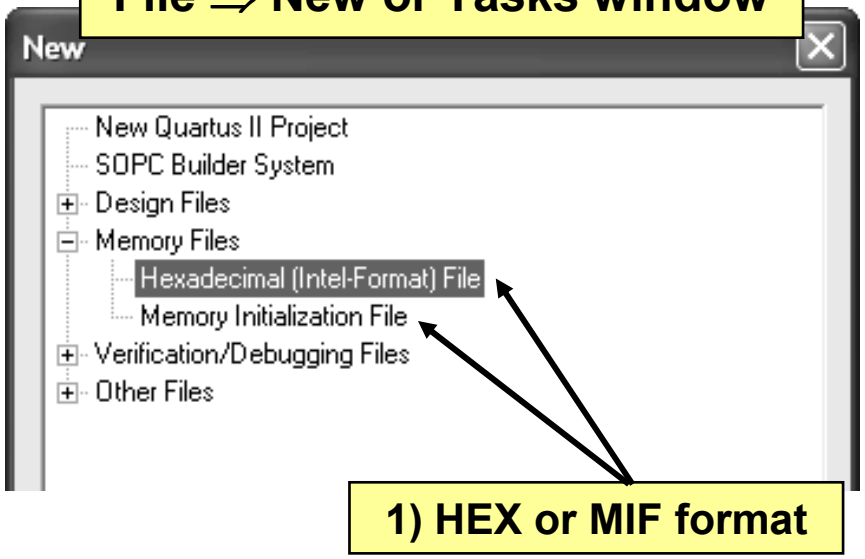


Memory Editor

- Create or edit memory initialization files in Intel HEX (.HEX) or Altera-specific (.MIF) format
- Design entry
  - Use to initialize memory blocks (ex. RAM, ROM) during power-up
  - Initialization file data sent to device during device programming
- Simulation
  - Use to initialize memory blocks before simulation or after breakpoints

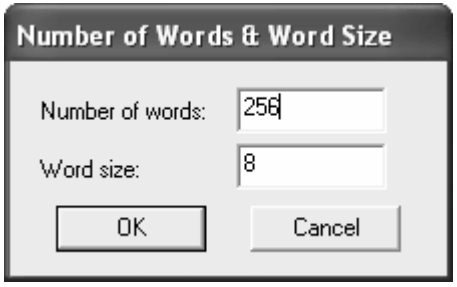
Create Memory Initialization File

File ⇒ New or Tasks window

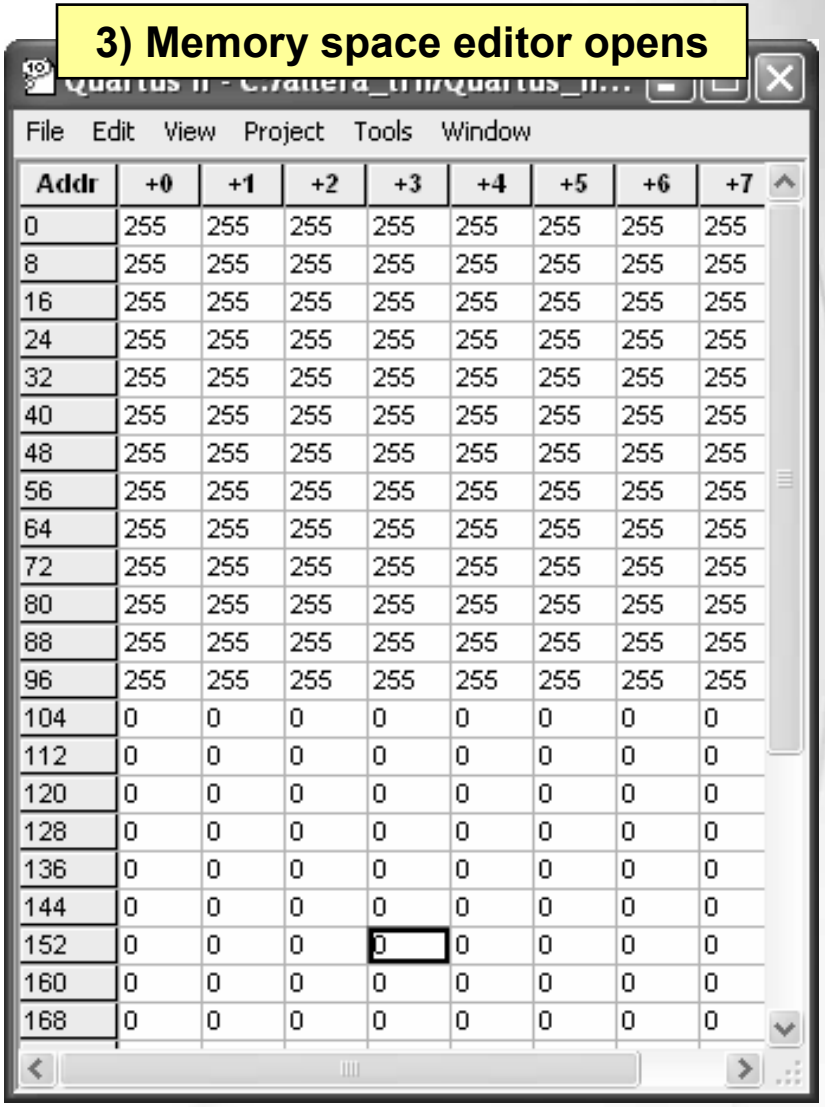


1) HEX or MIF format

2) Select memory size

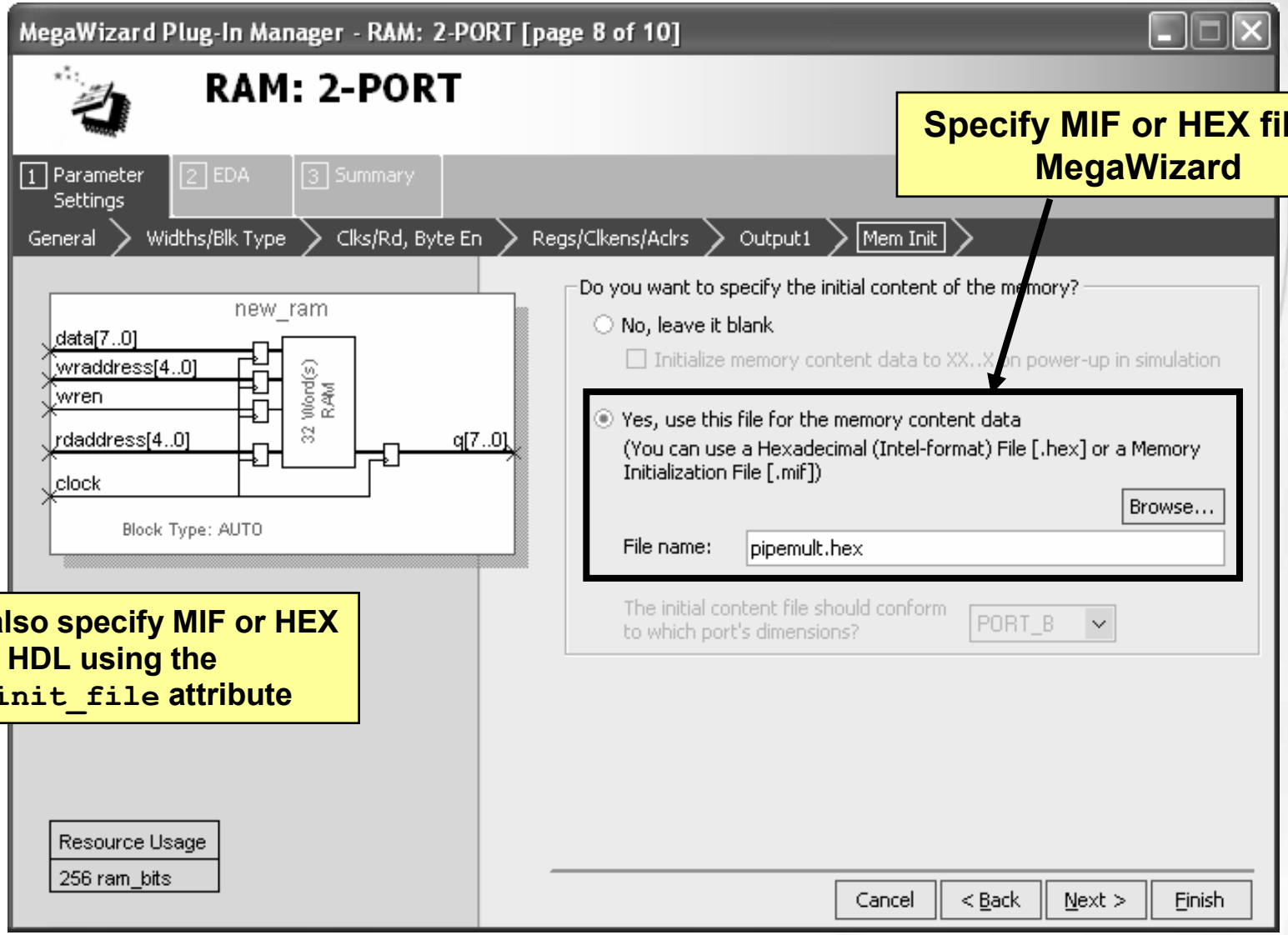


3) Memory space editor opens



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

Using Memory File In Design



Specify MIF or HEX file in MegaWizard

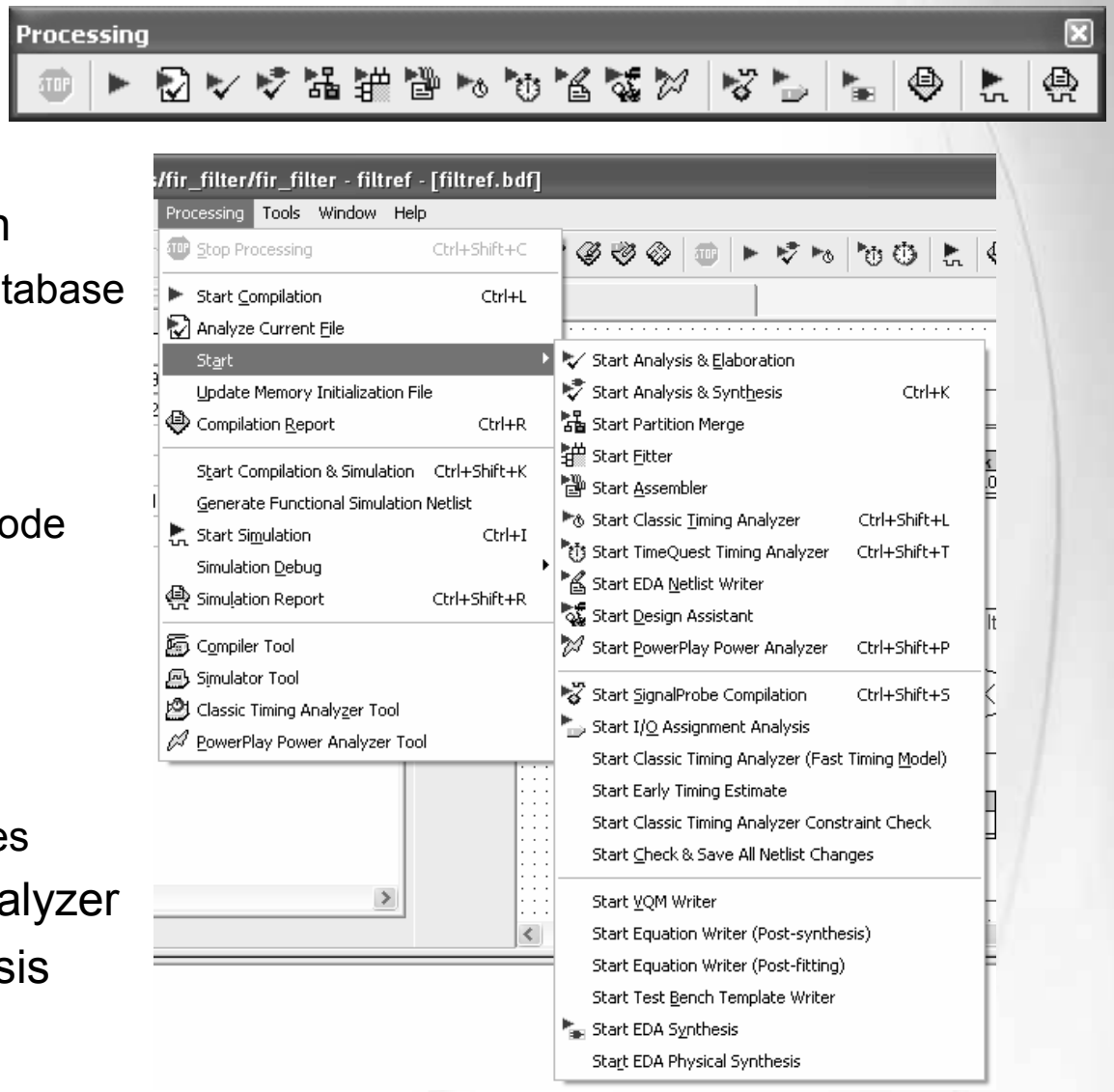
May also specify MIF or HEX file in HDL using the `ram_init_file` attribute

Quartus II Software Design Series: Foundation

Quartus II Compilation

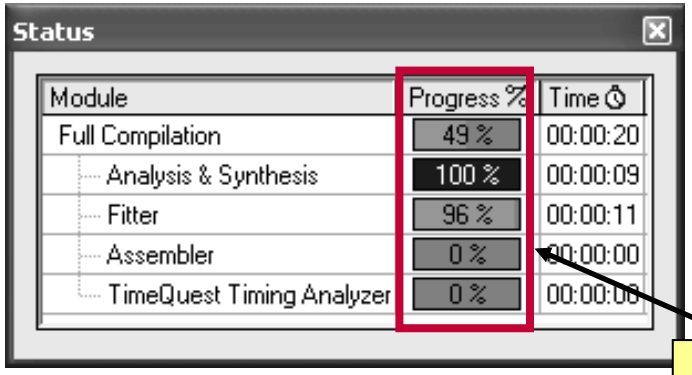
Processing Options

- Start Compilation
  - Performs full compilation
- Start Analysis & Elaboration
  - Checks syntax & builds database only
  - Performs initial synthesis
- Start Analysis & Synthesis
  - Synthesizes & optimizes code
- Start Fitter
  - Places & routes design
  - Generates output netlists
- Start Assembler
  - Generate programming files
- Start TimeQuest Timing Analyzer
- Start I/O Assignment Analysis
- Start Design Assistant

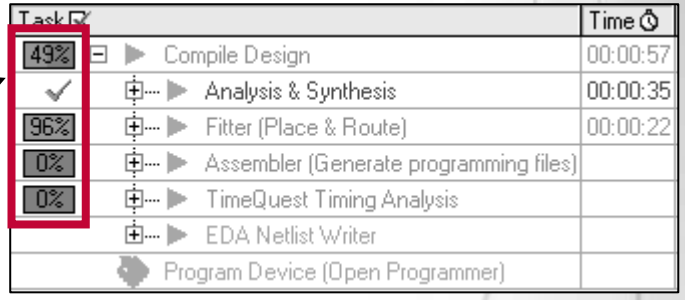


Status & Message Windows

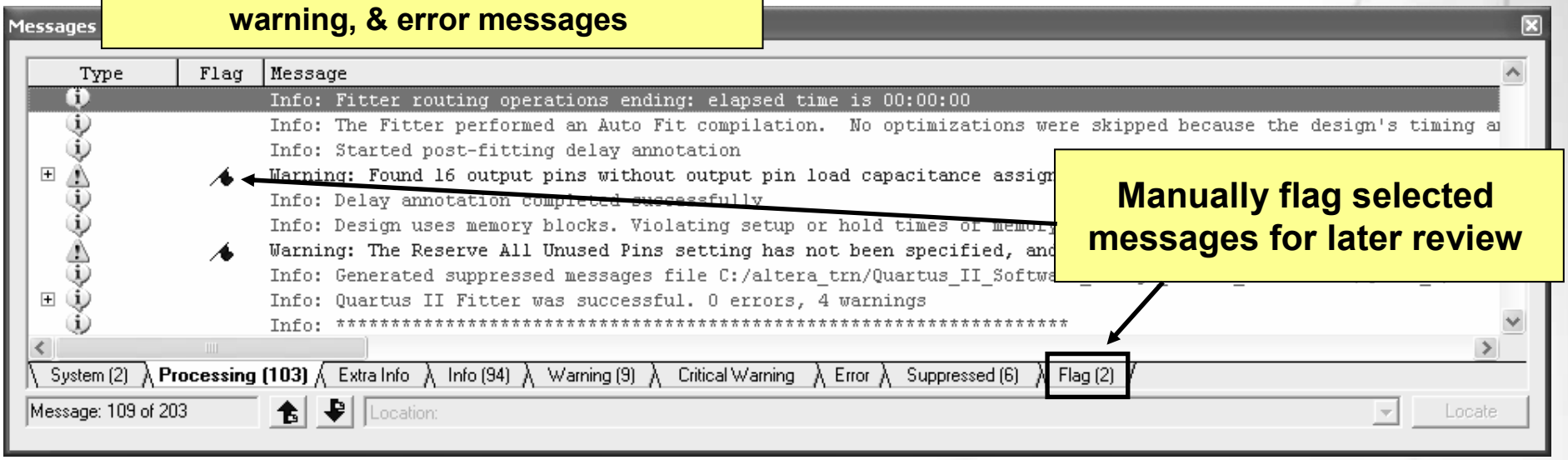
View menu ⇒ Utility Windows



Status bars indicate compilation progress



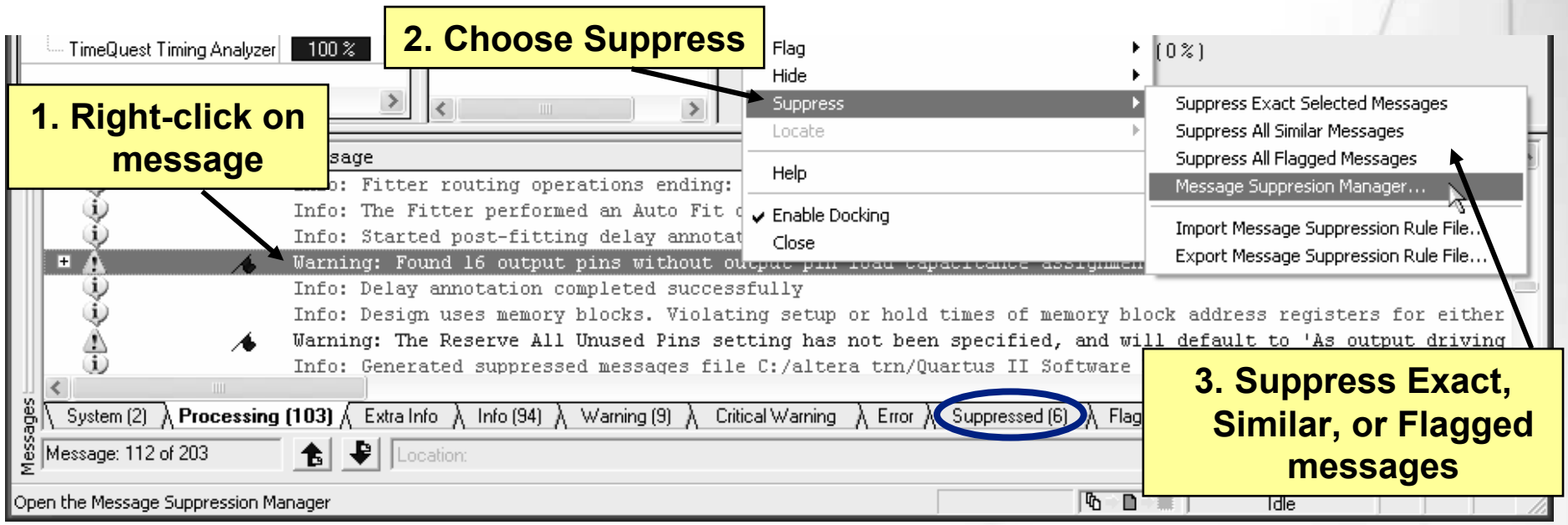
Message window displays informational, warning, & error messages



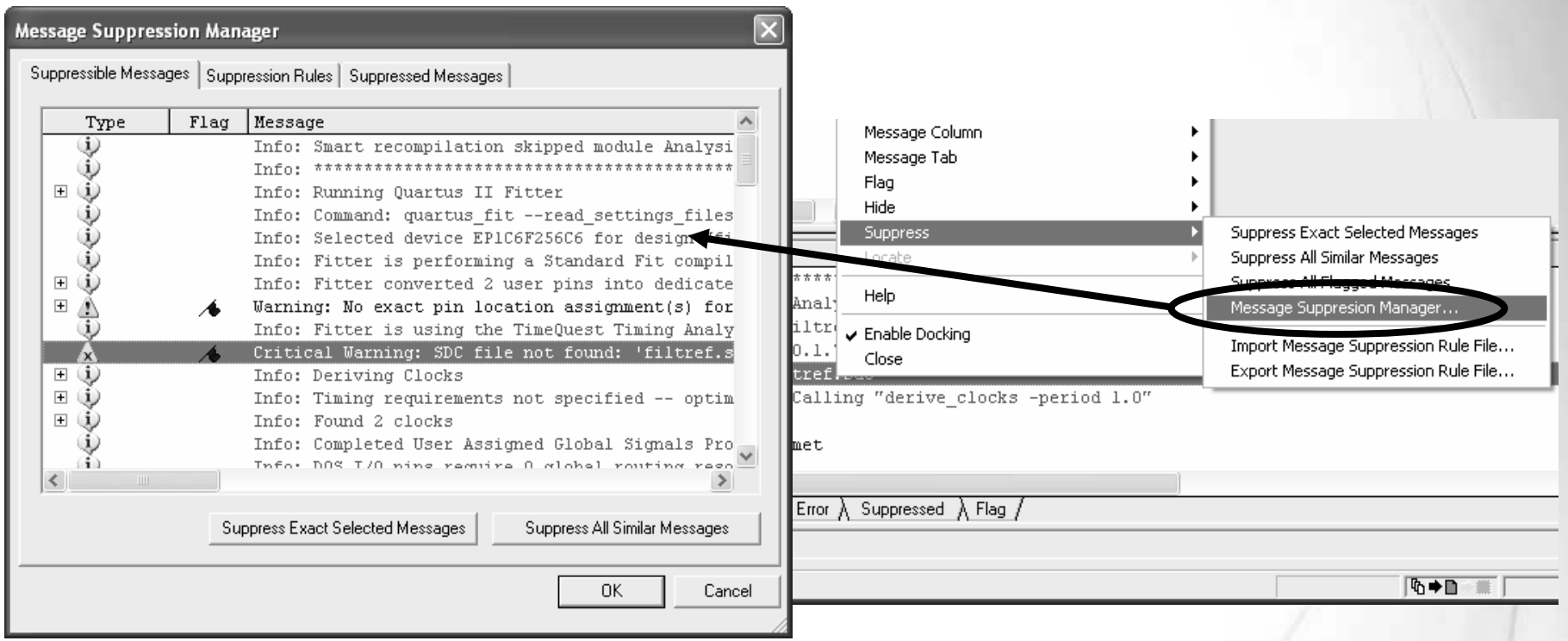
Manually flag selected messages for later review

Message Suppression

- Hides messages from current & future compiles
  - Ex. Known synthesis warning message already investigated
- Displays suppressed messages on different tab in message window
- Stores suppression rules in <revision_name>.SRF file



Message Suppression Manager Tool



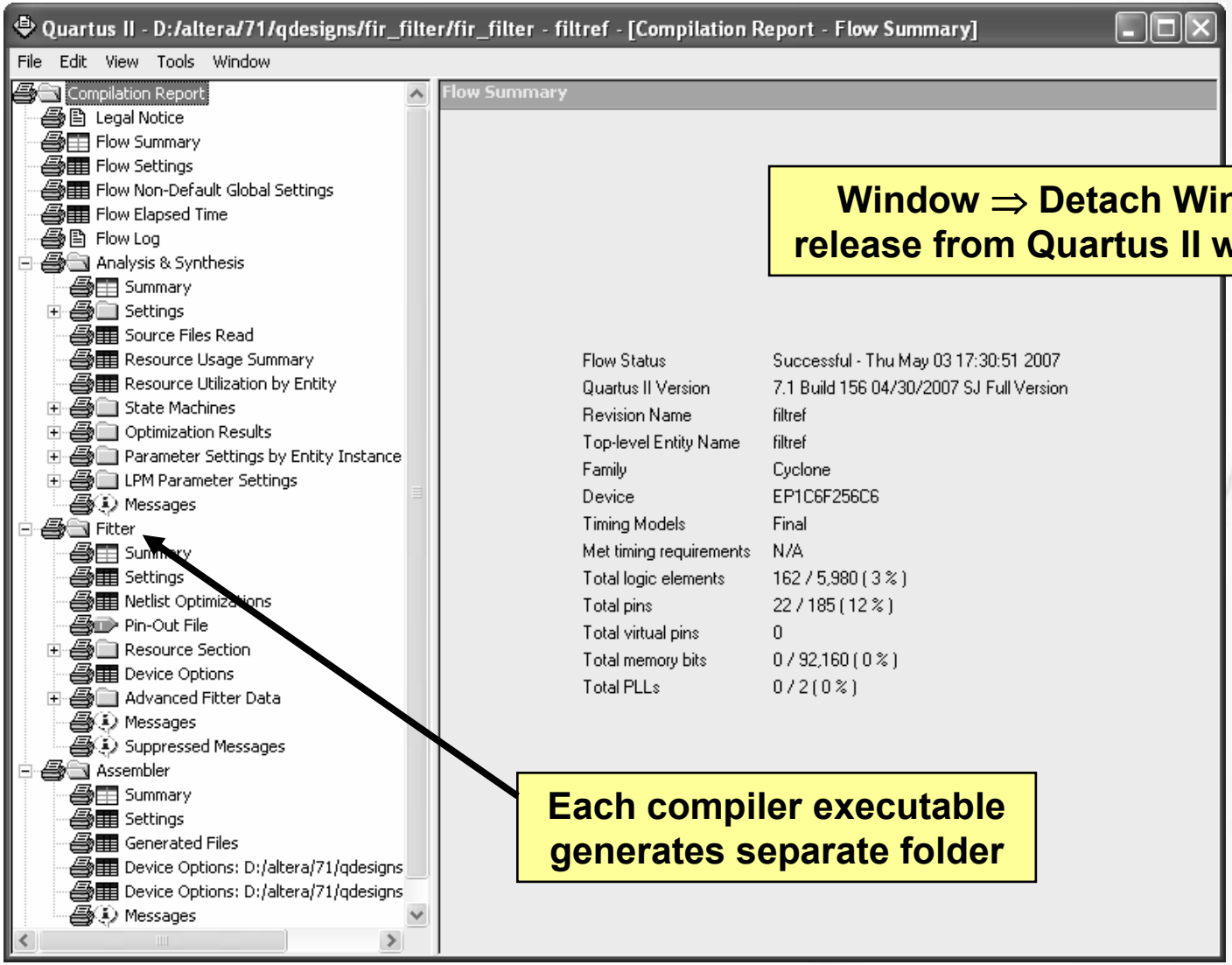
- Use to
  - View all suppressible messages
  - View/add/remove suppression rules
  - View messages suppressed for current & future compiles

Compilation Report

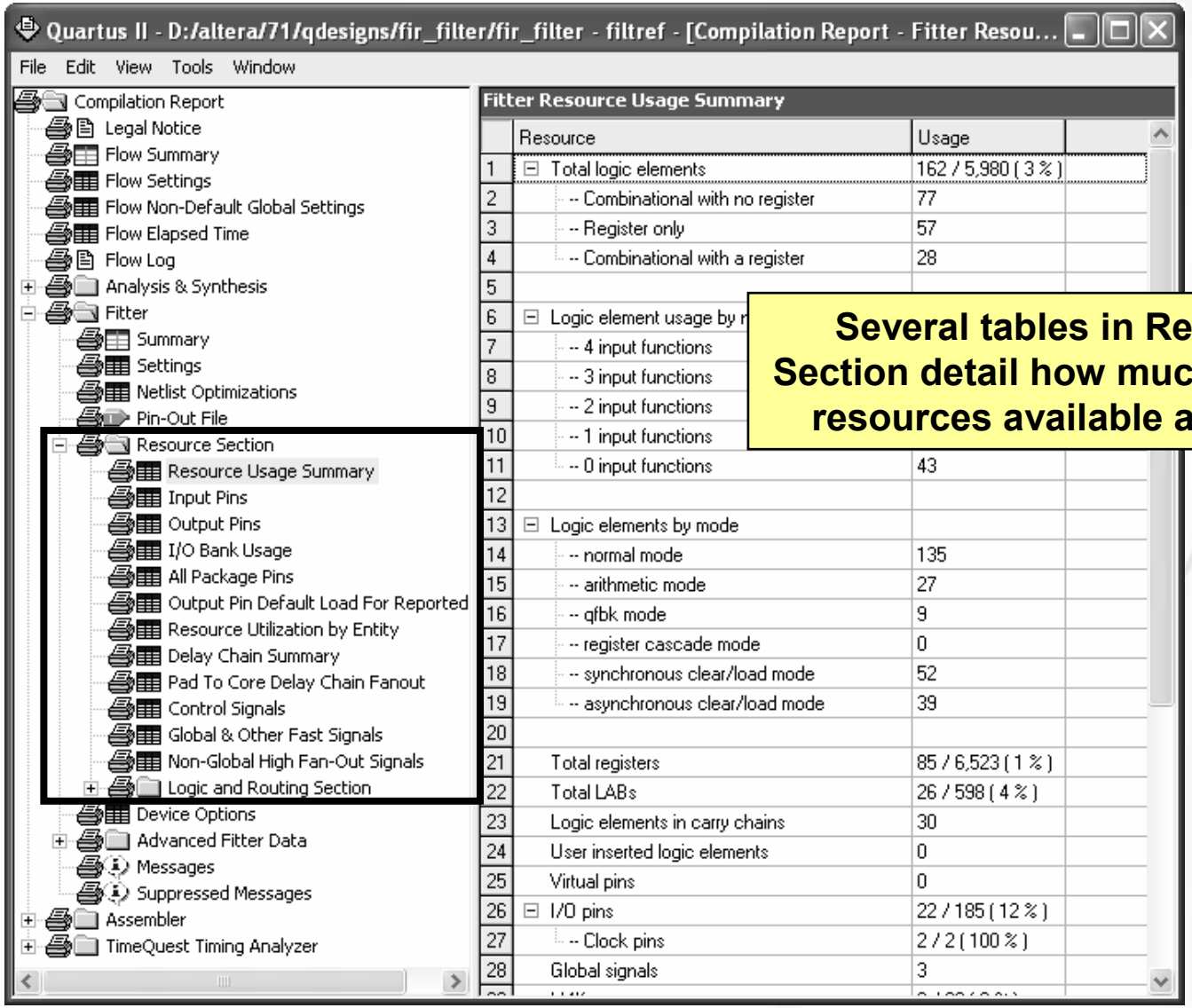
- Graphical window containing all compilation processing information
  - Resource usage
  - Device pin-out
  - Settings and constraints applied
  - Messages
- Opens automatically when processing begins
- *Recommendation:* Go through report for a design to get sense of information being provided

- Information also available as text files in project directory
  - Ex. *<project_name>.fit.rpt* & *<project_name>.map.rpt*

Compilation Report



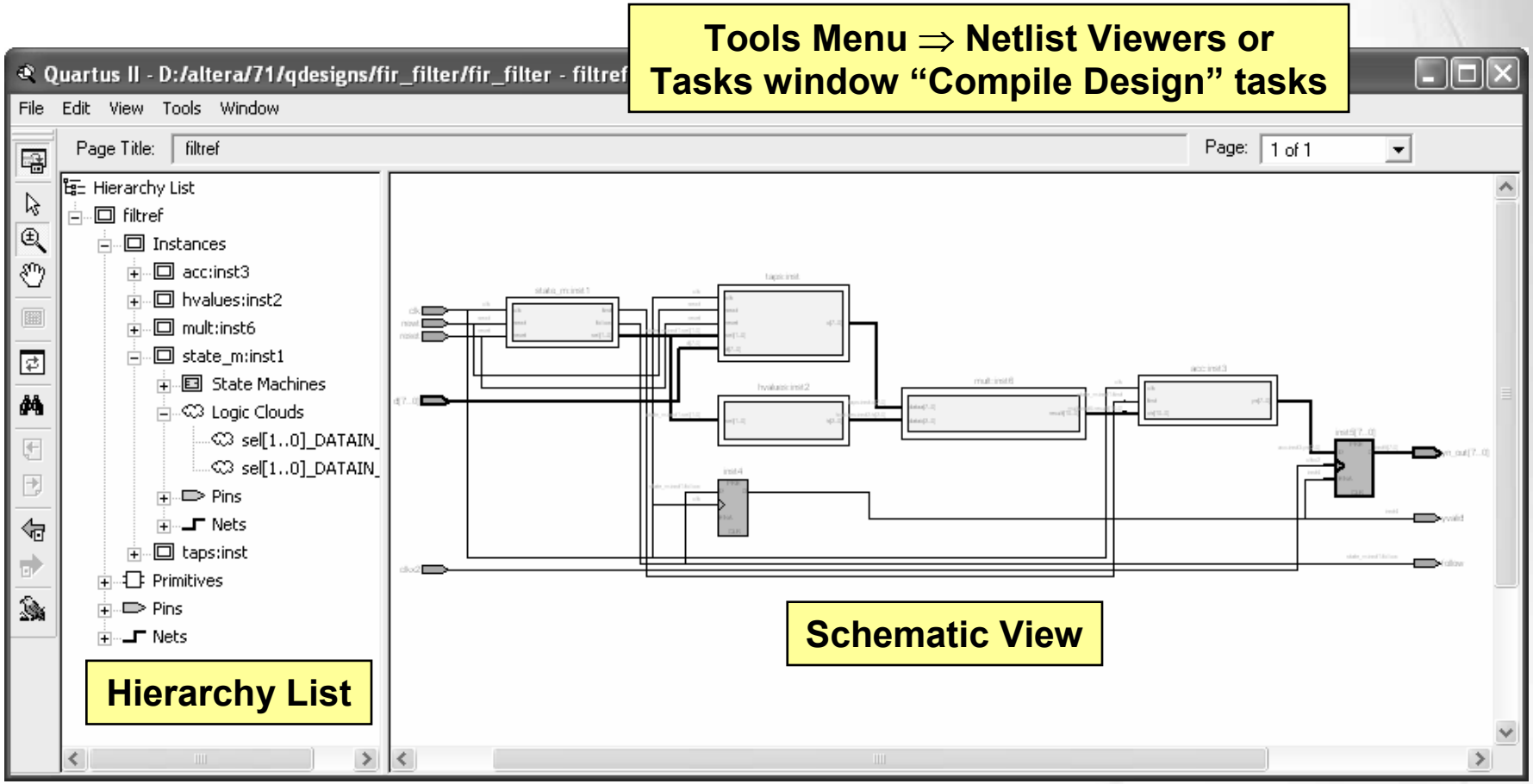
Example: Resource Usage



Netlist Viewers

- RTL Viewer
  - Graphically represents results of synthesis
  - Visually check initial HDL synthesis results
    - Before any Quartus II optimizations
  - Locate synthesized nodes for assigning constraints
  - Debug verification issues
- Technology Map Viewers (Post-Mapping & regular)
  - Graphically represents results of mapping (post-synthesis) & fitting
  - Analyze critical timing paths graphically
    - Delay values displayed if timing information available
  - Locate nodes & node names after optimizations (cross-probing)
    - Assigning constraints
    - Debugging

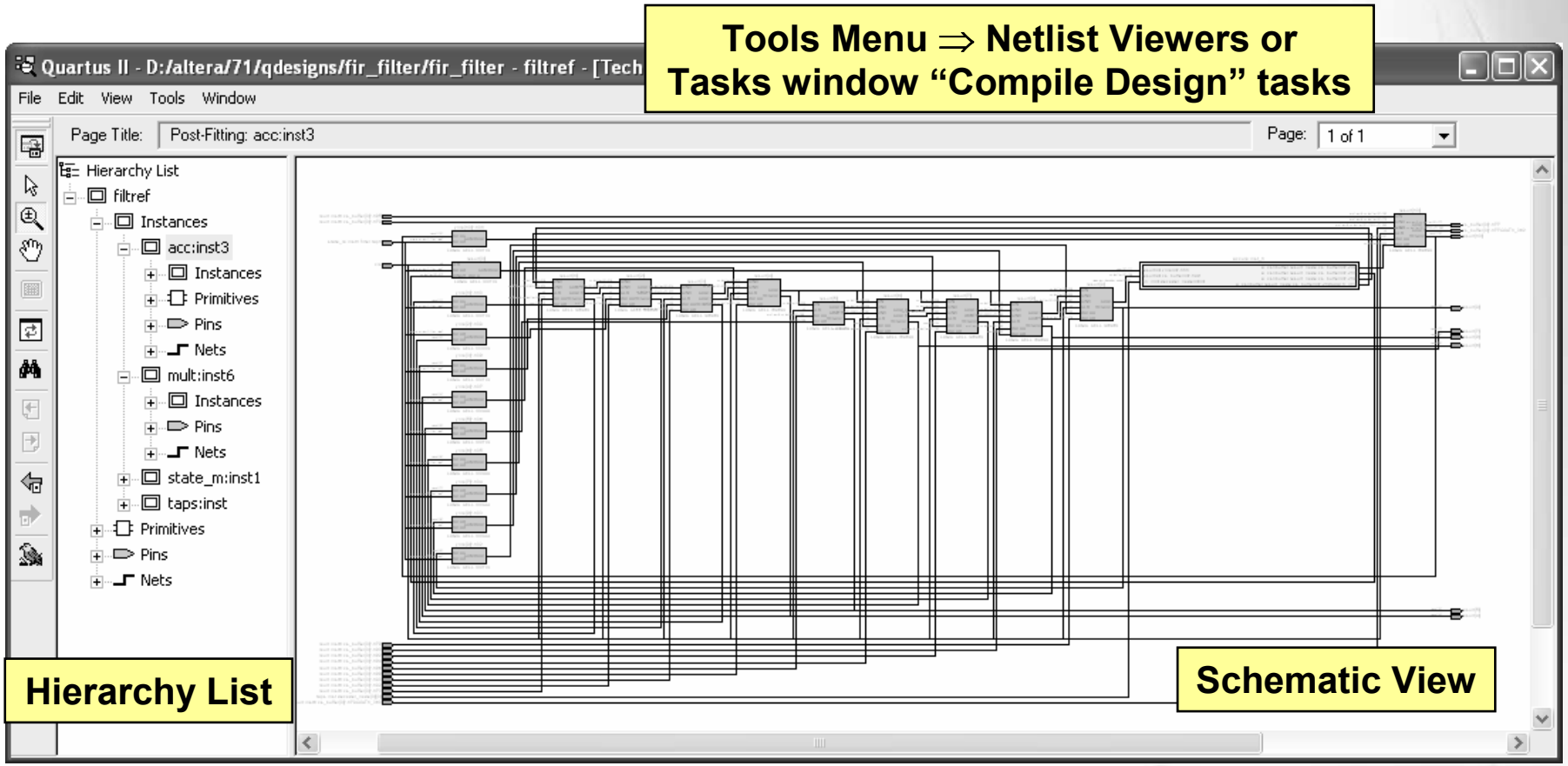
RTL Viewer



Note:

1) Must perform elaboration first (e.g. Analysis & Elaboration OR Analysis & Synthesis)

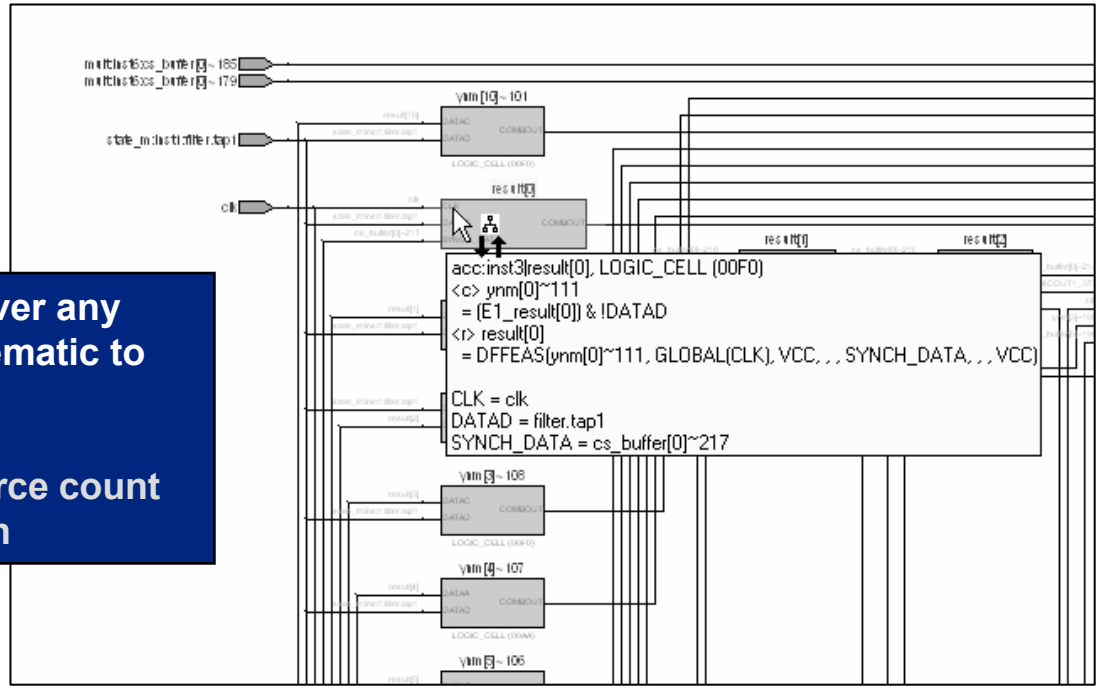
Technology Map Viewers



Note:

1) Must run synthesis and/or fitting first

Schematic View (Technology Viewer)

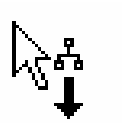


Place pointer over any element in schematic to see details
- **Name**
- **Internal resource count**
- **Logic equation**

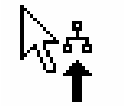
- Represents design using atoms
  - I/O pins & cells
  - Lcells
  - Memory blocks
  - MAC (DSP blocks)

Schematic Hierarchy Navigation

- Mouse pointer indicates action

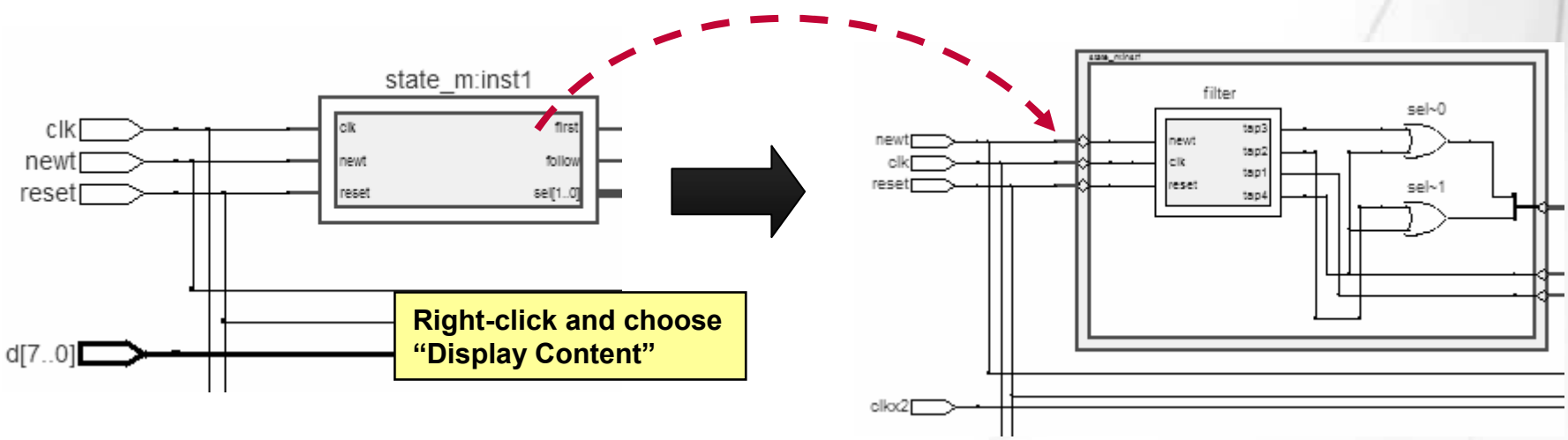


← Descending hierarchy
  - Double-click on instance
  - Right-click & select **hierarchy down**

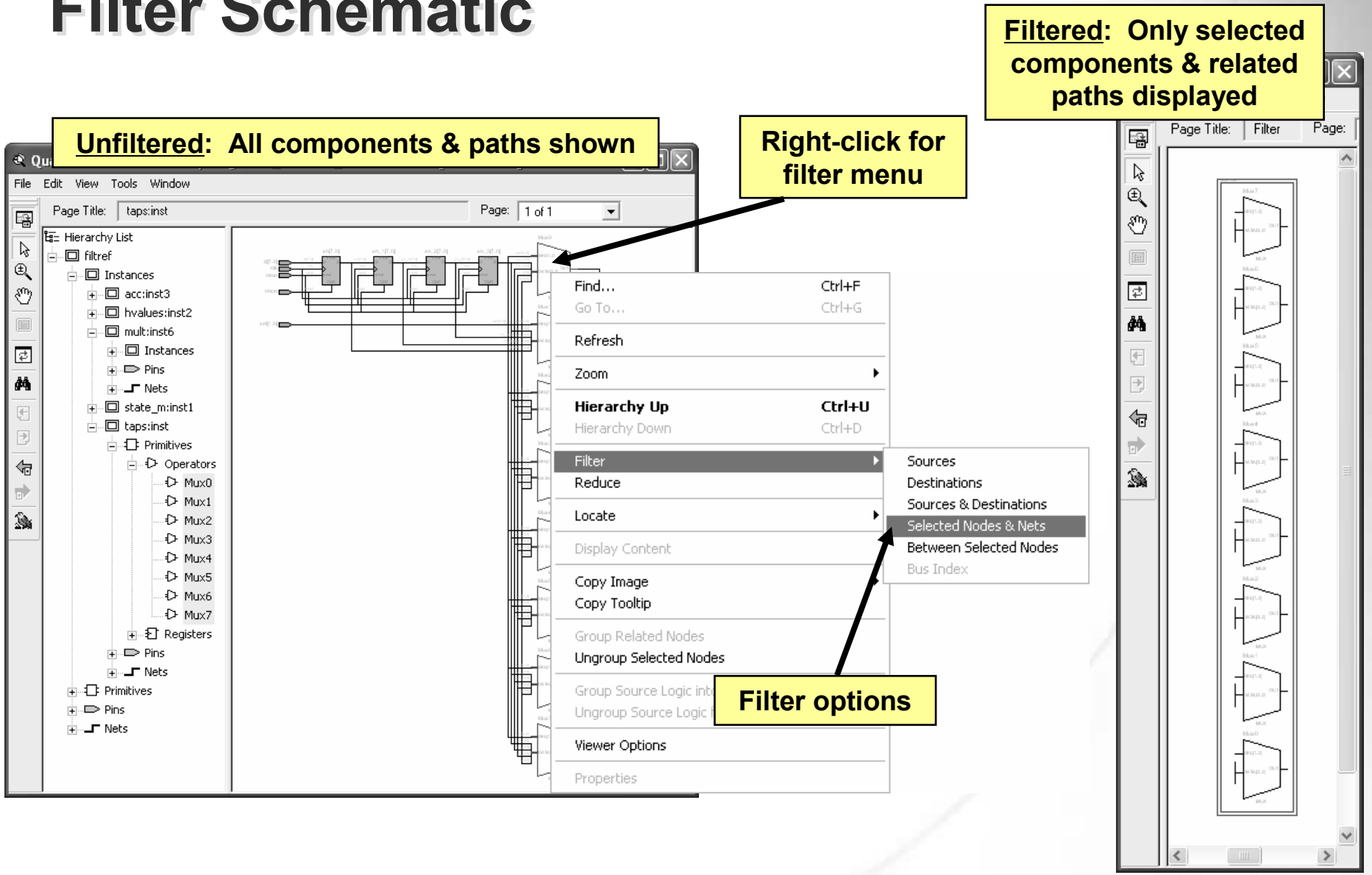


← Ascending hierarchy
  - Double-click in empty space
  - Right-click & select **hierarchy up**

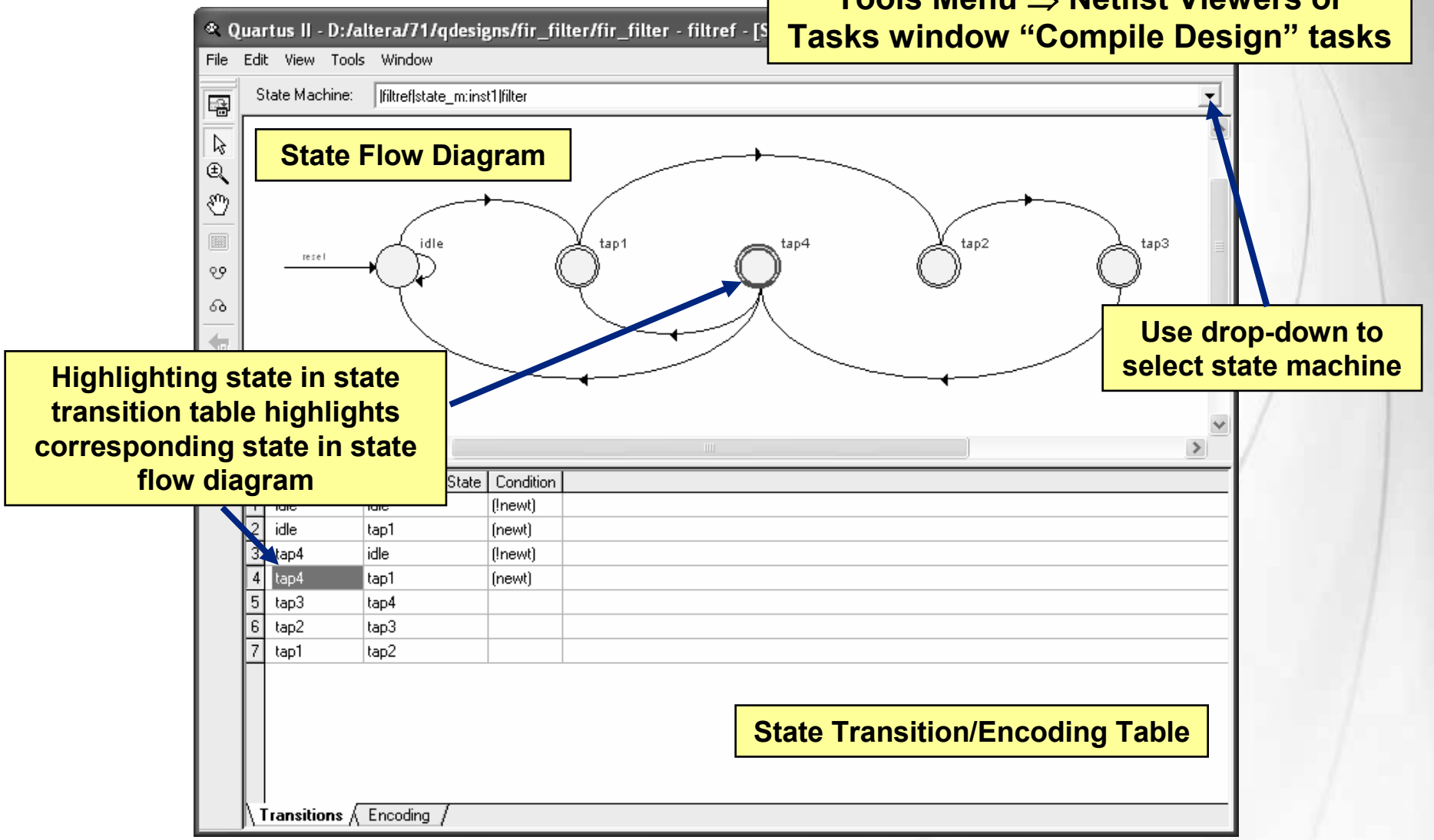
- Expand instances within current level of hierarchy
  - Tip: click instances (blocks) to highlight in red first before performing actions



Filter Schematic



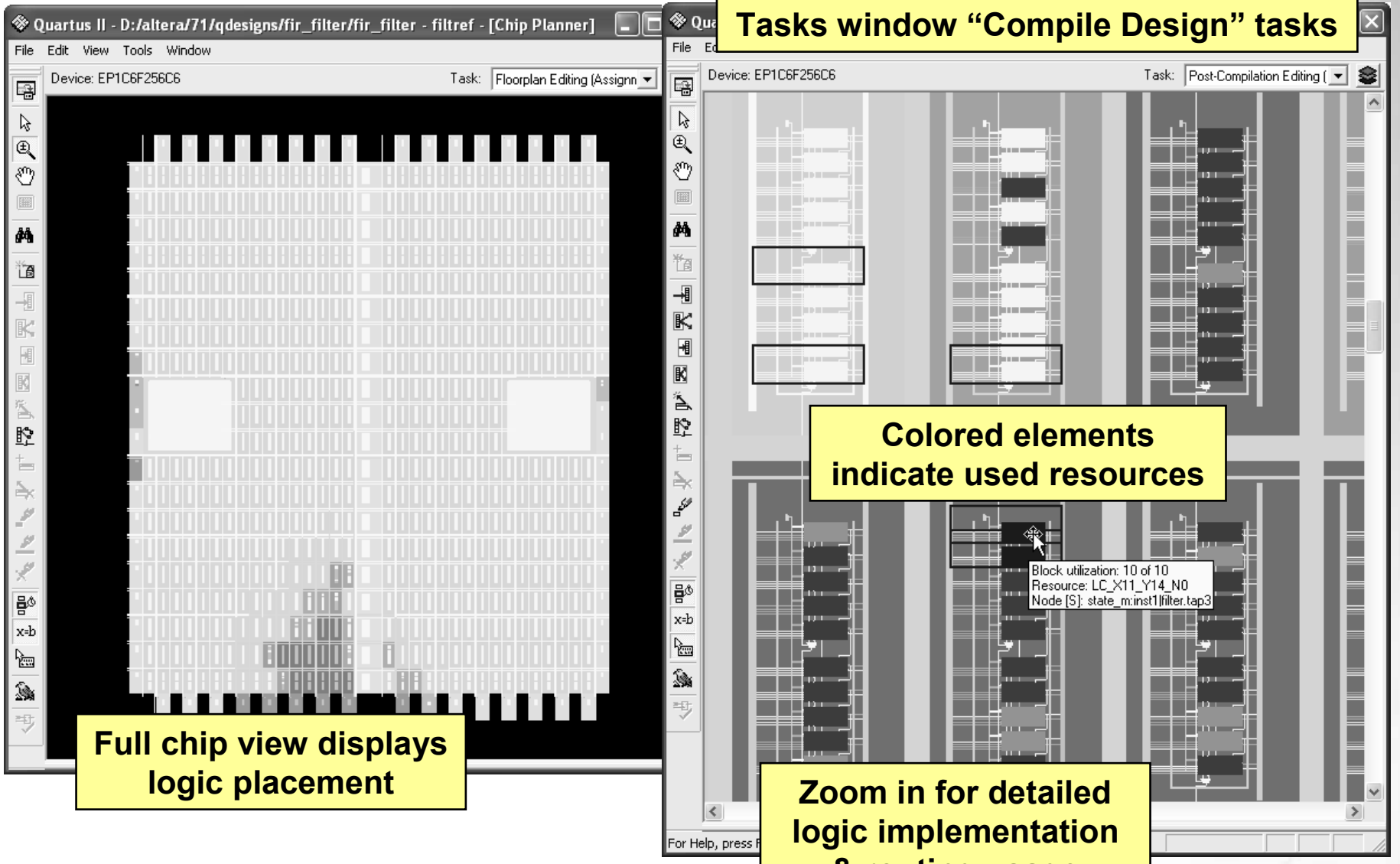
State Machine Viewer



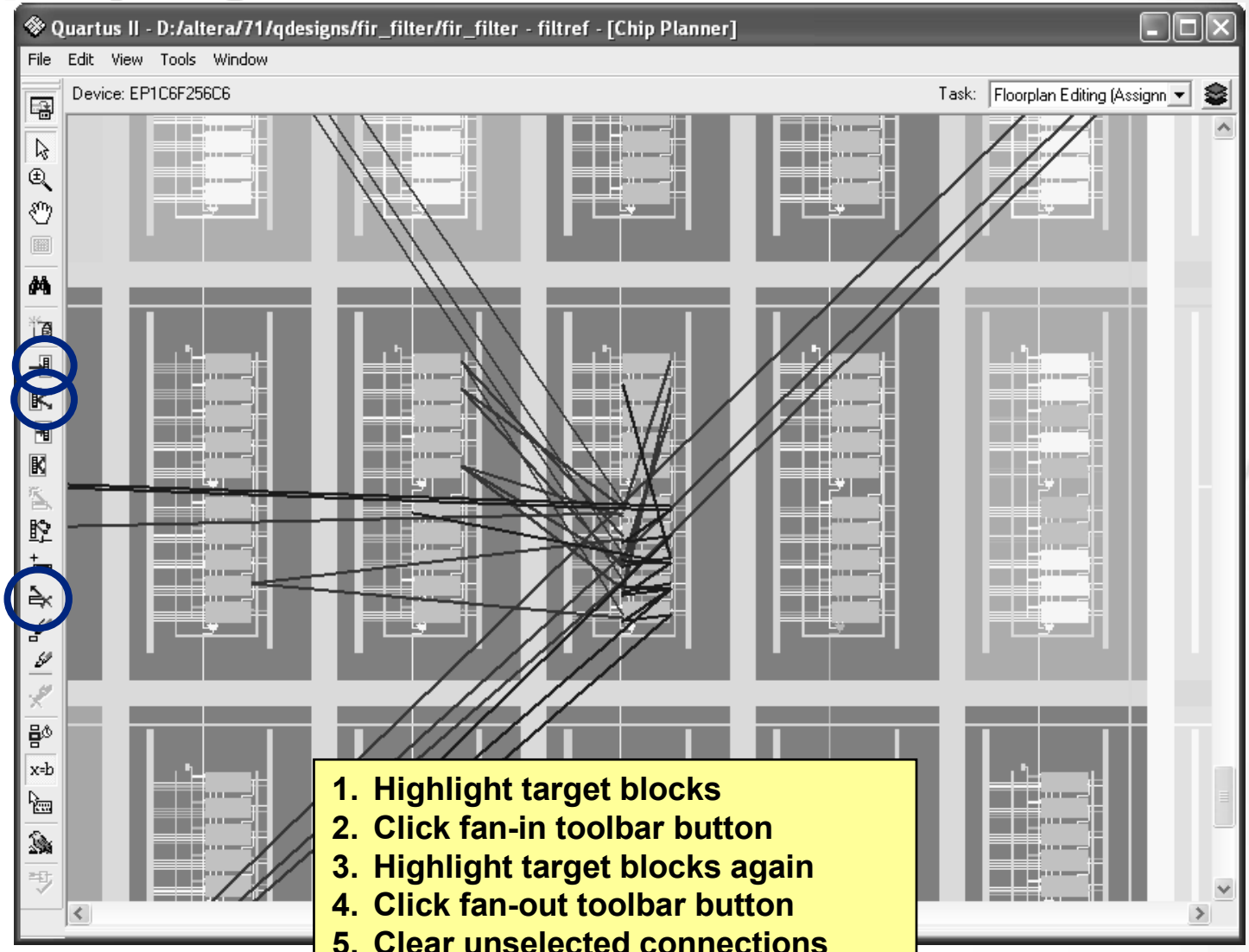
Chip Planner

- Editable graphical view of target device
- Displays
  - Graphical layout of device resources
  - Routing channels between device resources
    - Internal routing channels within LABs
  - Global clock regions
- Uses
  - View placement of design logic
  - View connectivity between resources used in design
  - Make placement assignments
  - Debugging placement related issues

Chip Planner



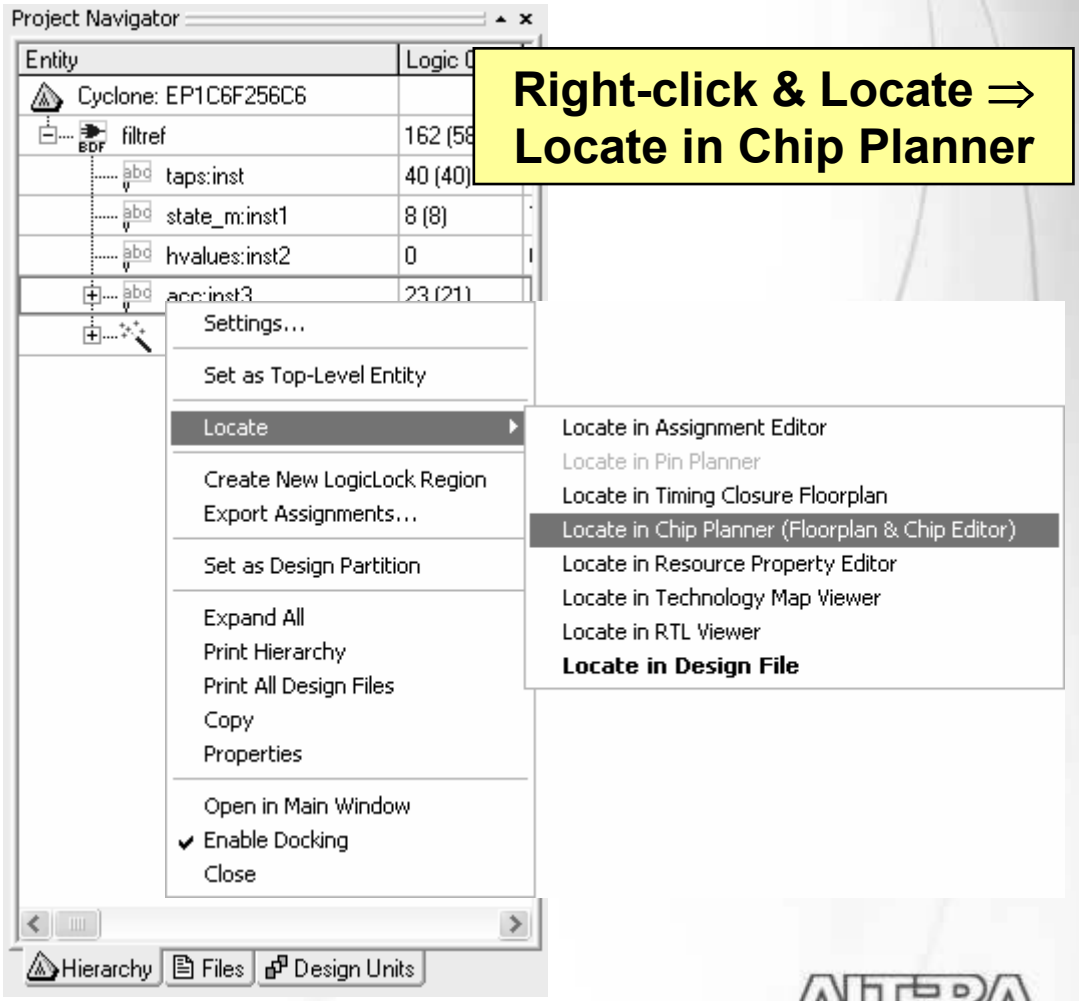
Displaying Fan-In & Fan-Out



Cross-Probing from/to Chip Planner

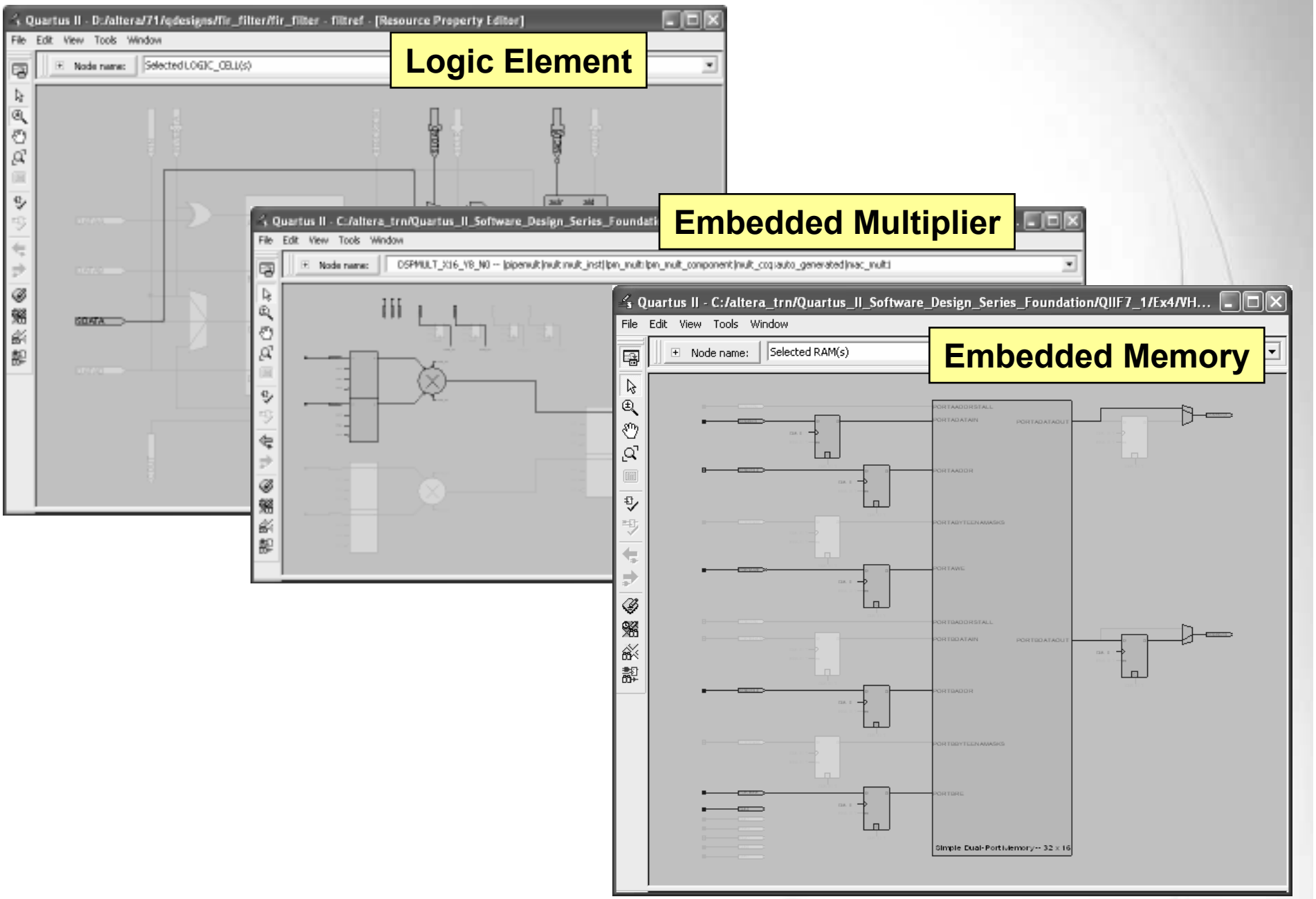
- Locate hierarchy blocks or specific logic from other Quartus II windows

- Project Navigator
- Compilation Report
- Design files
- RTL Viewer
- Technology Viewer
- Message window
- Pin Planner
- TimeQuest reports



Resource Property Editor

- Use to view detailed logic implementation & connections
  - No direct access from menus
  - Cross-probe from other Quartus II tools
- Advanced feature: make ECO changes post-fit without recompiling
- Views
  - Logic cells (look-up tables & registers)
  - Embedded memory
  - Embedded multipliers
  - I/O cells
  - PLLs

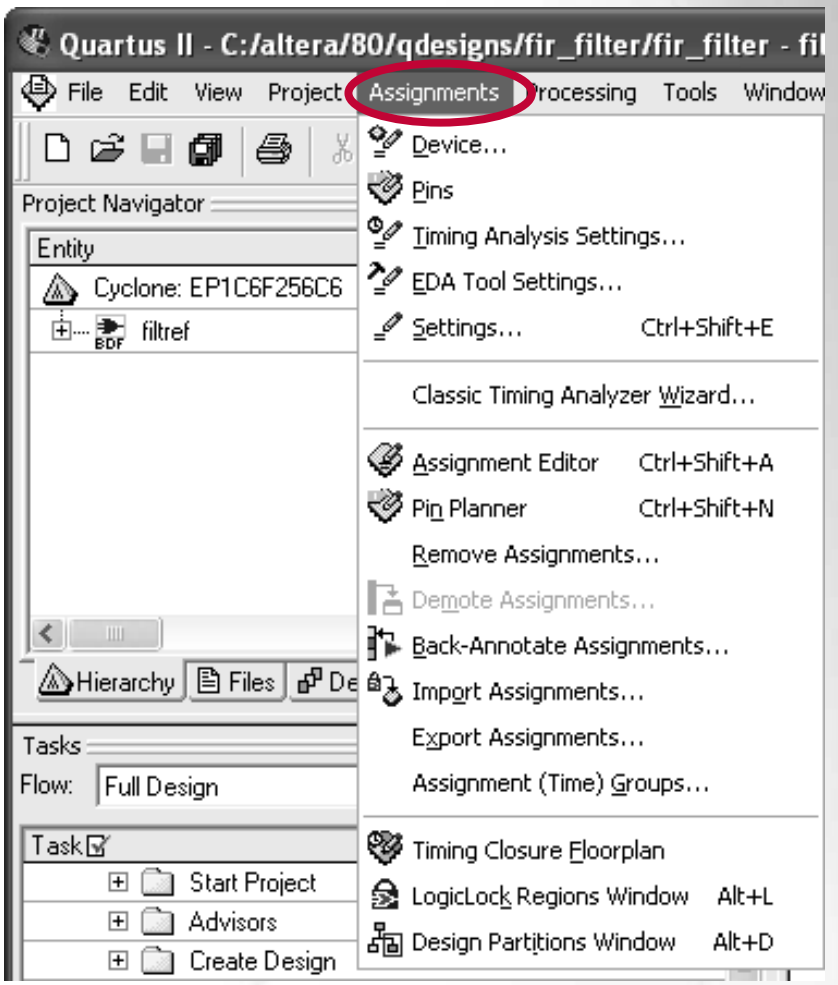


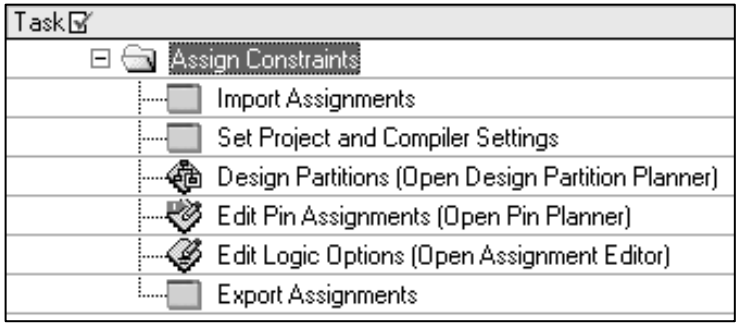
Quartus II Software Design Series: Foundation

Settings & Assignments

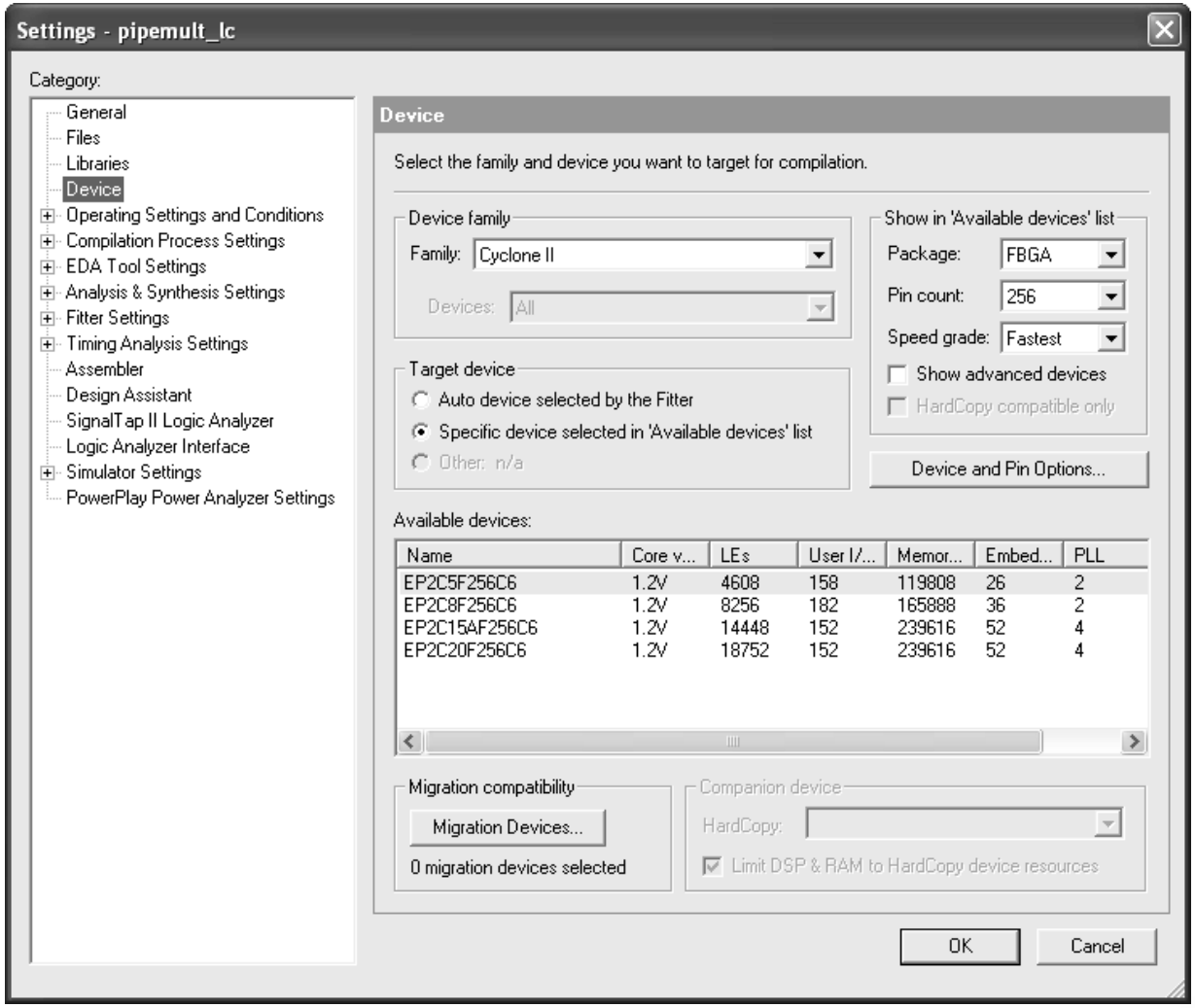
Synthesis & Fitting Control

- Controlled using two methods
  - **Settings**
    - Project-wide switches
  - **Assignments** (aka logic options; constraints)
    - Individual entity/node controls
- Both accessed in **Assignments** menu or Tasks window
- Stored in QSF file for project/revision





Settings Dialog Box

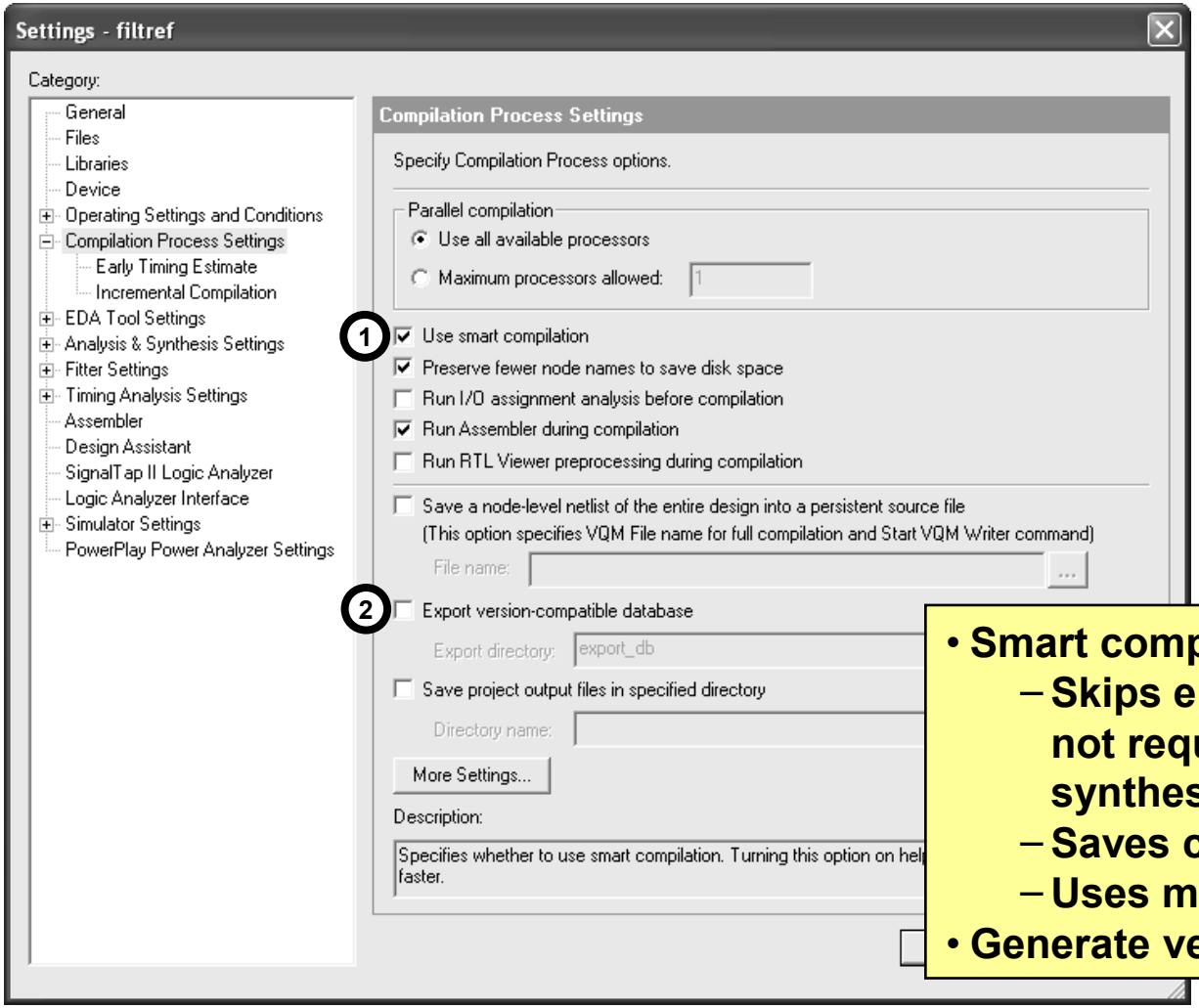


Change settings

- **Top-level entity**
- **Target device**
- **Add/remove files**
- **Libraries**
- **VHDL ’87 or ‘93?**
- **Verilog ‘95, ’01 or SystemVerilog?**
- **EDA tool settings**
- **Timing settings**
- **Compiler settings**
- **Synthesis settings**
- **Fitter settings**
- **Simulator settings**
- **Power analysis settings**

Tcl: set_global_assignment –name <assignment_name*> <value>

Compilation Process

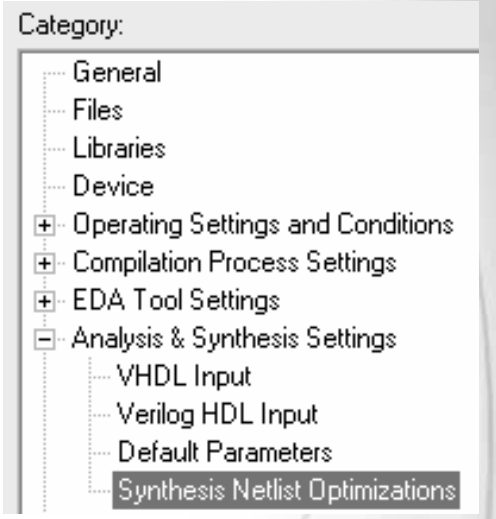


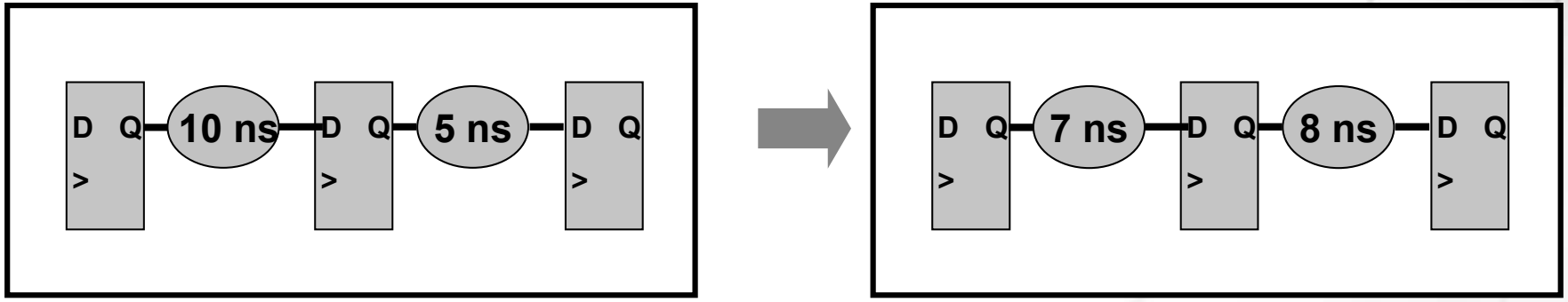
- **Smart compilation(1)**
  - **Skips entire compiler modules when not required (i.e. elaboration, synthesis, etc.)**
  - **Saves compiler time**
  - **Uses more disk space**
- **Generate version-compatible database(2)**

Tcl: set_global_assignment -name SMART_RECOMPILE ON

Synthesis Netlist Optimizations: Gate-Level Register Retiming

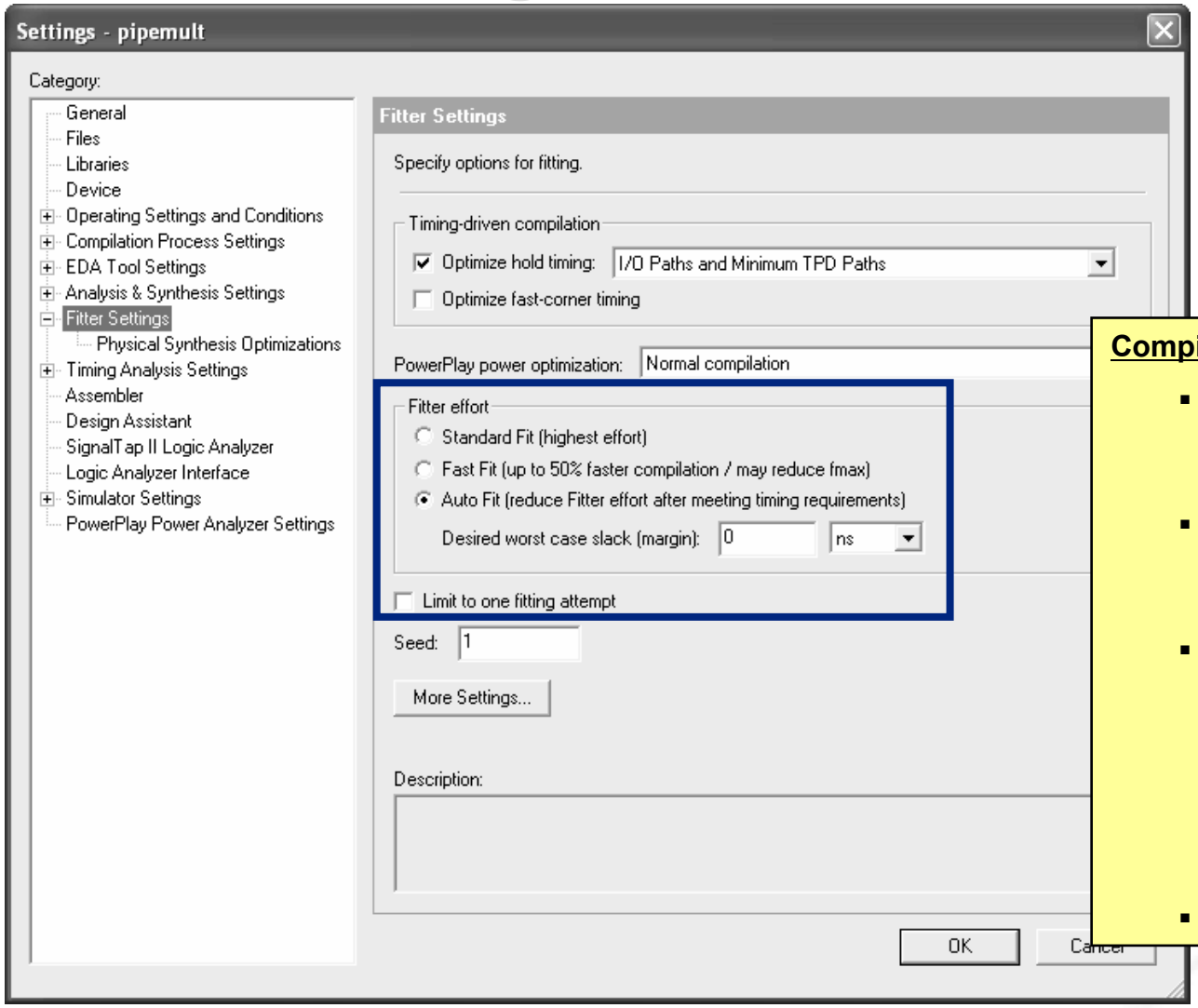
- Optimize netlist during Synthesis
- Moves registers across combinatorial logic to balance timing
- Trades between critical & non-critical paths
- Makes changes at gate level
- Created/modified nodes noted in Compilation Report





Tcl: set_global_assignment –name ADV_NETLIST_OPT_SYNTH_GATE_RETIME ON

Fitter Settings



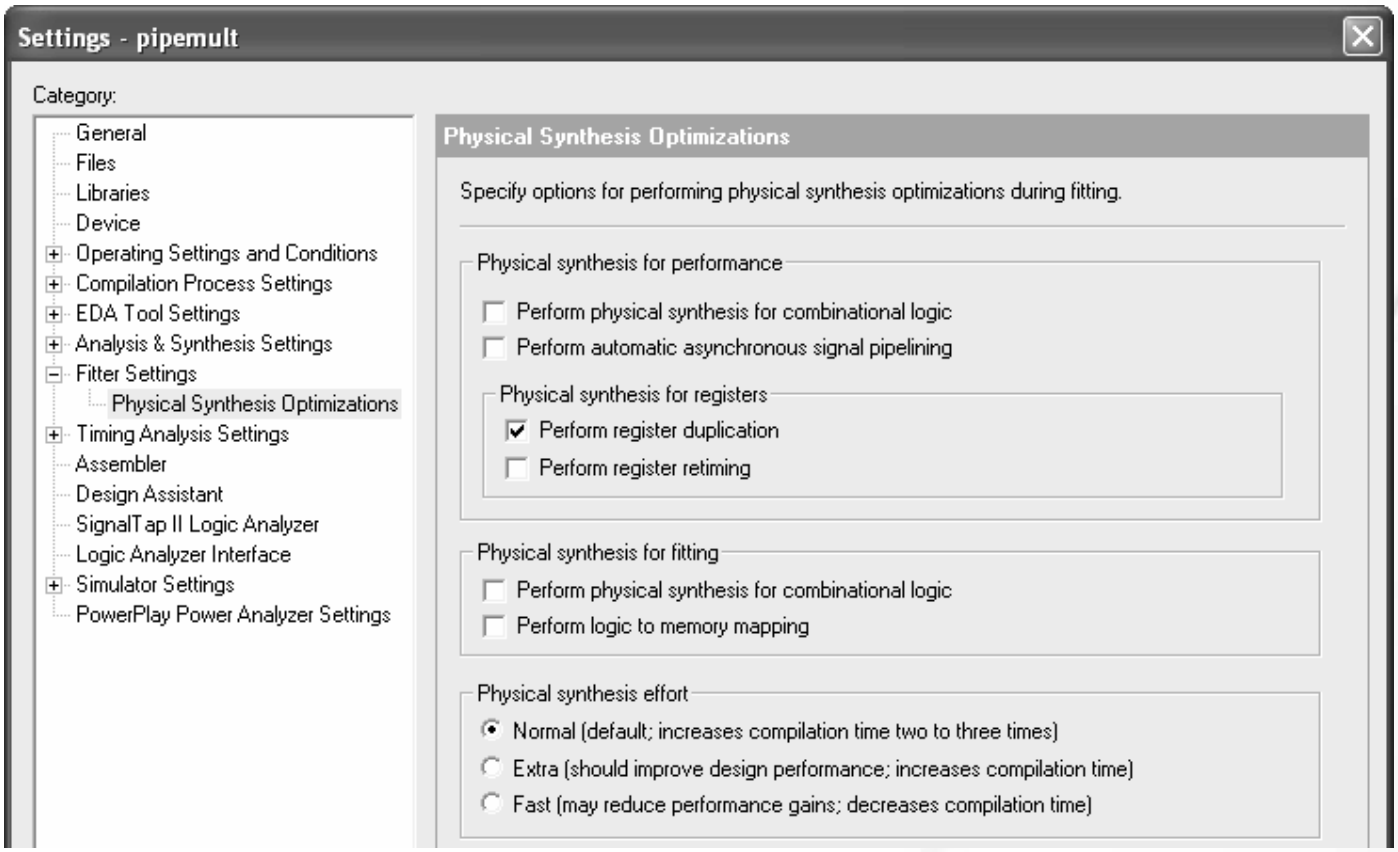
Compilation speed/fitter effort

- **Standard fit**
  - **Highest effort**
  - **Longest compile time**
- **Fast fit**
  - **Faster compile but possibly lesser design performance**
- **Auto fit**
  - **Compile stops after meeting timing**
  - **Conserves CPU time**
  - **Will mimic standard fit for hard-to-fit designs**
  - **Default for new designs**
- **One fitting attempt**

Tcl: set_global_assignment –name FITTER_EFFORT "<Effort Level>"

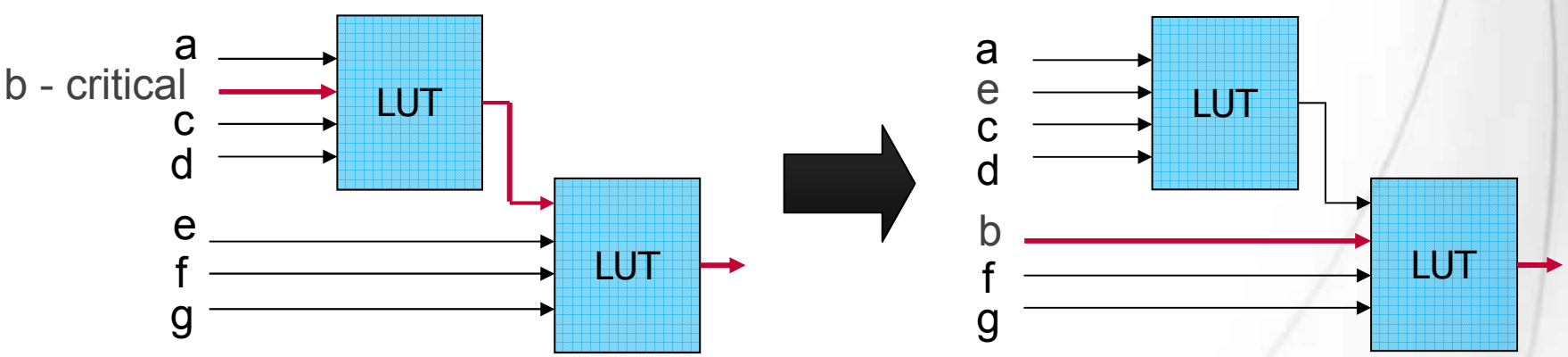
Physical Synthesis

- Re-synthesize based on fitter output
  - Makes incremental changes that improve results for a given placement
  - Compensates for routing delays from Fitter



Combinational Logic

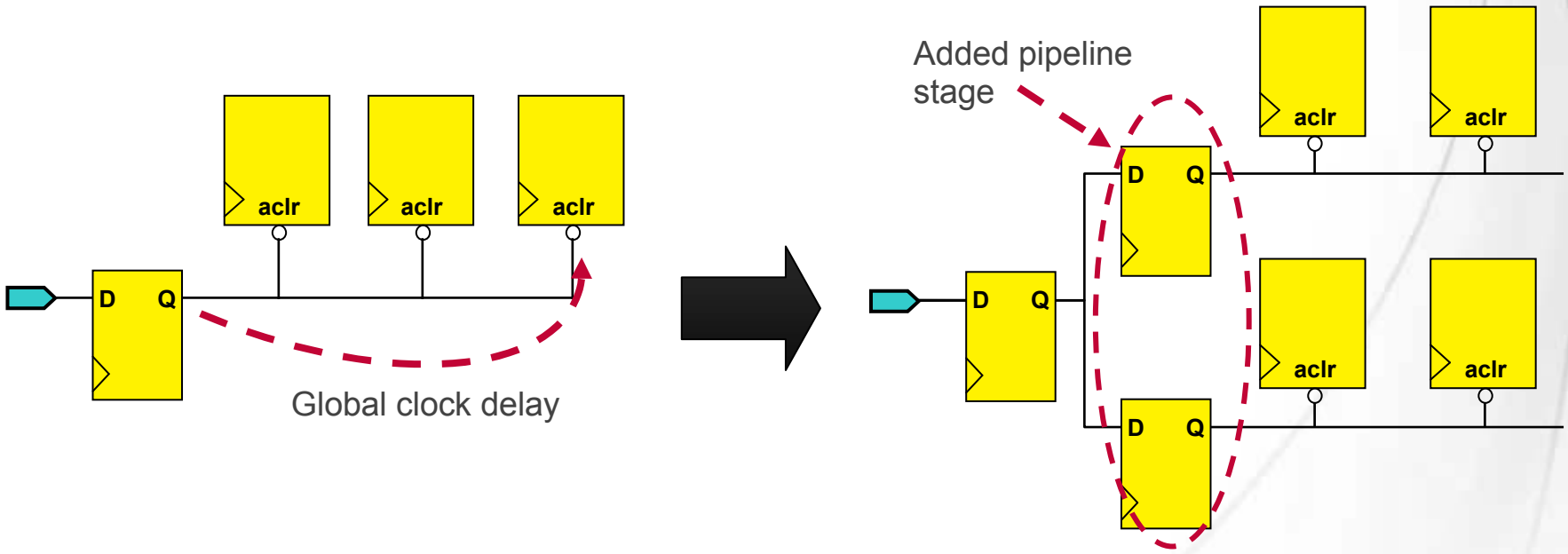
- Swaps look-up table (LUT) ports within LEs to reduce critical path LEs



Tcl: set_global_assignment –name PHYSICAL_SYNTHESIS_COMBO_LOGIC ON

Asynchronous Signal Pipelining

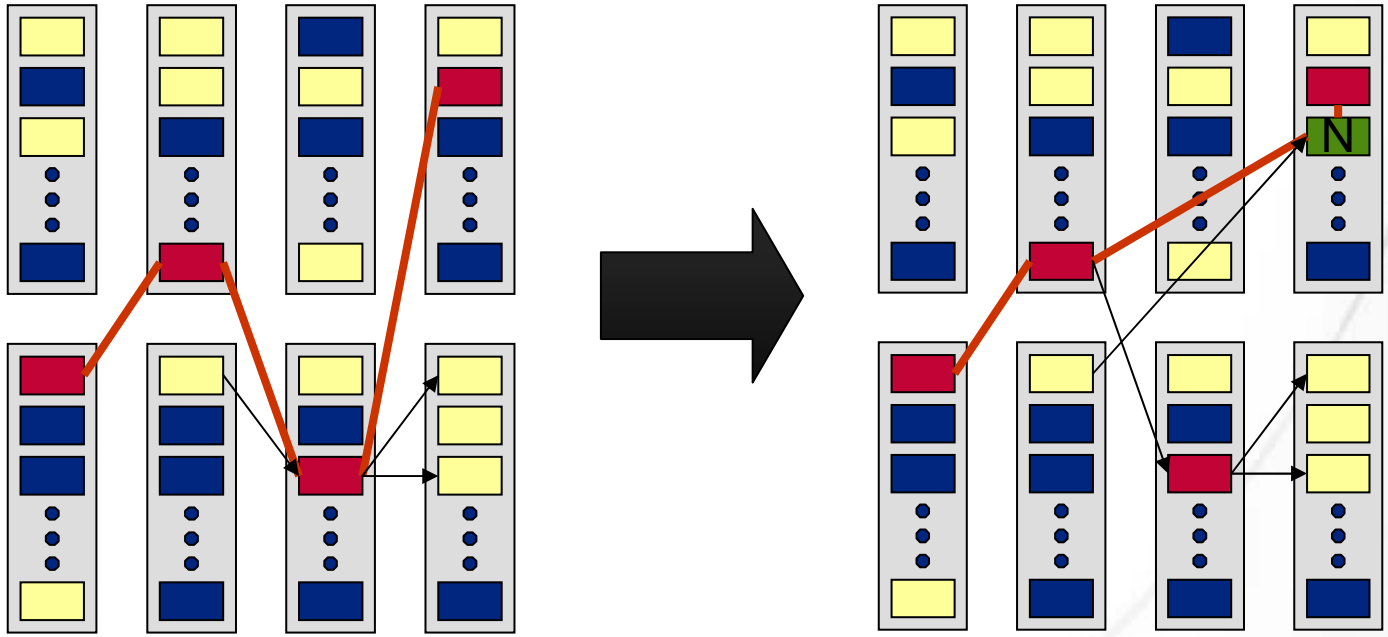
- Adds pipeline registers to asynchronous clear or load signals in very fast clock domains



Tcl: set_global_assignment –name PHYSICAL_SYNTHESIS_ASYNCHRONOUS_SIGNAL_PIPELINING ON

Duplication

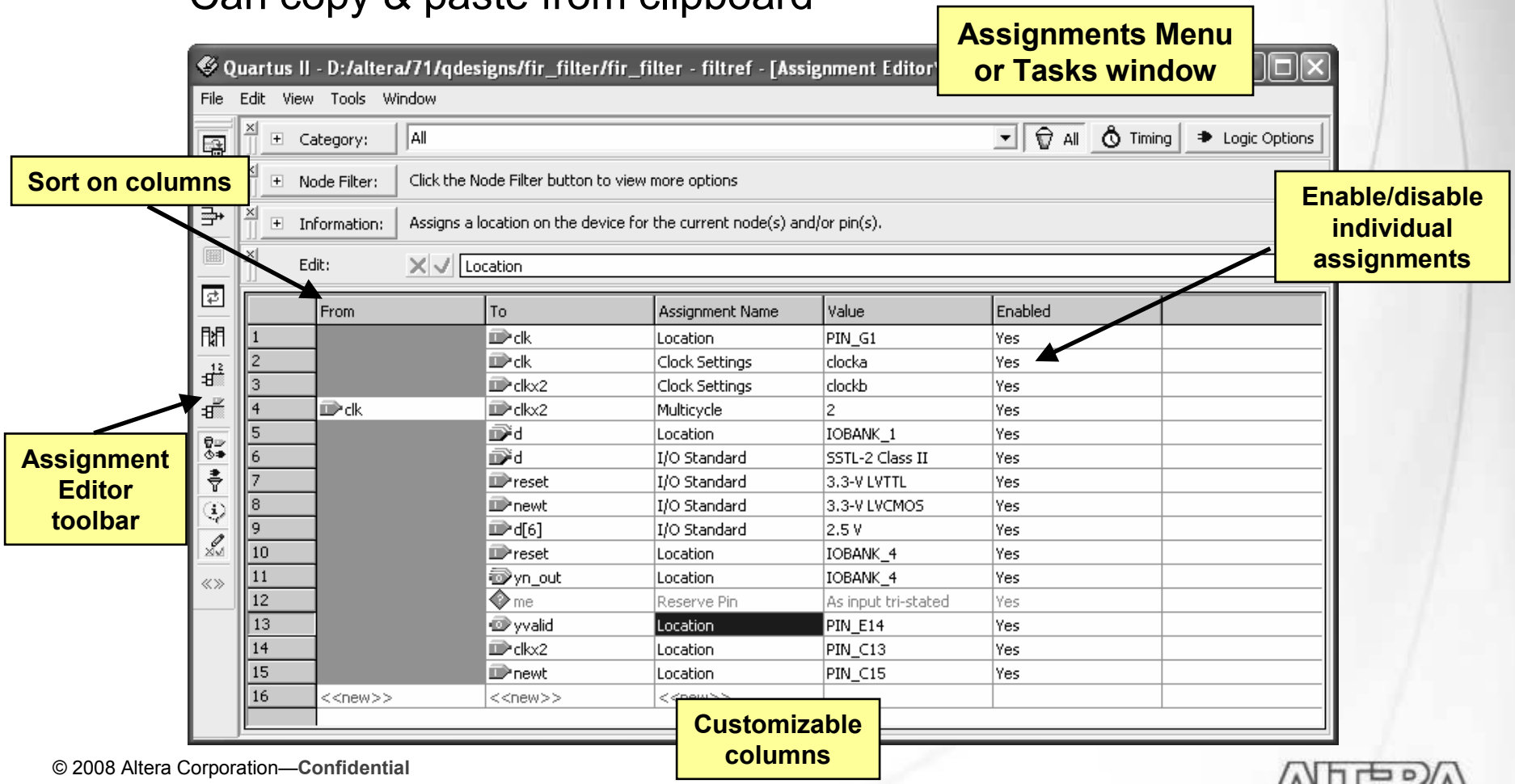
- High fan-out registers or combinatorial logic duplicated & placed to reduce delay



Tcl: set_global_assignment –name PHYSICAL_SYNTHESIS_REGISTER_DUPLICATION ON

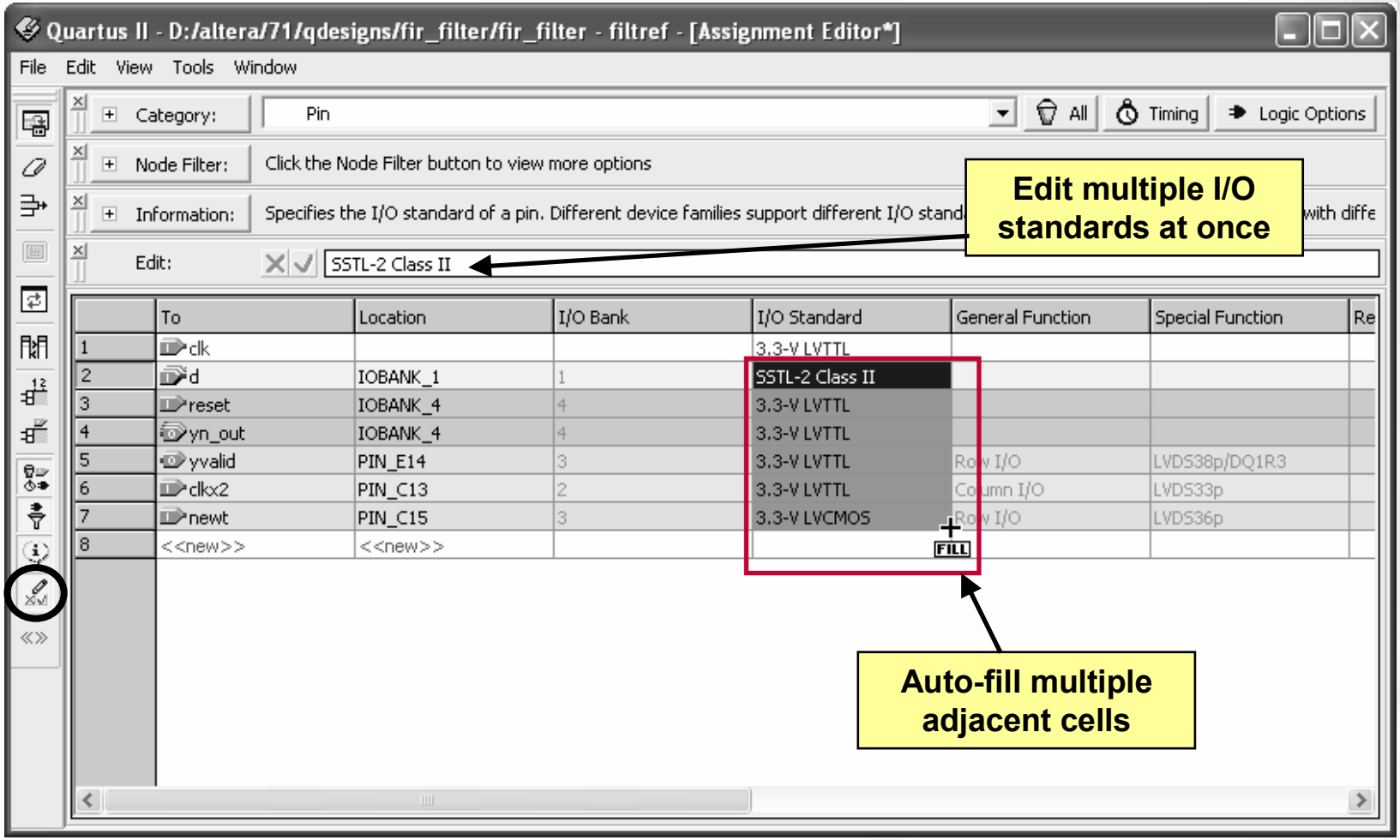
Assignment Editor (AE)

- Provides spreadsheet assignment entry & display
  - Can copy & paste from clipboard



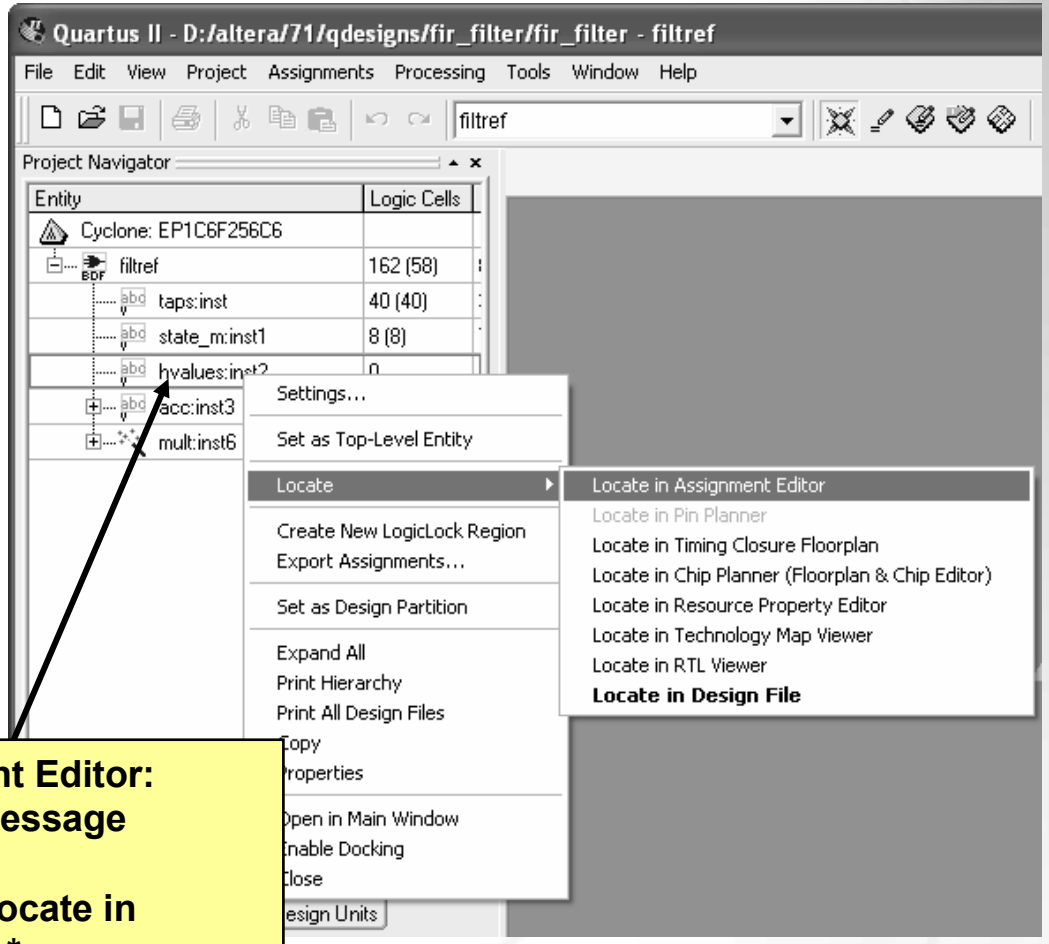
Editing Multiple Assignments

- Use Edit bar, auto-fill, copy & paste



Creating Assignments: Cross-Probing

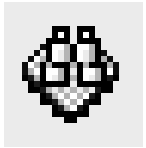
- Virtually all windows & tools cross-probe (locate) to Assignment Editor
- Examples
  - Project Navigator
  - Message window
  - Compilation Report
  - Design files



To invoke Assignment Editor:
1) Highlight object/message
2) Right-click
3) Select Locate ⇒ Locate in Assignment Editor*

****Note:* Assignment Editor pre-filled with target node/pin name***

Node Finder

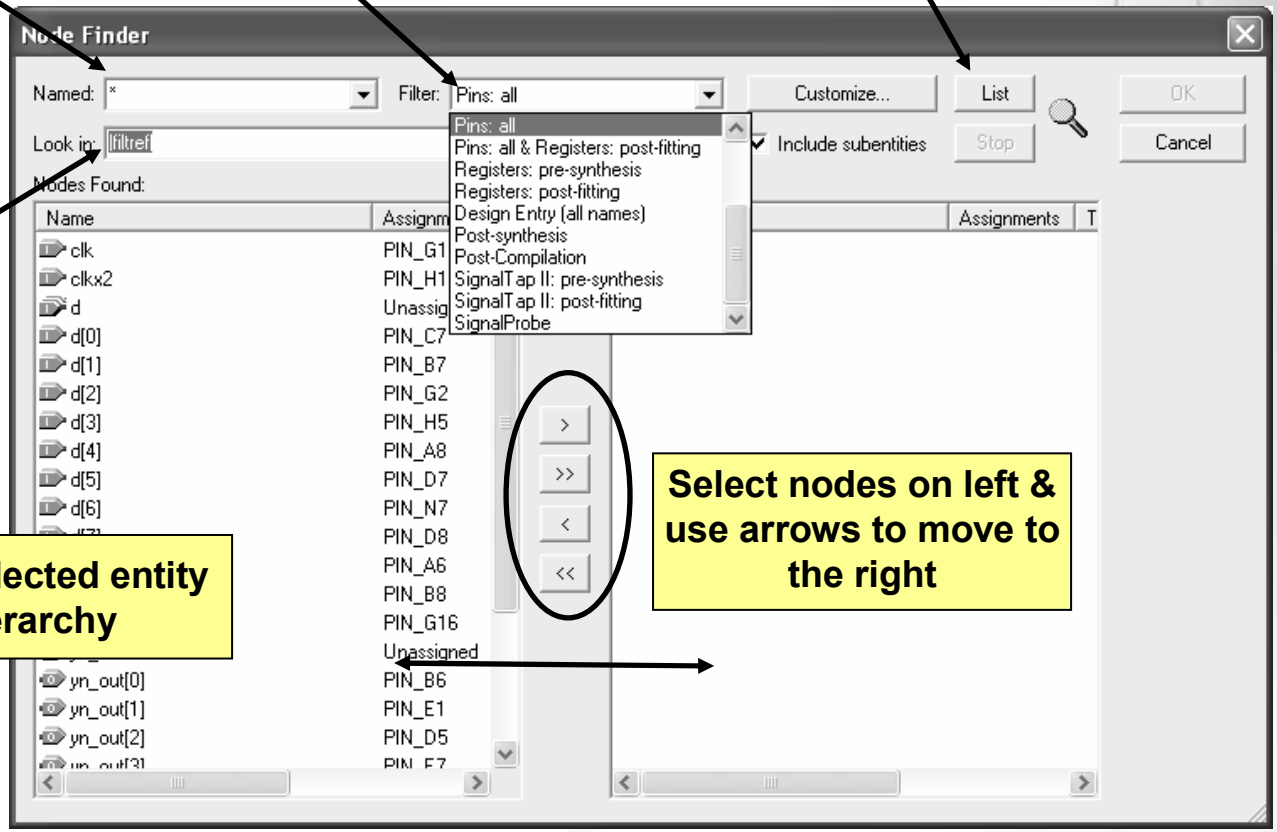


Search by name using wildcards (? or *)

Use filter to select the type of nodes to be displayed

Displays nodes meeting search criteria

Locate nodes in a certain level of hierarchy

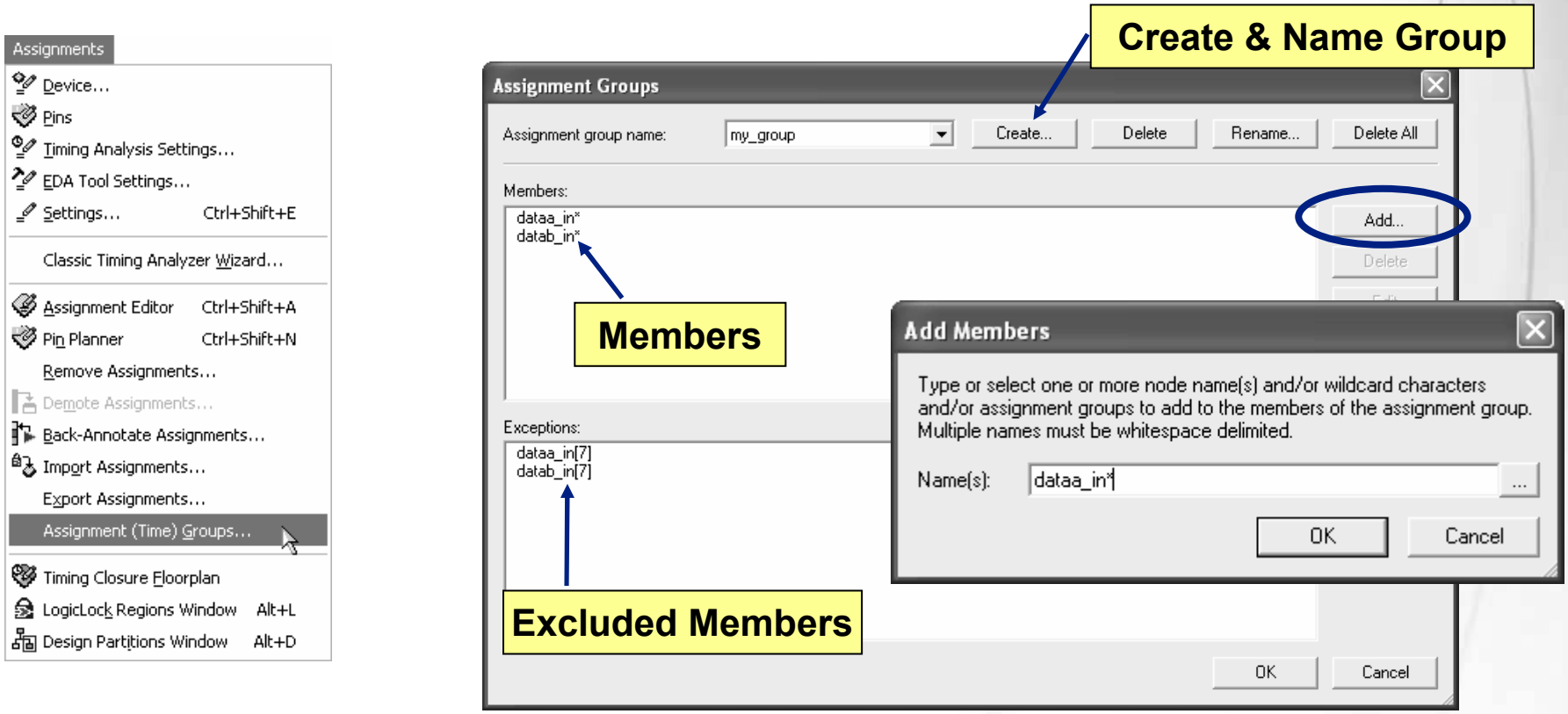


List of found nodes in selected entity & lower levels of hierarchy

Select nodes on left & use arrows to move to the right

Assignment (Time) Groups

- Assign names to user-defined groups of nodes
- Allows single assignment to constrain entire group



AE Dynamic Checking

- Validity of constraint checked during entry
- Color-coded to display status
  - Grey – disabled
  - Black – applied
  - Yellow – assignment warning
  - 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>>		

Optimization Technique

- Selects synthesis optimization goal
  - **Speed**
  - **Balanced** (default)
  - **Area**
- Applies only to hierarchical entities
  - Locate (cross-probe) from Project Navigator
    - Enables Assignment Editor Node Filter for selected entity
  - Drag and drop into Assignment Editor
- Effects synthesis & logic mapping
- Only applies to Quartus II integrated synthesis

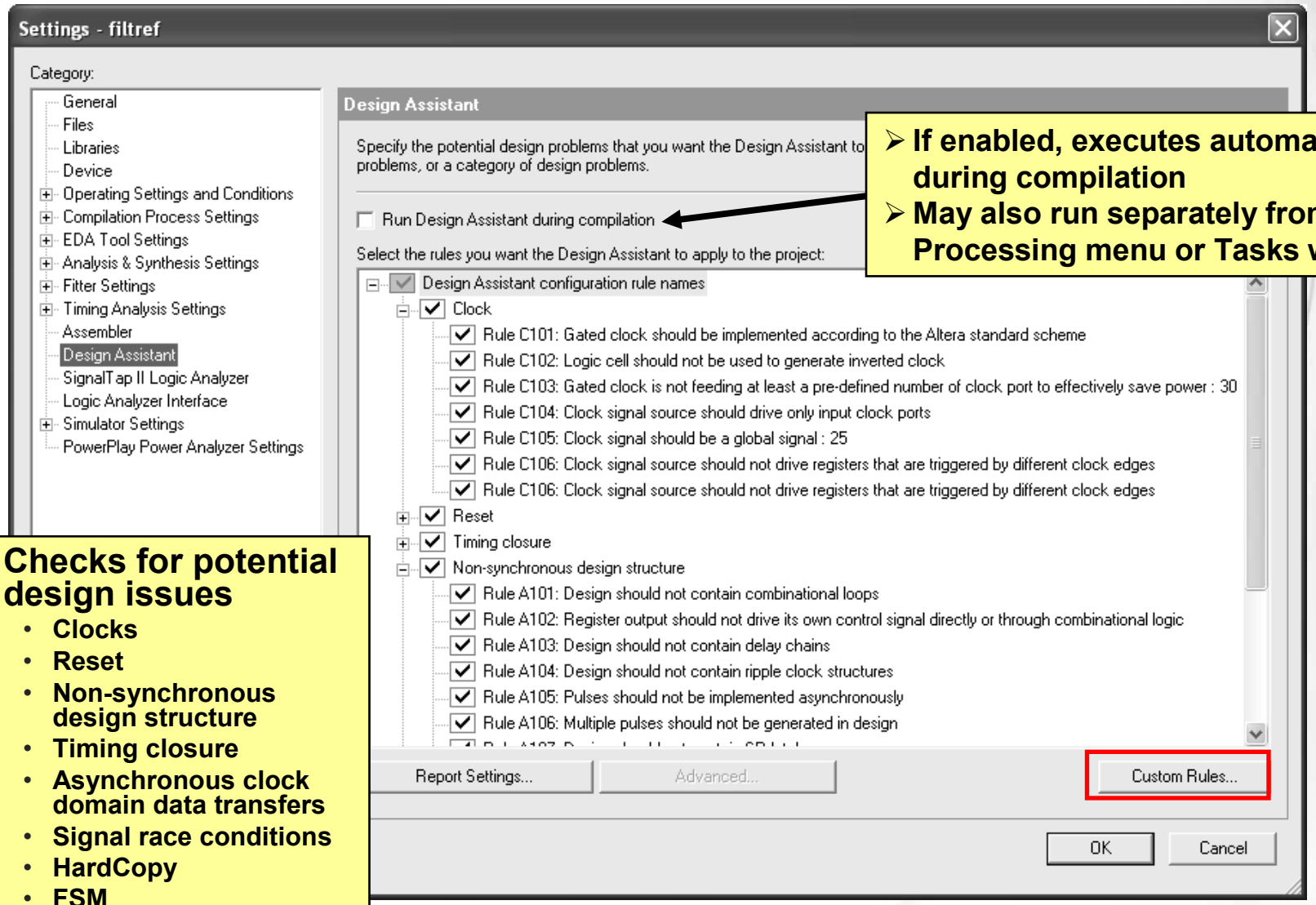
	From	To	Assignment Name	Value	Enabled	
1		acc:b2v_inst3	Optimization Technique -- Stratix II	Speed	Yes	
2	<<new>>	<<new>>	<<new>>			

Tcl: set_instance_assignment –name STRATIXII_OPTIMIZATION_TECHNIQUE SPEED –to <node name>

Updating QSF File

- QSF not updated automatically when constraint entered or Assignment Editor saved
- QSF updated only when
  - Project is saved (**File** menu)
  - Beginning of compilation
- Change behavior to updating assignments immediately (**Tools** menu ⇒ **Options** ⇒ **General** ⇒ **Processing**)
  - May impact software performance due to file accesses

Design Assistant

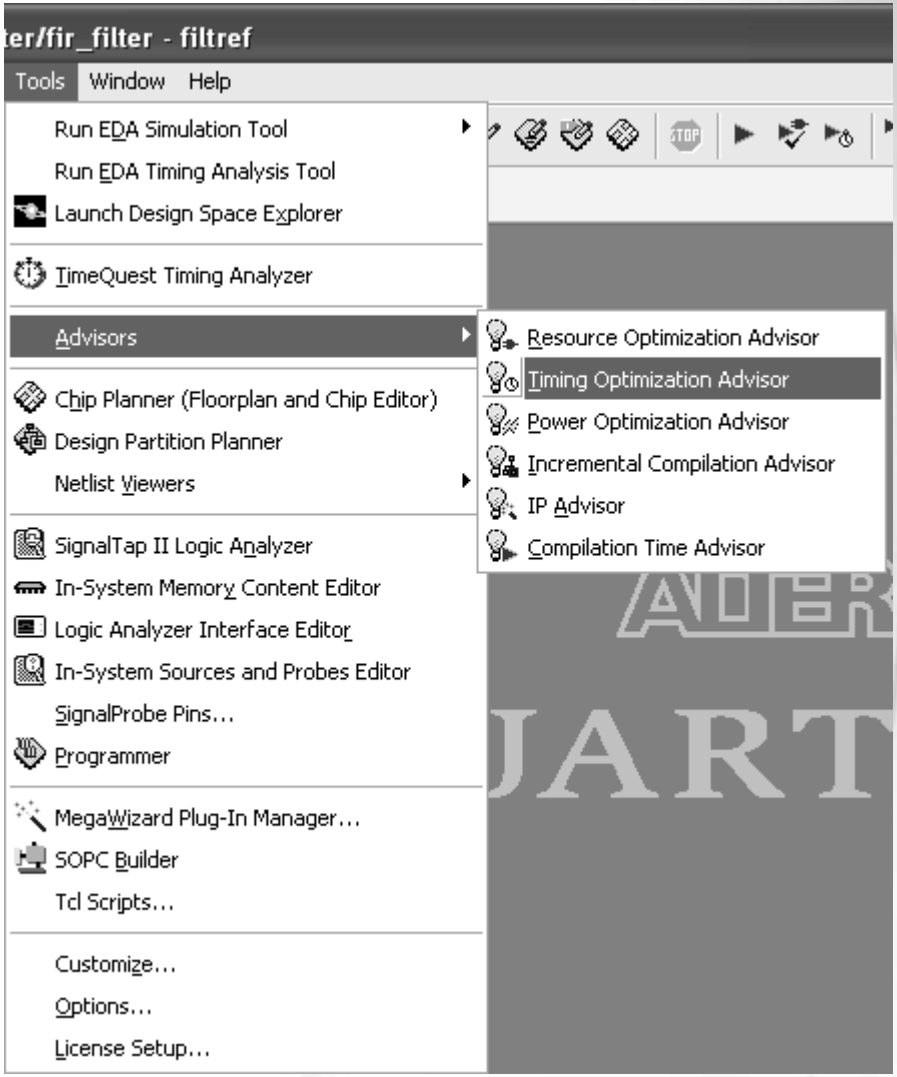


- If enabled, executes automatically during compilation
- May also run separately from Processing menu or Tasks window

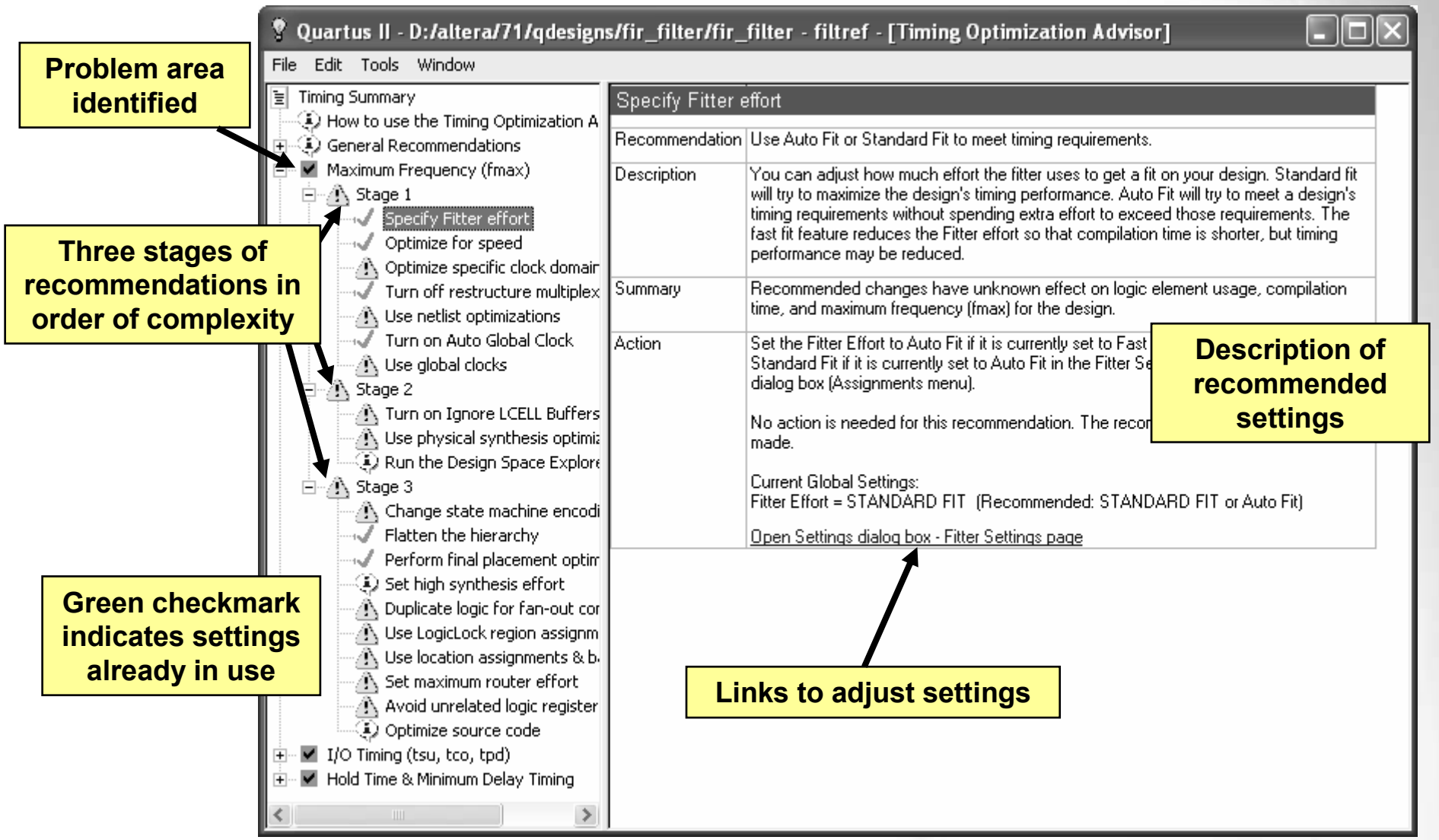
- **Checks for potential design issues**
  - Clocks
  - Reset
  - Non-synchronous design structure
  - Timing closure
  - Asynchronous clock domain data transfers
  - Signal race conditions
  - HardCopy
  - FSM

Optimization Advisors

- Provide design-specific recommendations (feedback) on optimizing designs
- Access through **Tools** menu or Tasks window
- Six types
  - Resource usage optimization
  - Timing (performance) optimization
  - Power optimization
  - Incremental compilation suggestions
  - Implementing IP (DDR3 & PCIE)
  - Compilation time reduction



Example Optimization Advisor



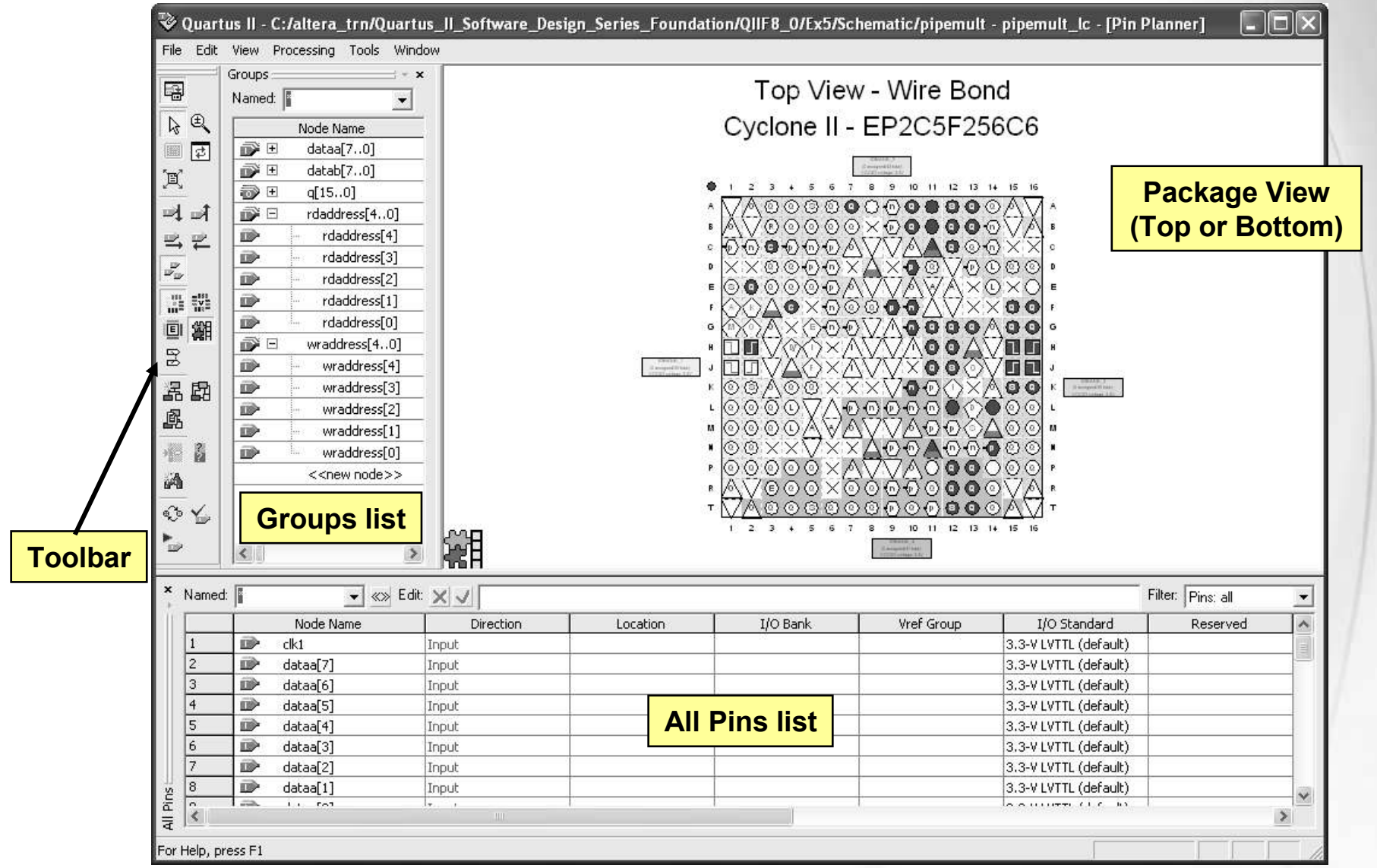
Quartus II Software Design Series: Foundation

I/O Planning

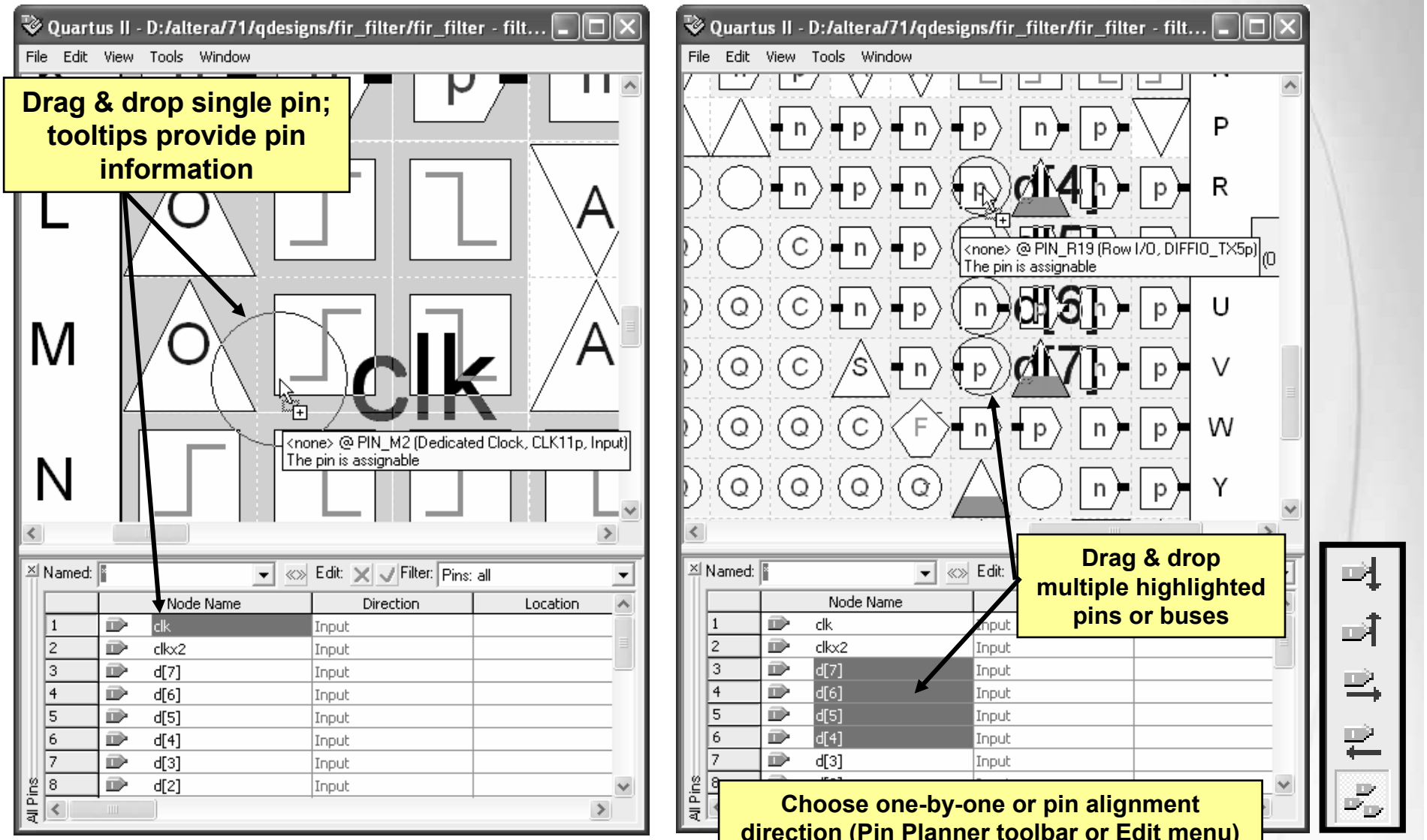
I/O Planning Need

- I/O standards increasing in complexity
- FPGA/CPLD I/O structure increasing in complexity
  - Results in increased pin placement guidelines
- PCB development performed simultaneously with FPGA design
  - Sometimes before!
- Pin assignments need to be verified earlier in design cycle
- Designers need easy way to transfer pin assignments into board tools

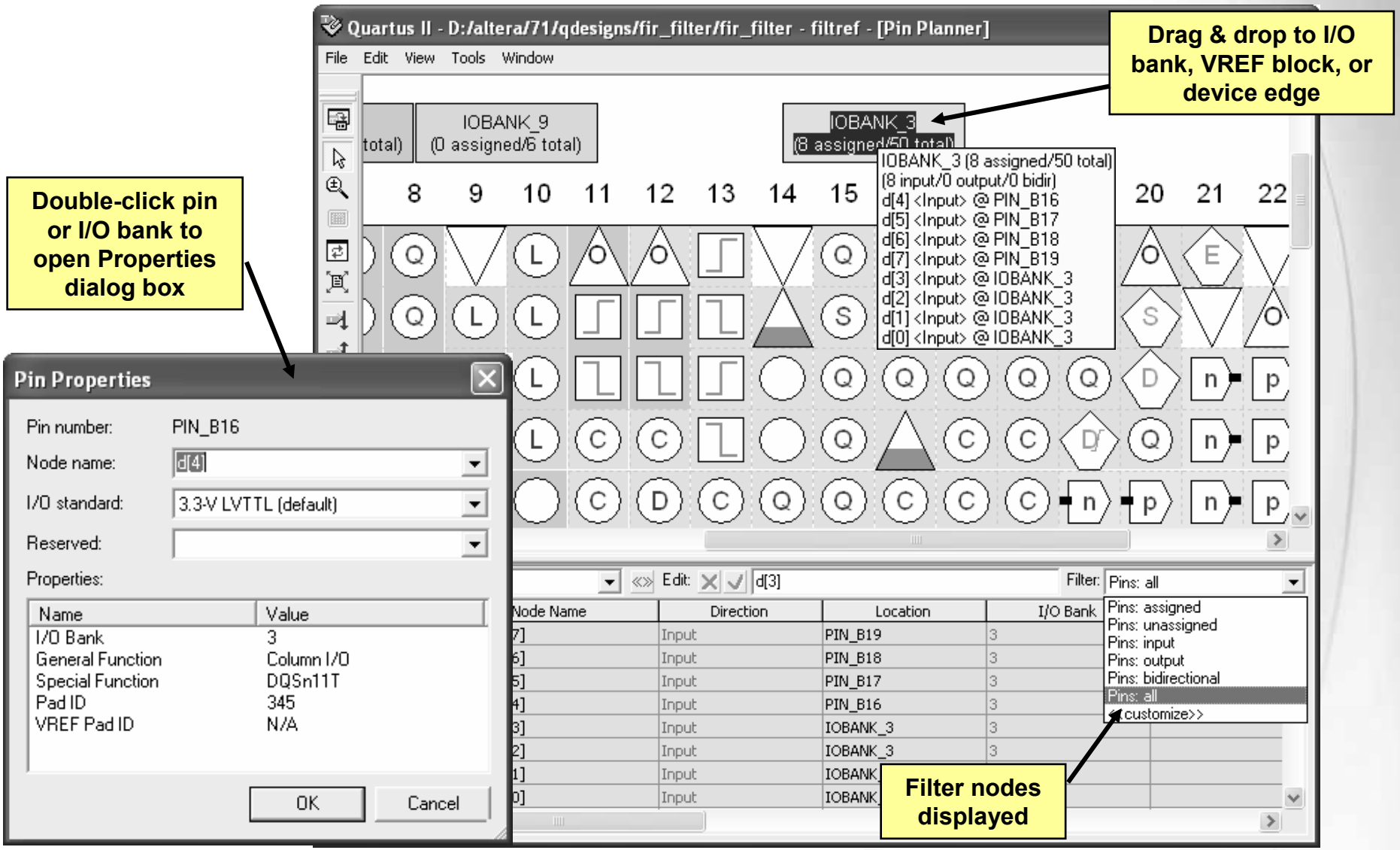
Pin Planner Window



Assigning Pins Using Pin Planner

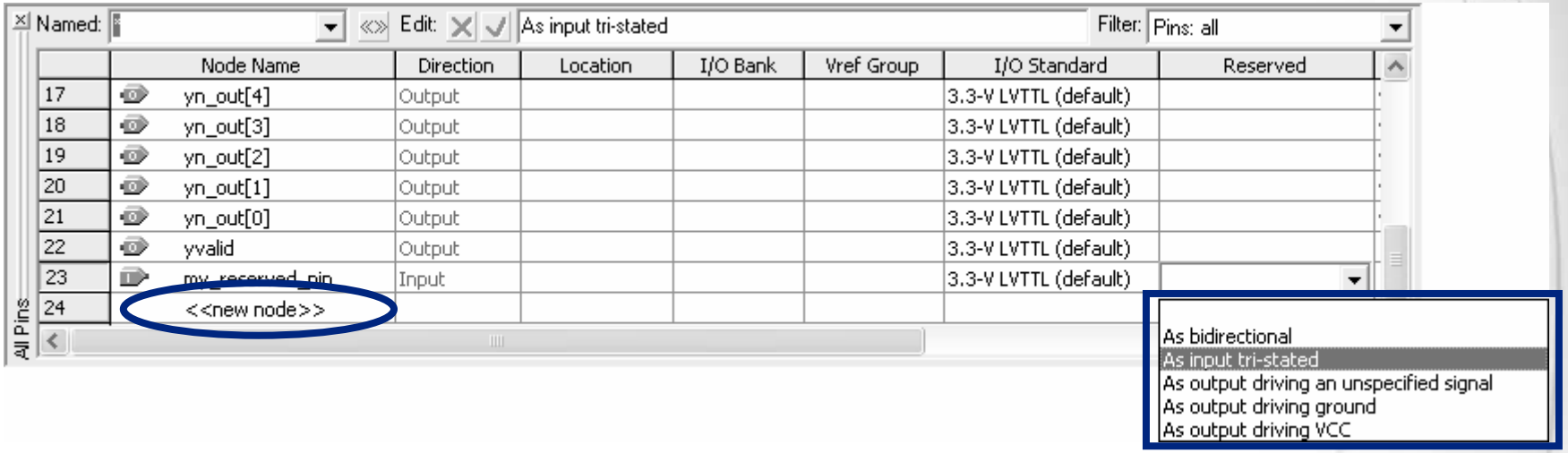


Assigning Pins Using Pin Planner (2)



Reserving I/O Pins

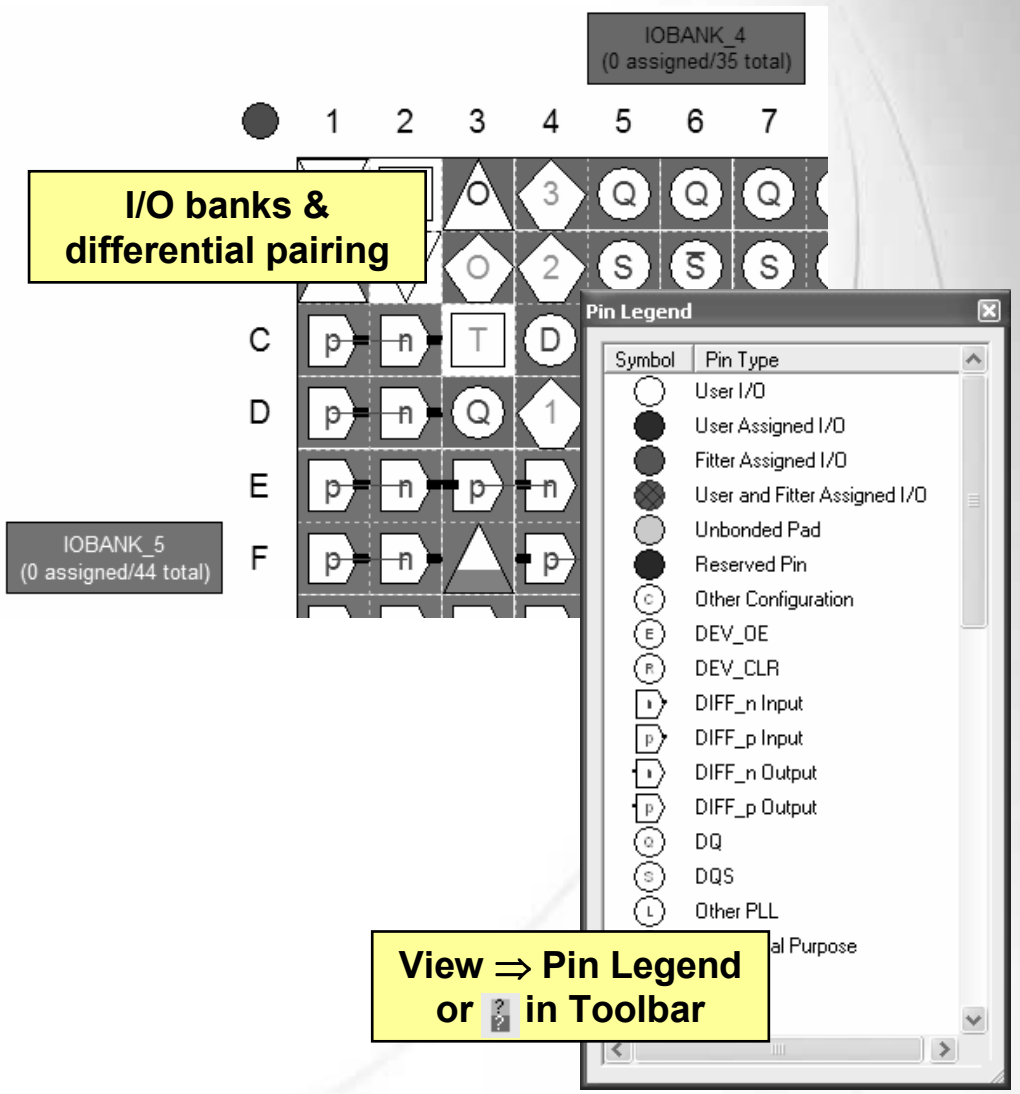
- Type reserved I/O name directly into Pins List & select reserve configuration



- Or right-click on pin in Package View and choose **Reserve ⇒ As…**
  - Pin name set to *user_reserve_<pin_number>*
  - Reserved pins colored blue in Package View
- Set initial state of other unused pins in Device settings in Settings dialog box (**Device & Pin Options** button)

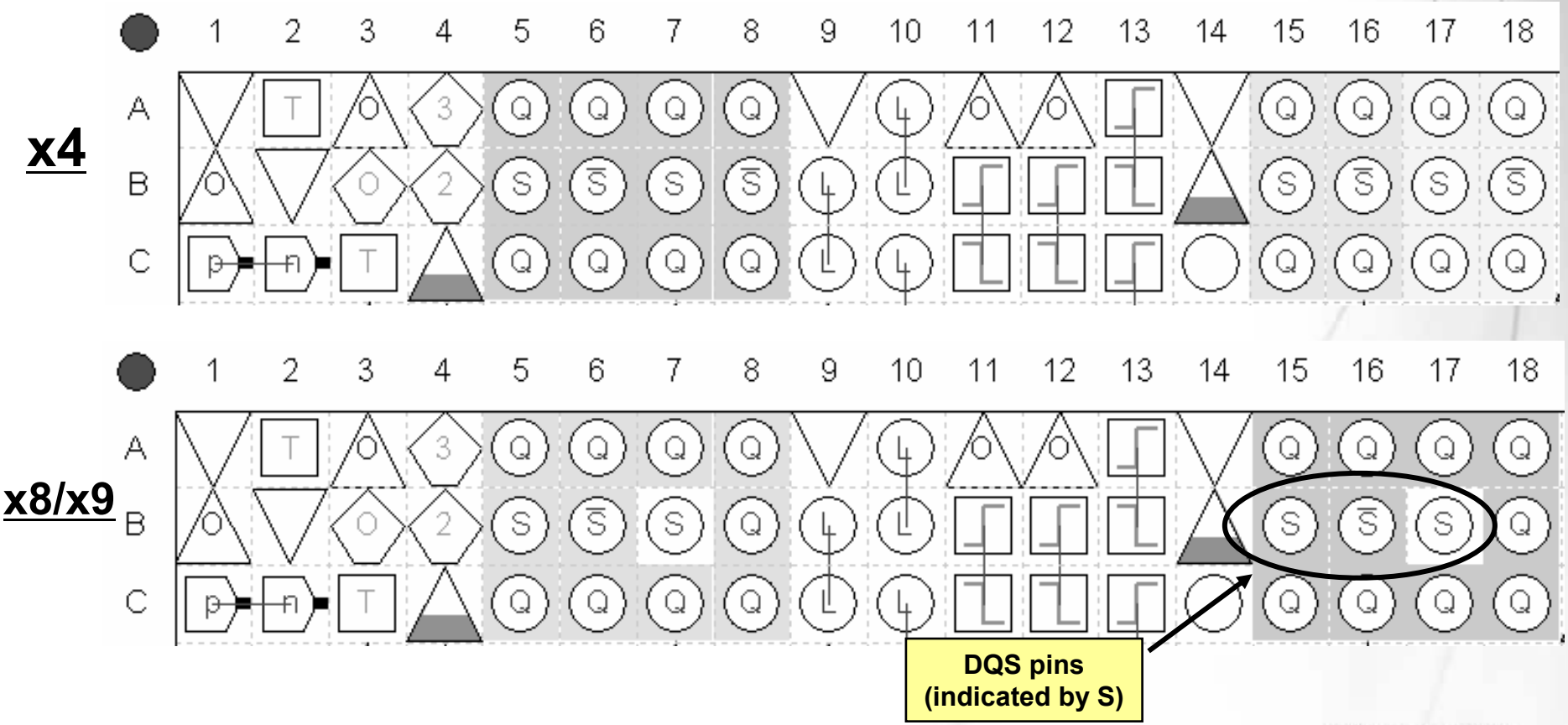
Other Pin Planner Features

- Displays (**View ⇒ Show**, Toolbar buttons, or right-click in Package View)
  - Device edges
  - I/O banks
  - VREF groups
  - Differential pin pairing
  - DQ/DQS pins (next slide)
- Easy-to-read pin legend



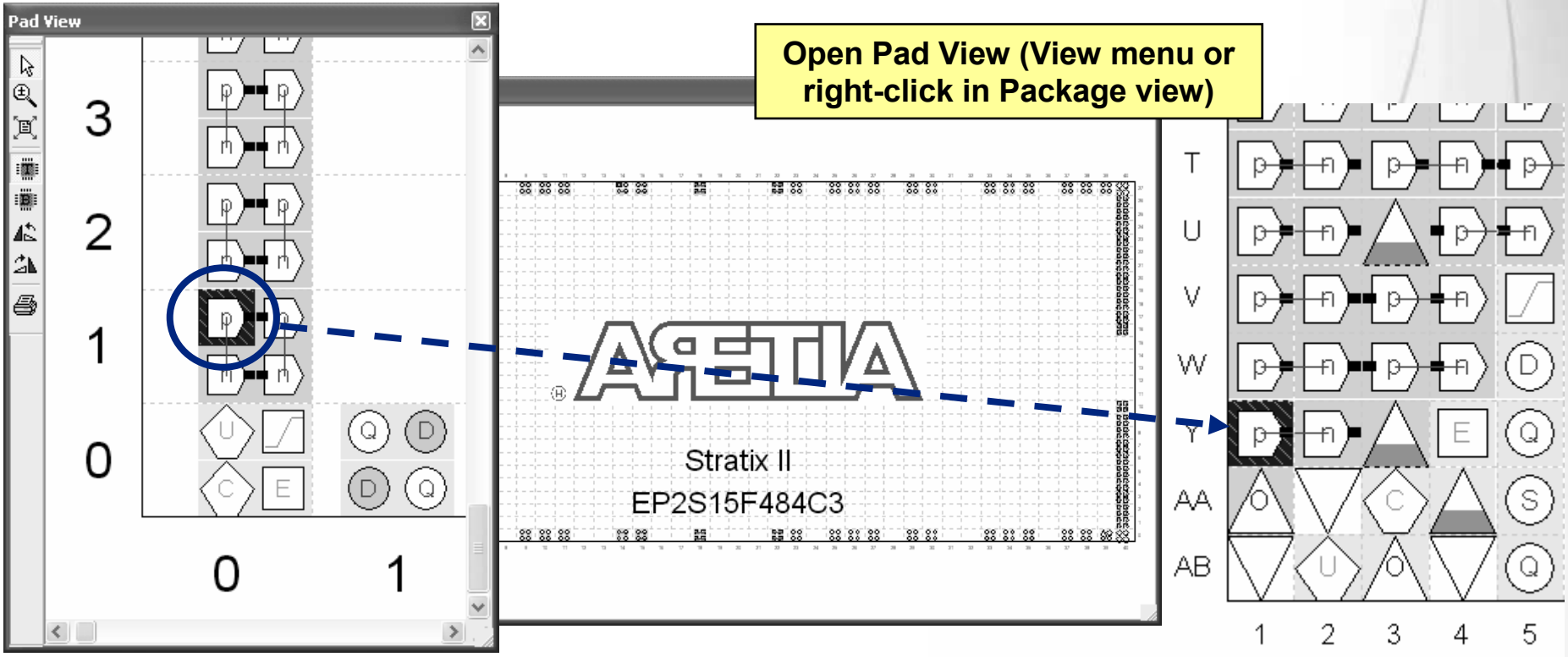
Show DQ/DQS Pins

- Show color-coded DQ/DQS sets in x4, x8/x9, x16/x18, or x32/x36 modes in the Package View for DDR interfaces



Pad View

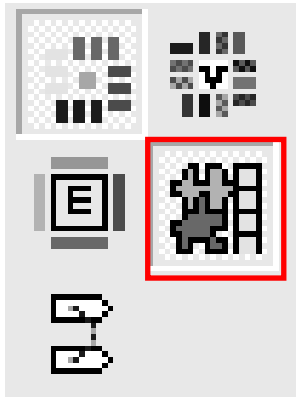
- Cross-reference package pin location to silicon pad location
  - Assign pins in Pad View based on pad location
- Reversed “Altera” indicates flip-chip die

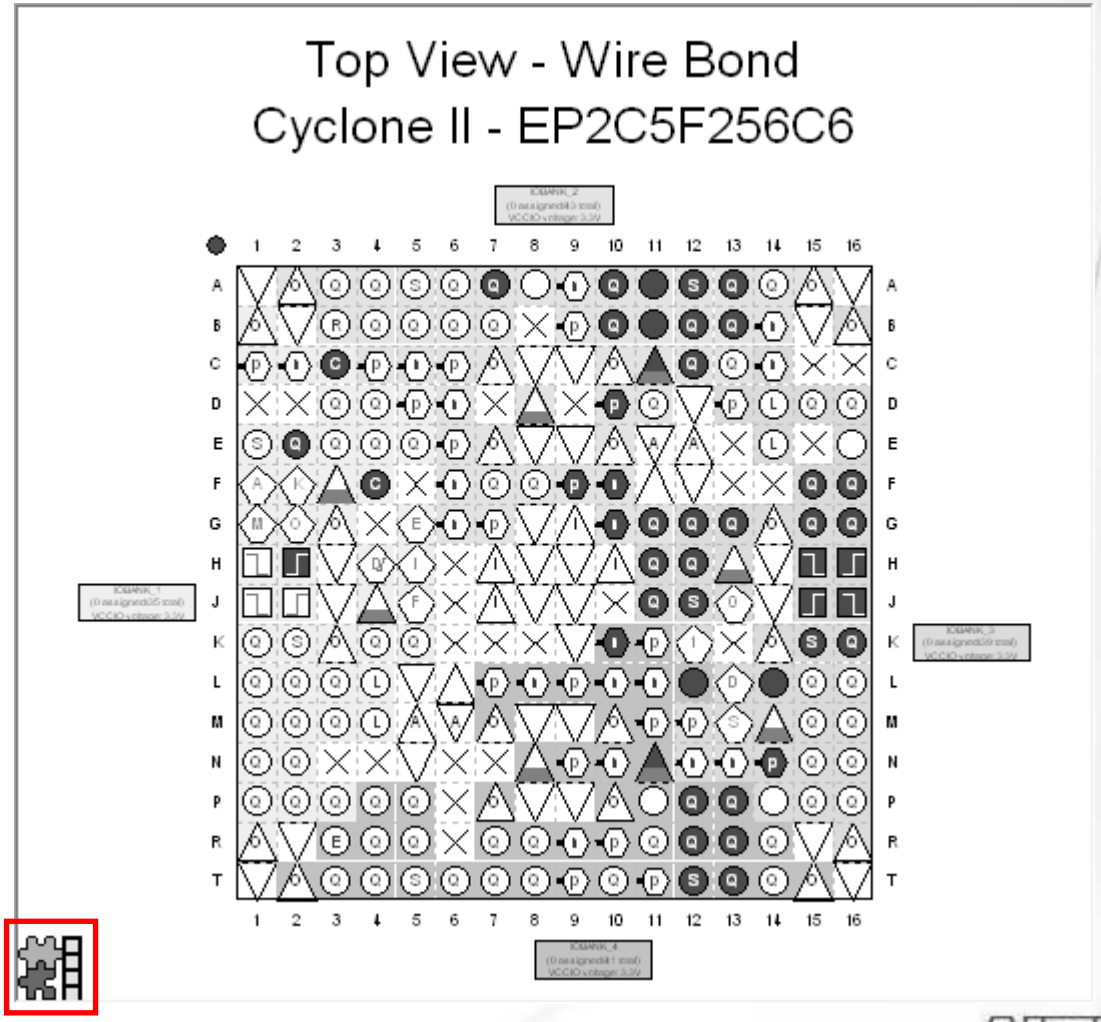


Show Fitter Placements

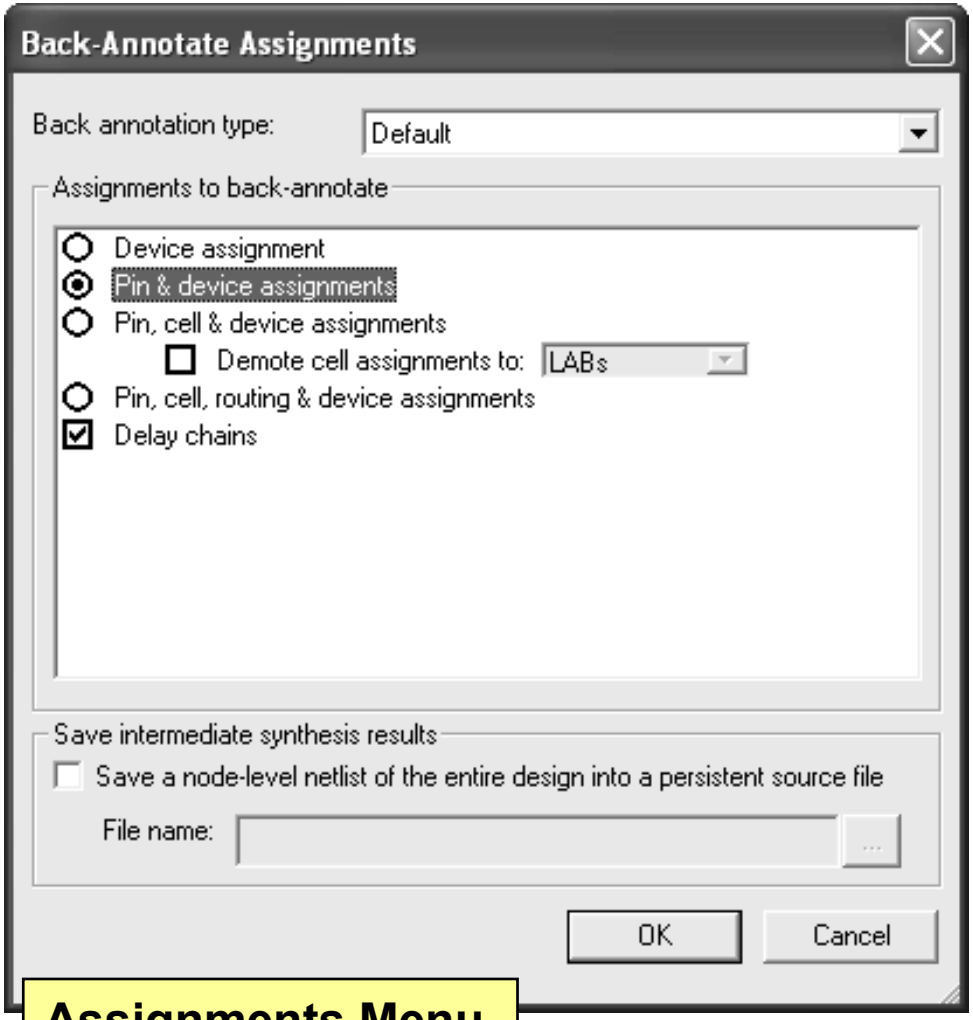
- View I/O locations automatically selected by Fitter

View ⇒ Show or in Toolbar



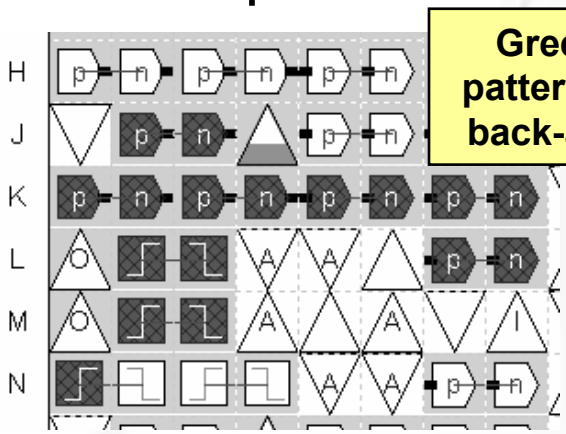


Back-Annotation



Assignments Menu

- Use to lock fitter-chosen (green) pin assignments for future compilations
  - Copies device & resource locations chosen by fitter into QSF file
    - Pins
    - Logic
    - Routing
- "Locks down" locations in floorplan

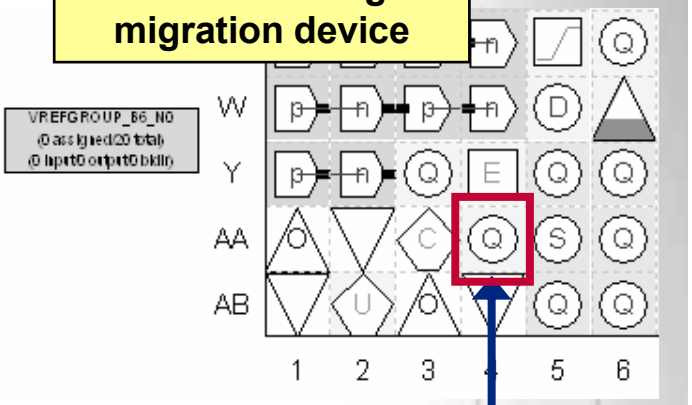


Green/brown pattern indicates back-annotation

Pin Migration View

- Select migration devices in Device Settings
- View & compare pin function differences between migration devices
- Package View adjusts to prevent non-migratable assignments

Before adding migration device



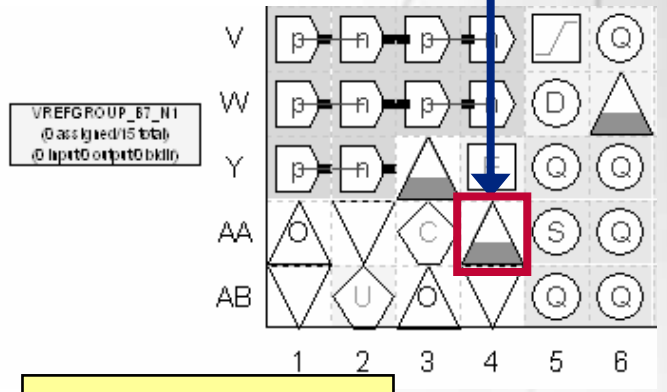
View ⇒ Pin Migration View or right-click in Package View

Pin Migration View

Current Device: EP2S15F484C3

	Pin Number	Migration Result			Migration Devices					
					EP2S15F484C3			EP2S60F484I4		
		Pin Function	I/O Bank	VREF Group	Pin Function	I/O Bank	VREF Group	Pin Function	I/O Bank	VREF Group
1	PIN_A6	Column I/O	4	B4_N1	Column I/O	4	B4_N1	Column I/O	4	B4_N2
2	PIN_A7	Column I/O	4	B4_N1	Column I/O	4	B4_N1	Column I/O	4	B4_N2
3	PIN_A8	Column I/O	4	B4_N1	Column I/O	4	B4_N1	Column I/O	4	B4_N2
4	PIN_A10	Column I/O	9	B4_N1	Column I/O	9	B4_N1	Column I/O	9	B4_N2
5	PIN_A16	Column I/O	3	B3_N0	Column I/O	3	B3_N0	Column I/O	3	B3_N1
6	PIN_A17	Column I/O	3	B3_N0	Column I/O	3	B3_N0	Column I/O	3	B3_N1
7	PIN_A18	Column I/O	3	B3_N0	Column I/O	3	B3_N0	Column I/O	3	B3_N1
8	PIN_A19	Column I/O	3	B3_N1	Column I/O	3	B3_N1	Column I/O	3	B3_N2
9	PIN_A21	Dedicated...	3	B3_N1	Dedicated...	3	B3_N1	Dedicated...	3	B3_N2
10	PIN_AA4	VREFB7N2	7		Column I/O	7	B7_N1	VREFB7N2	7	B7_N2
11	PIN_AA5	Column I/O	7	B7_N0	Column I/O	7	B7_N0	Column I/O	7	B7_N1
12	PIN_AA12	Column I/O	8	B8_N1	Column I/O	8	B8_N1	Column I/O	8	B8_N2
13	PIN_AA13	Column I/O	8	B8_N1	Column I/O	8	B8_N1	Column I/O	8	B8_N2

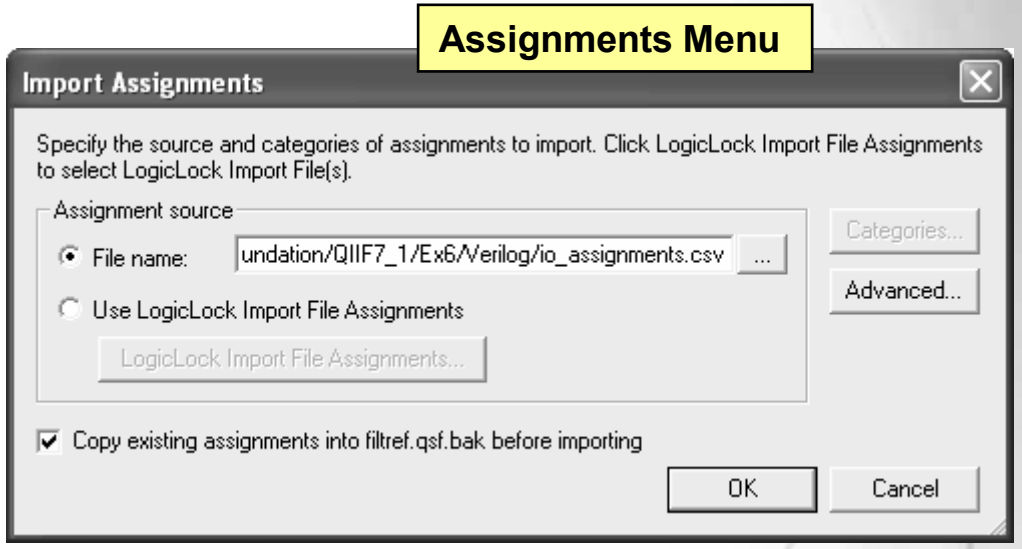
Device... Pin Finder... Show only highlighted pins Show migration differences



After adding migration device

Import/Export via CSV

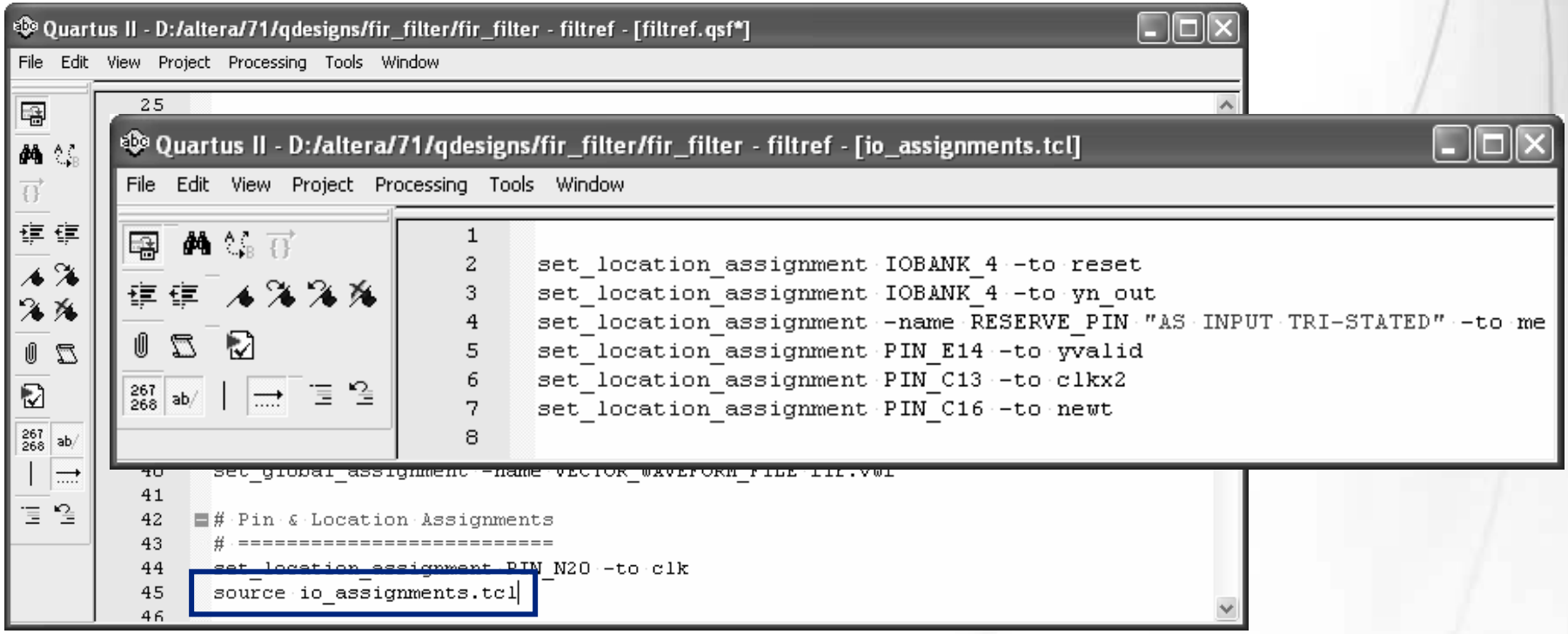
- Use spreadsheet Comma Separated Value (.CSV) file to enter or edit I/O locations
- Convenient for transferring assignments between project revisions
- CSV column names must match Pin Planner column headings
  - To
    - Pin name
  - Assignment Name
    - Location
  - Value
    - PIN_<*pin_number*>
  - I/O standard



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

Type I/O Assignments & Scripting

- Type pin-related assignments directly into QSF
- Type pin-related assignments into separate Tcl
  - Source Tcl file in project QSF
  - Execute Tcl file to write assignments into QSF



I/O Assignment Analysis Command

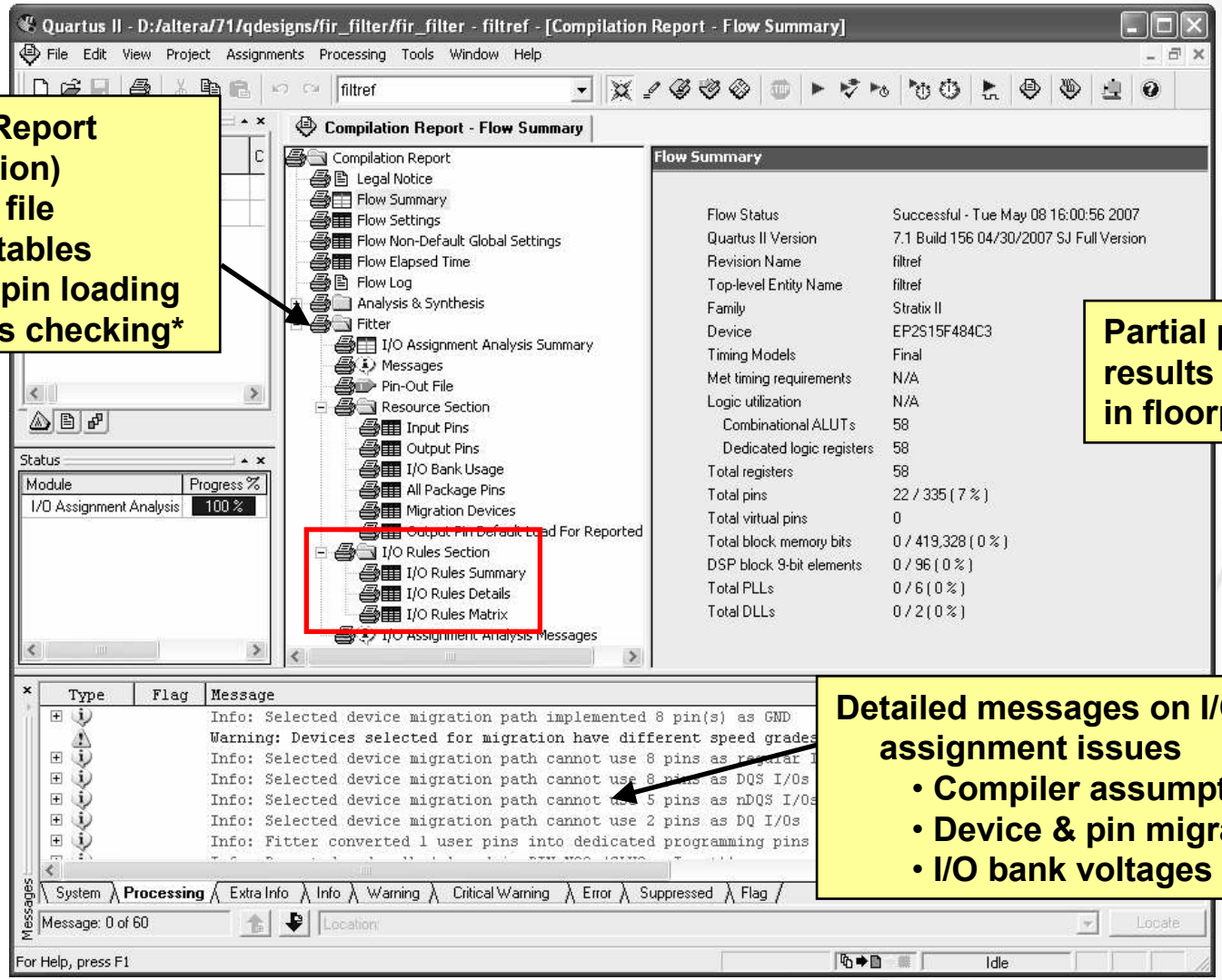
Run from Pin Planner toolbar

- Use to check legality of all I/O assignments without full compilation
- Requirements
  - I/O declaration
    - HDL port declaration
    - Reserved pin
  - Pin-related assignments
    - I/O standard
    - Current strength
    - Pin location (pin, bank, edge)
    - PCI clamping diode
    - Toggle rate

Processing menu ⇒ Start ⇒ Start I/O Assignment Analysis or Tasks window

I/O Assignment Analysis Output

Compilation Report (Fitter section)
- Pin-out file
- I/O pin tables
- Output pin loading
- I/O rules checking*



Partial placement results also shown in floorplan

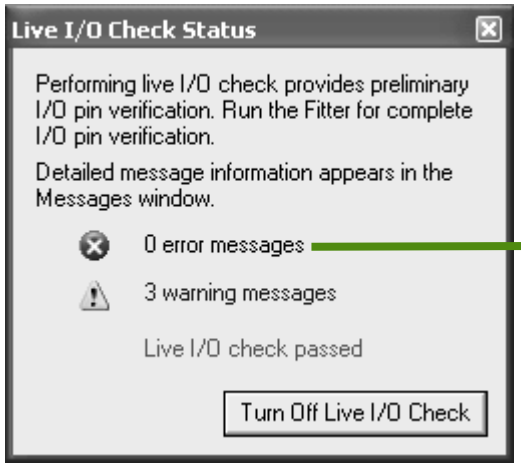
Detailed messages on I/O assignment issues
- Compiler assumptions
- Device & pin migration issues
- I/O bank voltages & standards

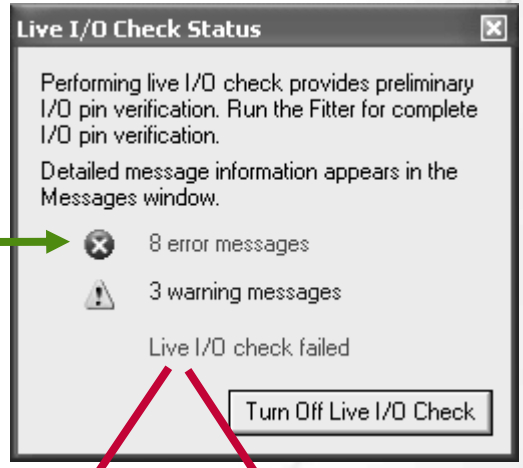
Note: See Appendix for special reports and information generated only for Arria GX, Stratix II, II GX, and HardCopy II devices

Live I/O Checking

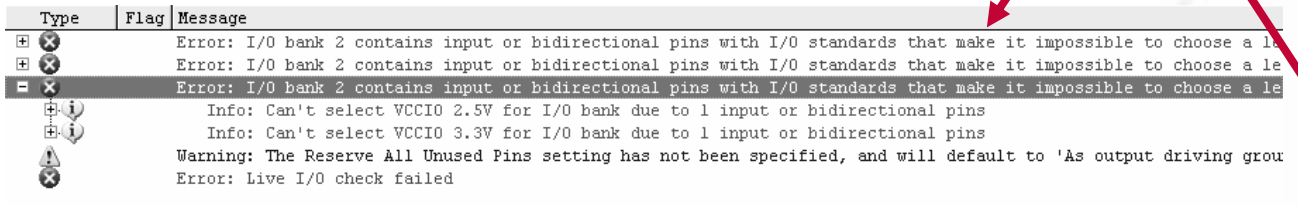
- Perform limited I/O checks live as assignments made
- Status window alerts to failing assignments
  - Errors detailed in Messages window and Package view tooltips
- Full I/O Assignment Analysis still required

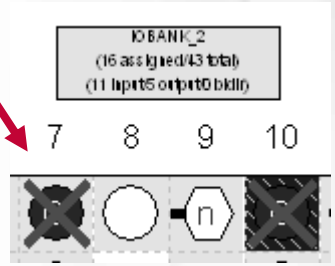
In Pin Planner, View menu ⇒ Live I/O Check Status Window



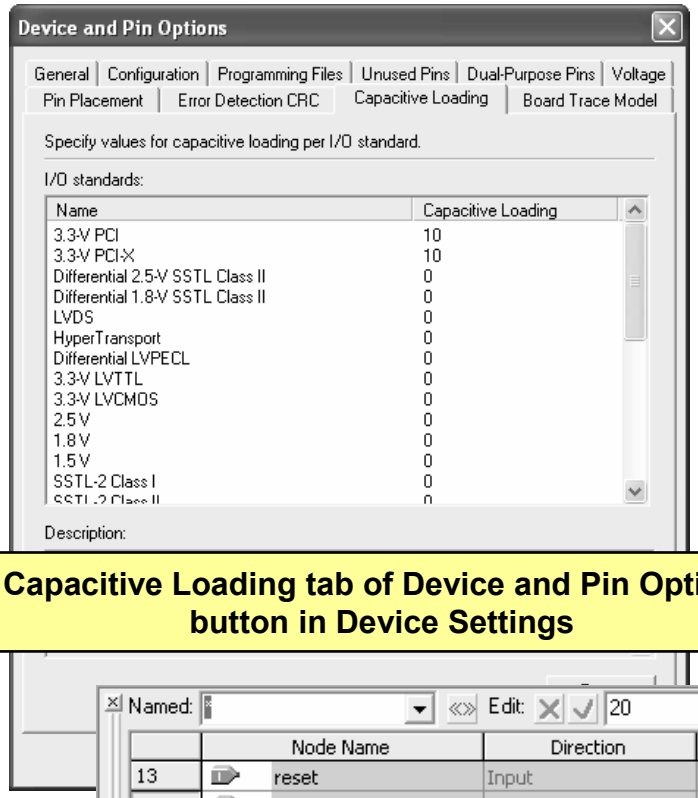


Turn on or off in Pin Planner toolbar



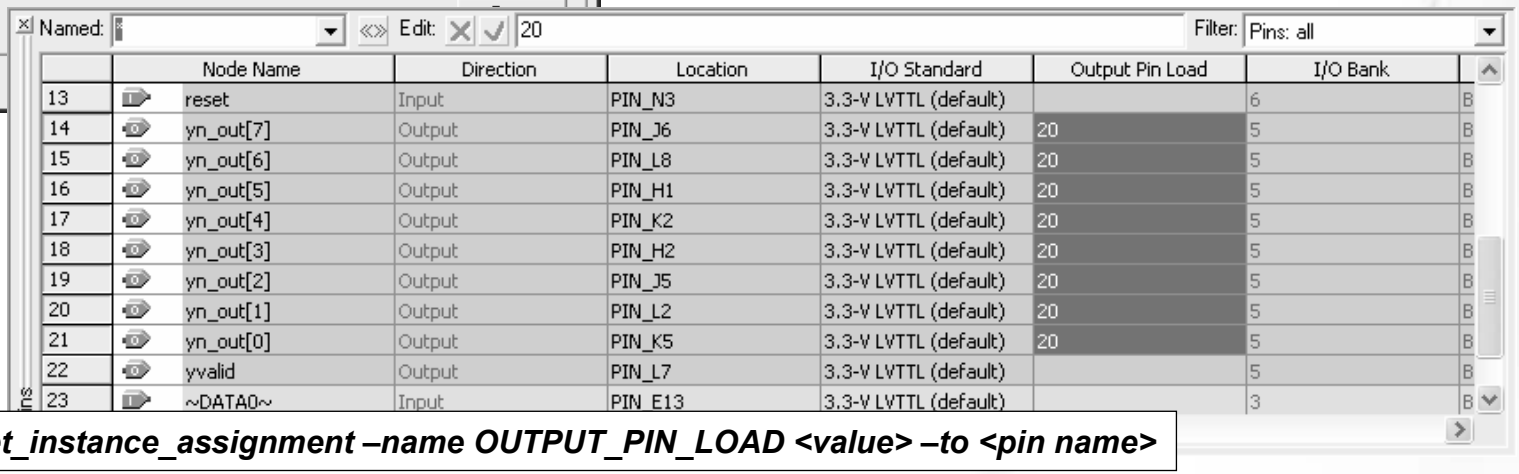


"Board-Aware" Settings: Output Pin Load



Capacitive Loading tab of Device and Pin Options button in Device Settings

- 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
- Specify for entire I/O standard in Device Settings
- Apply to individual output or bidirectional pins in Assignment Editor or Pin Planner All Pins list



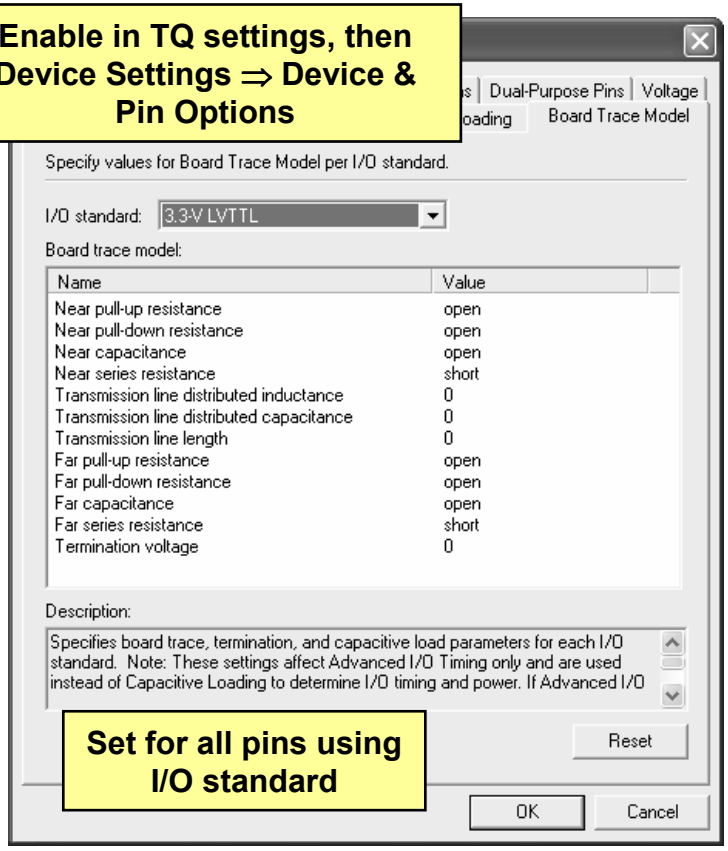
	Node Name	Direction	Location	I/O Standard	Output Pin Load	I/O Bank	
13	reset	Input	PIN_N3	3.3-V LVTTL (default)		6	B
14	yn_out[7]	Output	PIN_J6	3.3-V LVTTL (default)	20	5	B
15	yn_out[6]	Output	PIN_L8	3.3-V LVTTL (default)	20	5	B
16	yn_out[5]	Output	PIN_H1	3.3-V LVTTL (default)	20	5	B
17	yn_out[4]	Output	PIN_K2	3.3-V LVTTL (default)	20	5	B
18	yn_out[3]	Output	PIN_H2	3.3-V LVTTL (default)	20	5	B
19	yn_out[2]	Output	PIN_J5	3.3-V LVTTL (default)	20	5	B
20	yn_out[1]	Output	PIN_L2	3.3-V LVTTL (default)	20	5	B
21	yn_out[0]	Output	PIN_K5	3.3-V LVTTL (default)	20	5	B
22	yvalid	Output	PIN_L7	3.3-V LVTTL (default)		5	B
23	~DATA0~	Input	PIN_E13	3.3-V LVTTL (default)		3	B

Tcl: set_instance_assignment –name OUTPUT_PIN_LOAD <value> –to <pin name>

Advanced I/O Timing

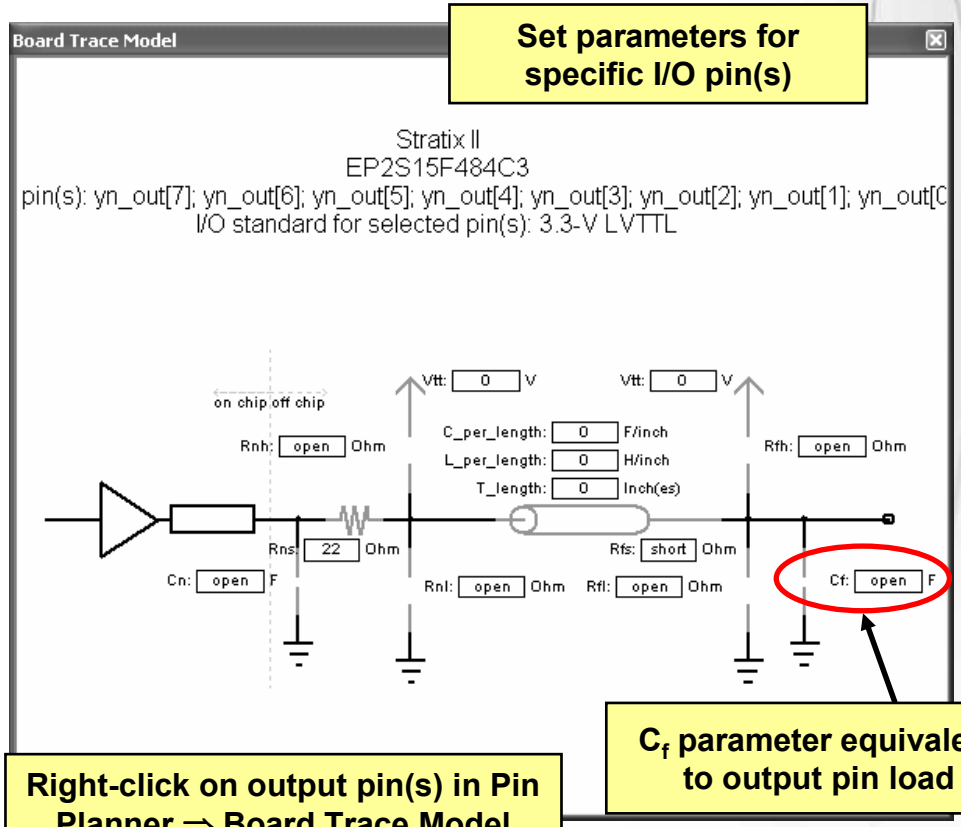
- Enhances analysis (over capacitive loading) by allowing user to enter board-level parameters (Cyclone III, Stratix II, & Stratix III devices only)
  - Use in lieu of or in addition to HSPICE & IBIS modeling
- View signal integrity metrics in Compilation Report (TimeQuest folder)

Enable in TQ settings, then Device Settings ⇒ Device & Pin Options



Set for all pins using I/O standard

Set parameters for specific I/O pin(s)



Right-click on output pin(s) in Pin Planner ⇒ Board Trace Model

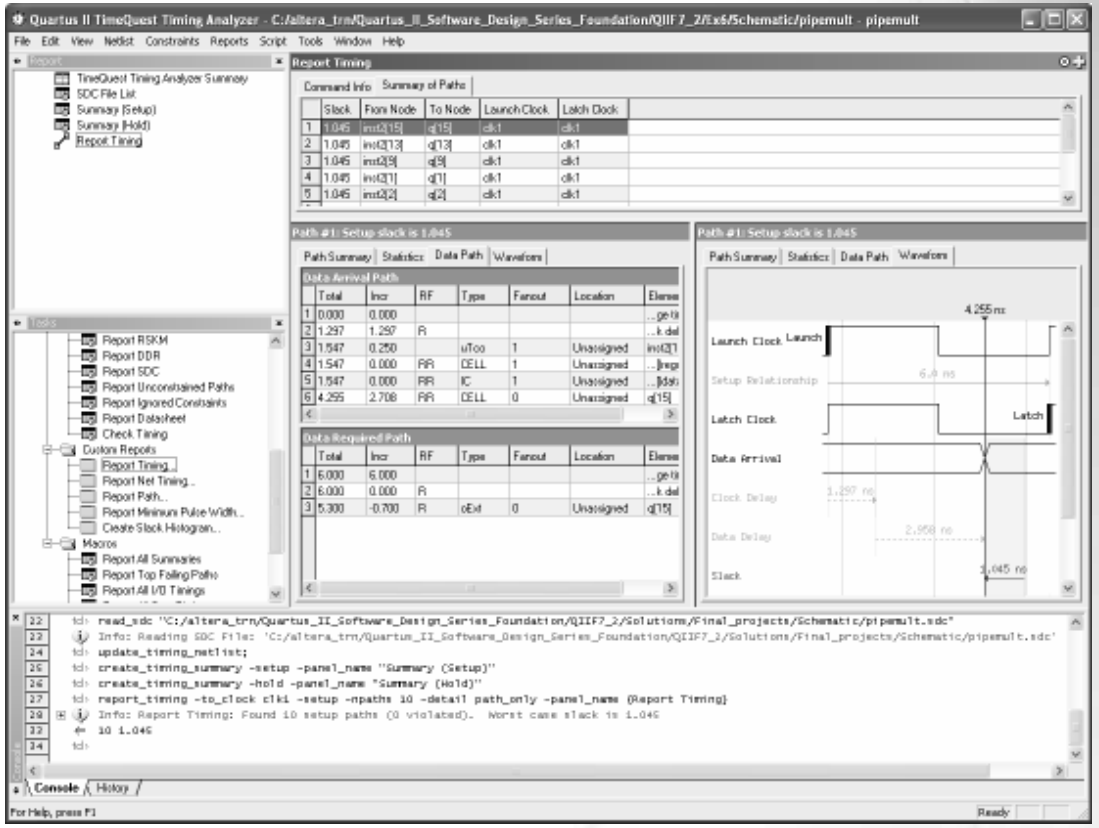
C_f parameter equivalent to output pin load

Quartus II Software Design Series: Foundation

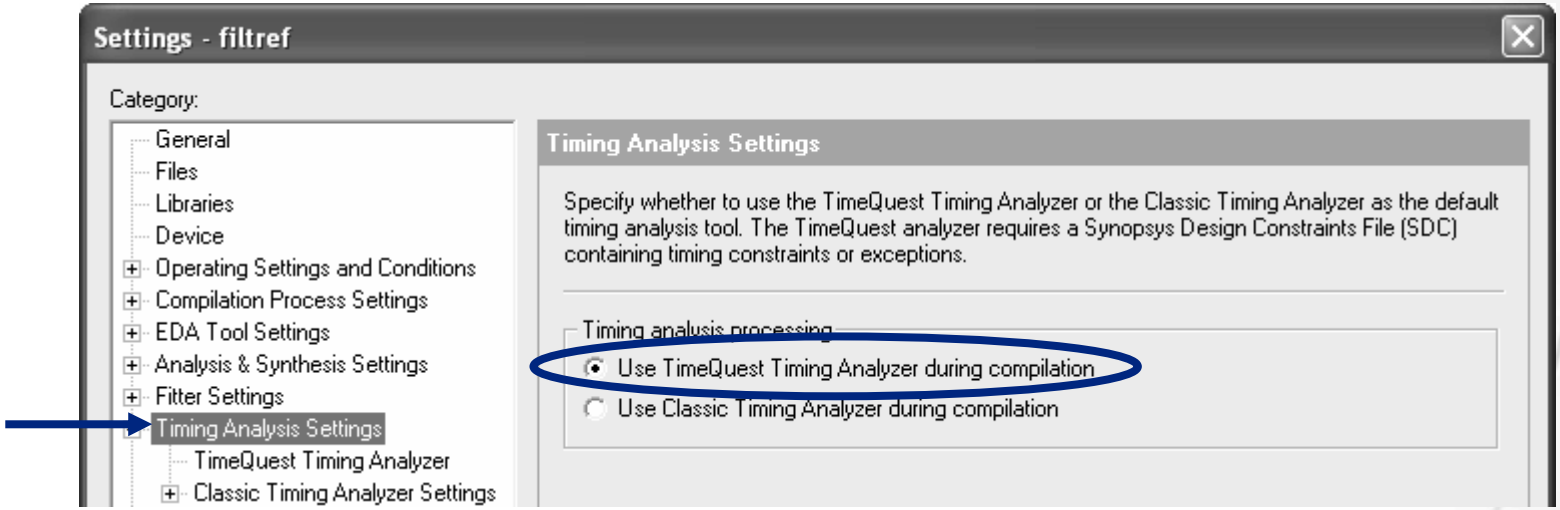
Timing Analysis

TimeQuest Timing Analyzer (TA)

- Timing engine in Quartus II software
- Provides timing analysis solution for all levels of experience
- Features
  - Synopsys Design Constraints (SDC) support
    - Standardized constraint methodology
  - Easy-to-use interface
    - Constraint entry
    - Standard reporting
  - Scripting emphasis
    - Presentation focuses on using GUI



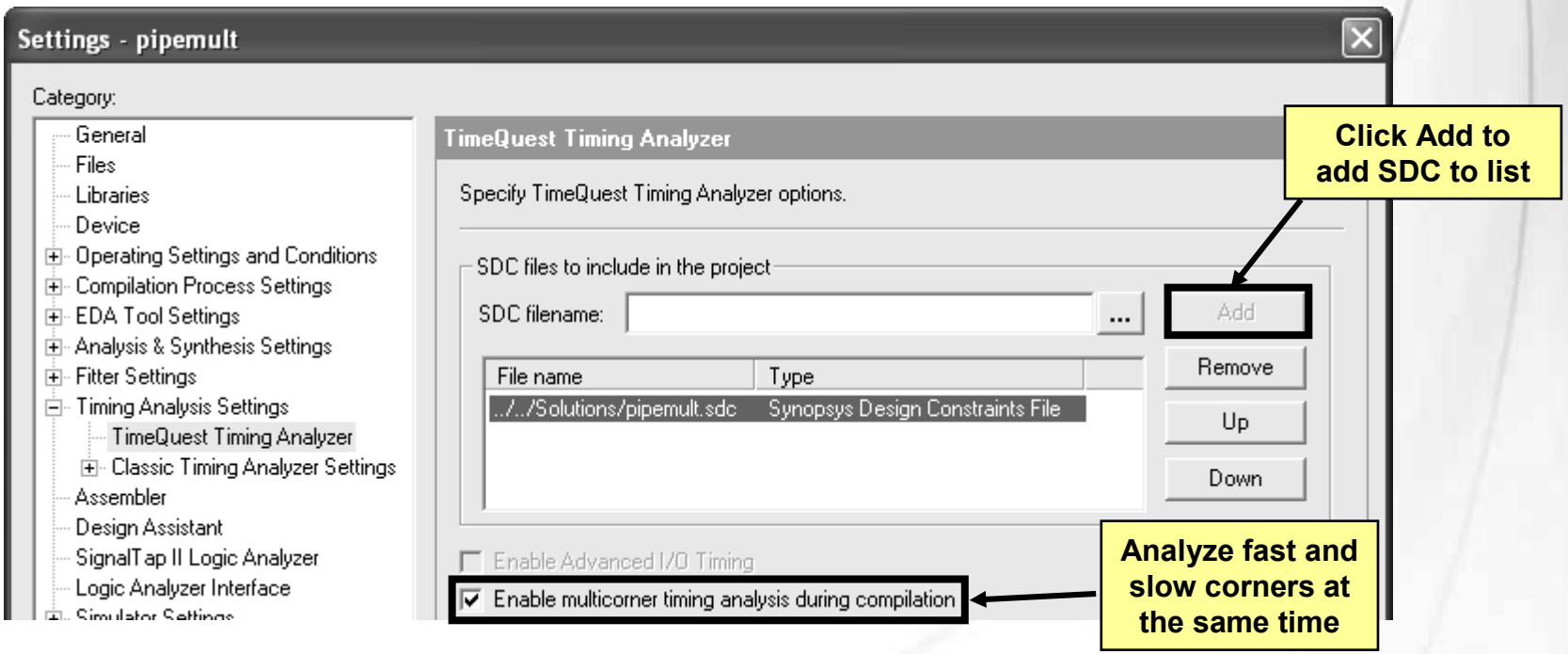
Enabling in the Quartus II Software



Notes:
- **Arria GX device *only* supports Timequest TA.**
- **TimeQuest TA is enabled by default for new Stratix III and Cyclone III designs.**

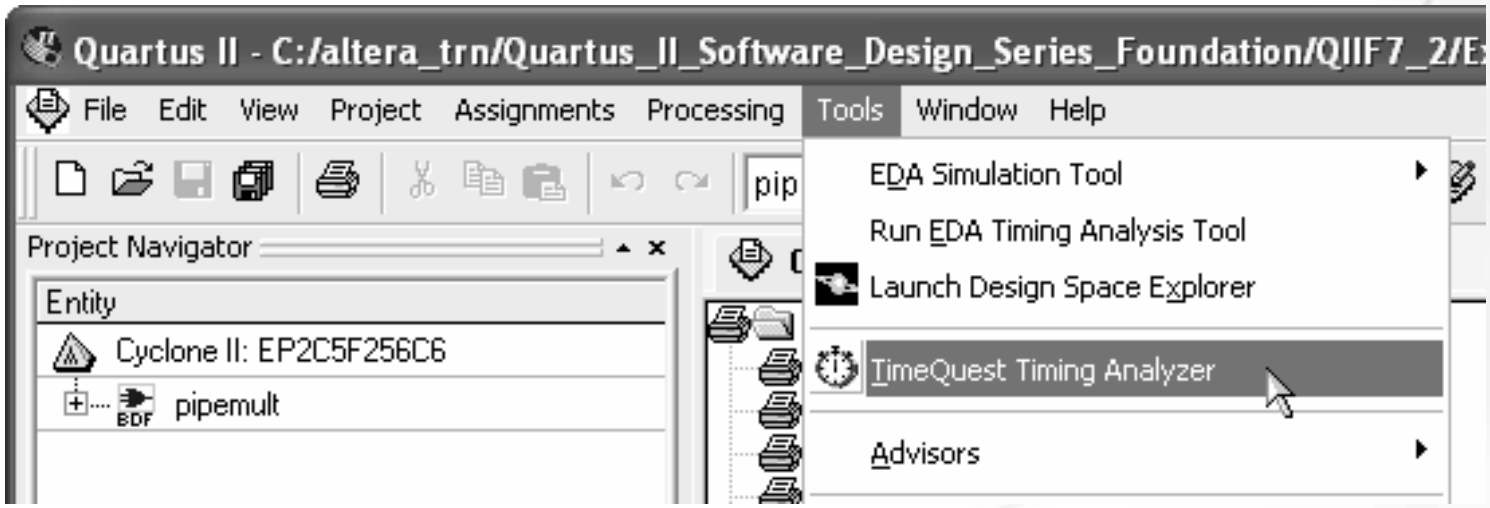
Adding SDC File to Quartus II Project

- Add SDC files to TimeQuest Timing Analyzer page of Settings dialog box
- Multicorner timing analysis checks all process corners in one analysis
  - On by default for Cyclone II & III, Stratix II & III devices

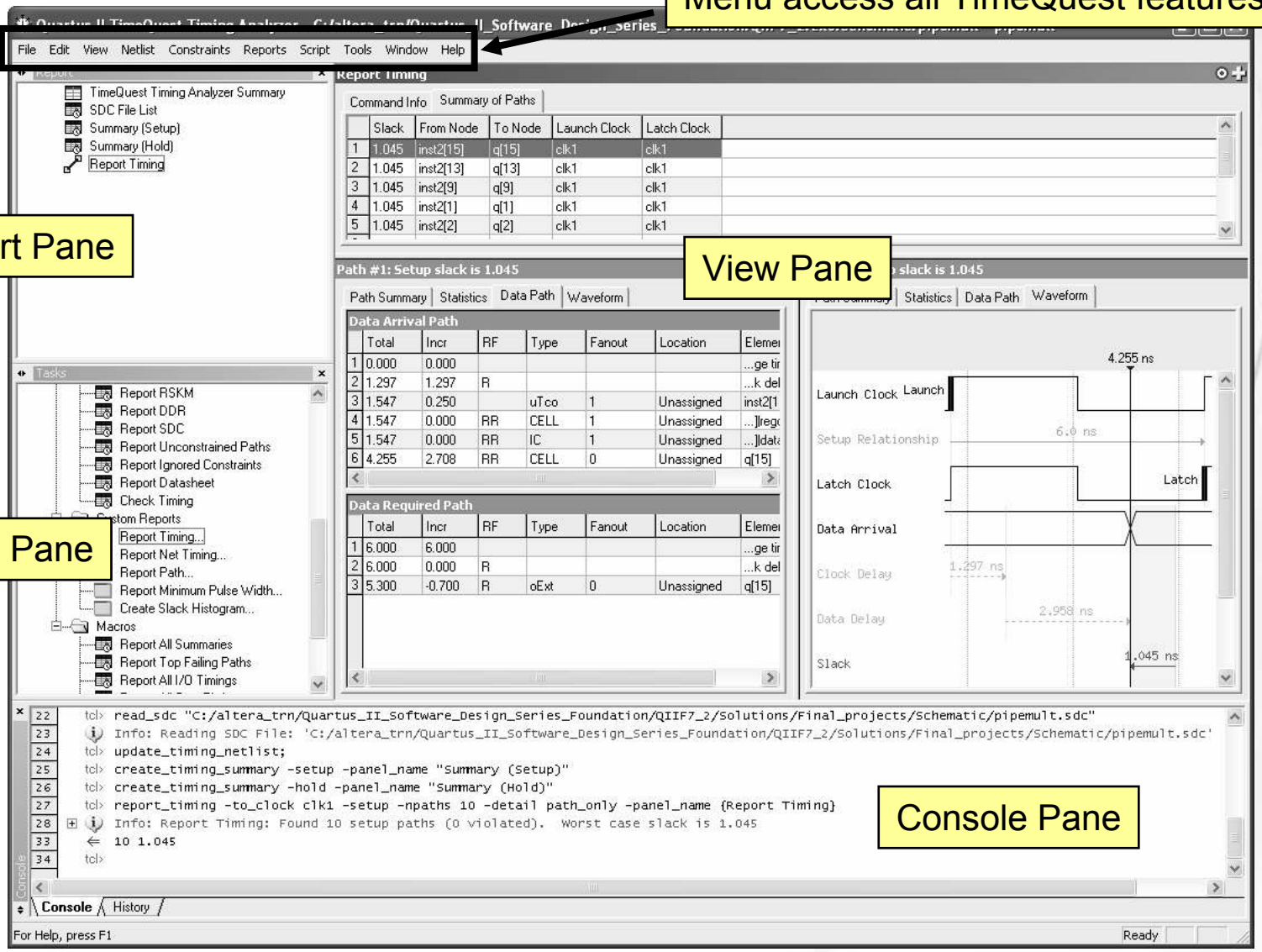


Opening the TimeQuest Interface

- Toolbar button 
- **Tools** menu
- Tasks window
- Stand-alone mode
  - **`quartus_staw`**
- Command line

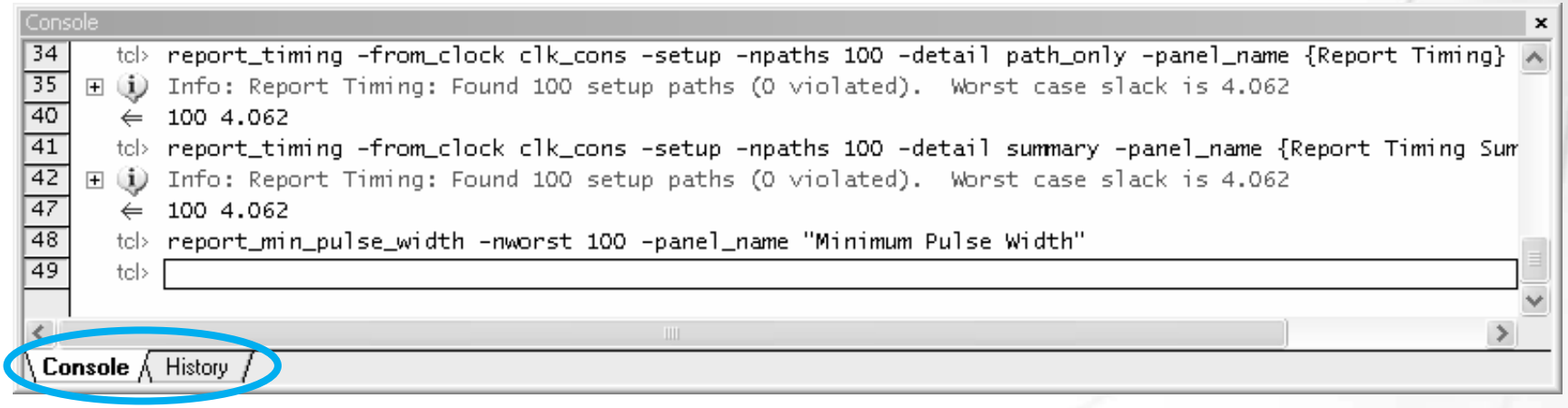


TimeQuest GUI



Console pane

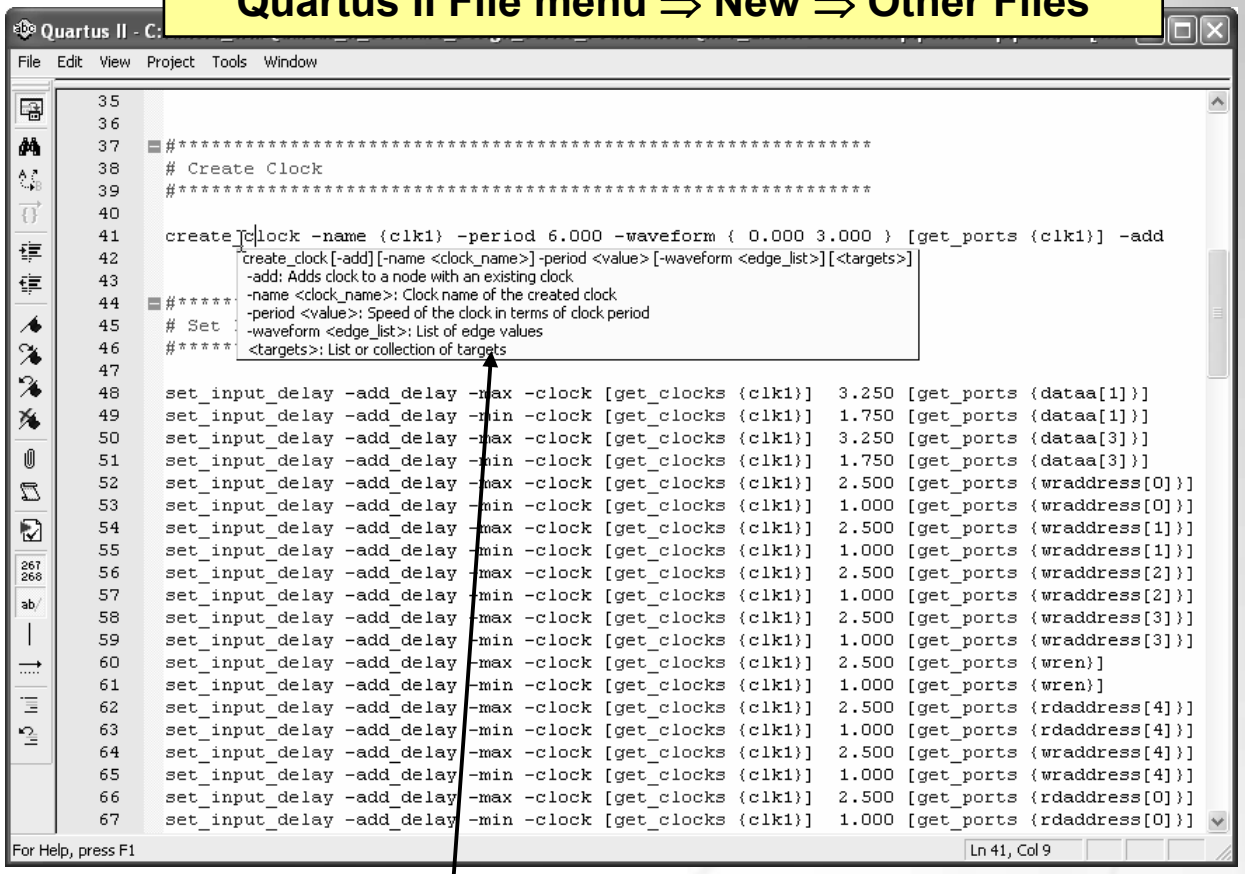
- Allows direct entry and execution of SDC & Tcl commands
  - Displays equivalent of command executed by GUI
- Displays TimeQuest output messages
- History tab records all executed SDC & Tcl commands
  - Copy & paste to create scripts or SDC files



SDC File Editor = Quartus II Text Editor

- Use Quartus II editor to create and/or edit SDC
- SDC editing unique features (for .sdc files)
  - Access to GUI dialog boxes for constraint entry (**Edit ⇒ Insert Constraint**)
  - Syntax coloring
  - Tooltip syntax help

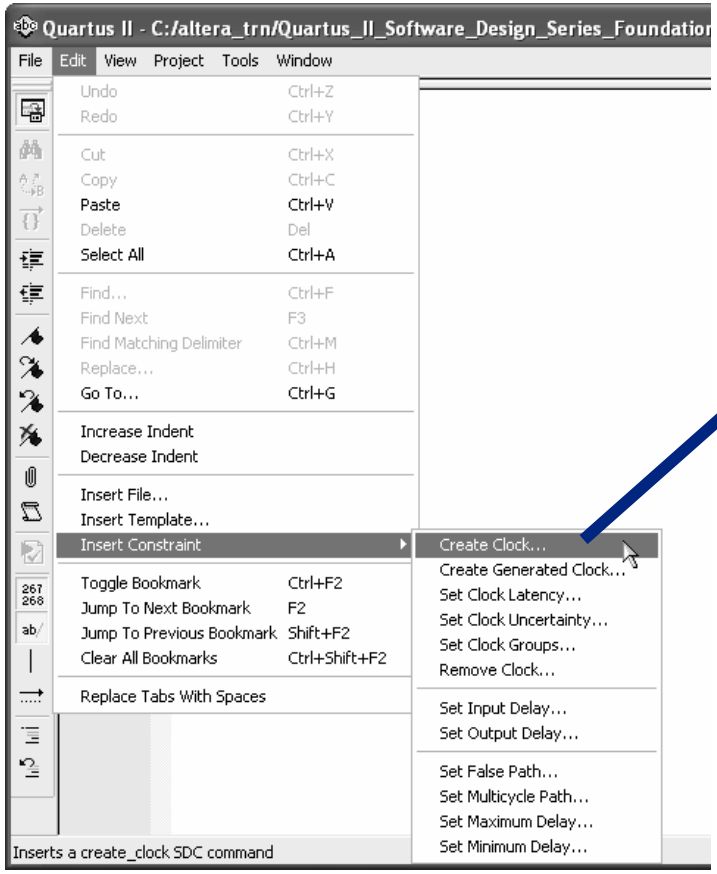
TimeQuest File menu ⇒ New/Open SDC File
Quartus II File menu ⇒ New ⇒ Other Files

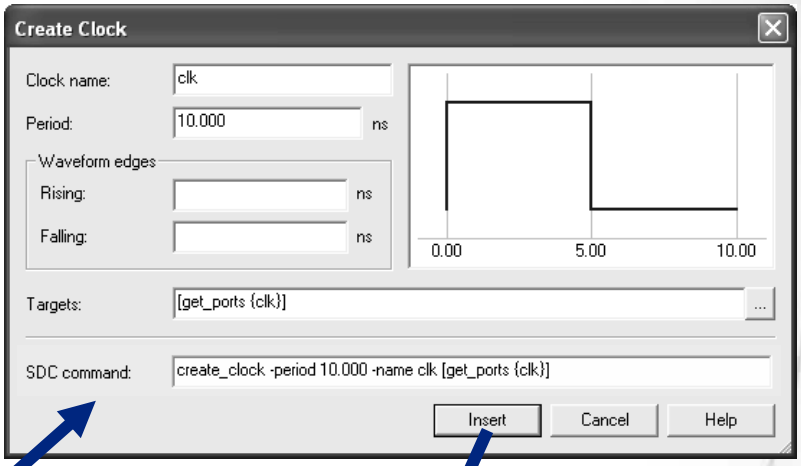


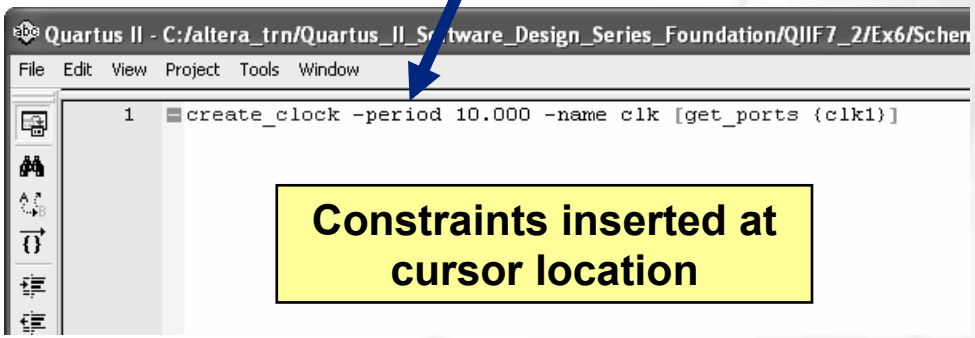
Place cursor over command to see tooltip

SDC File Editor (cont.)

Construct an SDC file using the TimeQuest graphical constraint creation tools







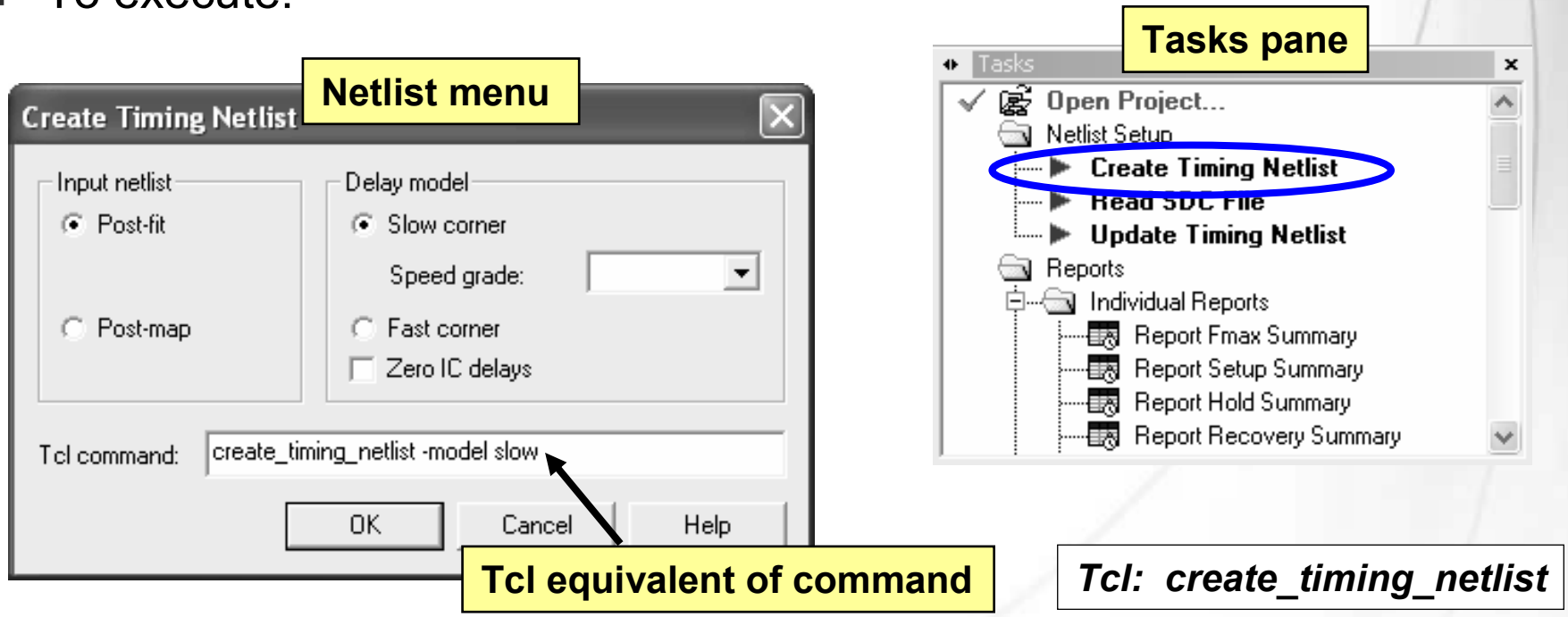
Constraints inserted at cursor location

Steps to Using TimeQuest Tool

1. Generate timing netlist
2. Enter SDC constraints by creating or reading in an SDC file
3. Update timing netlist
4. Generate timing reports

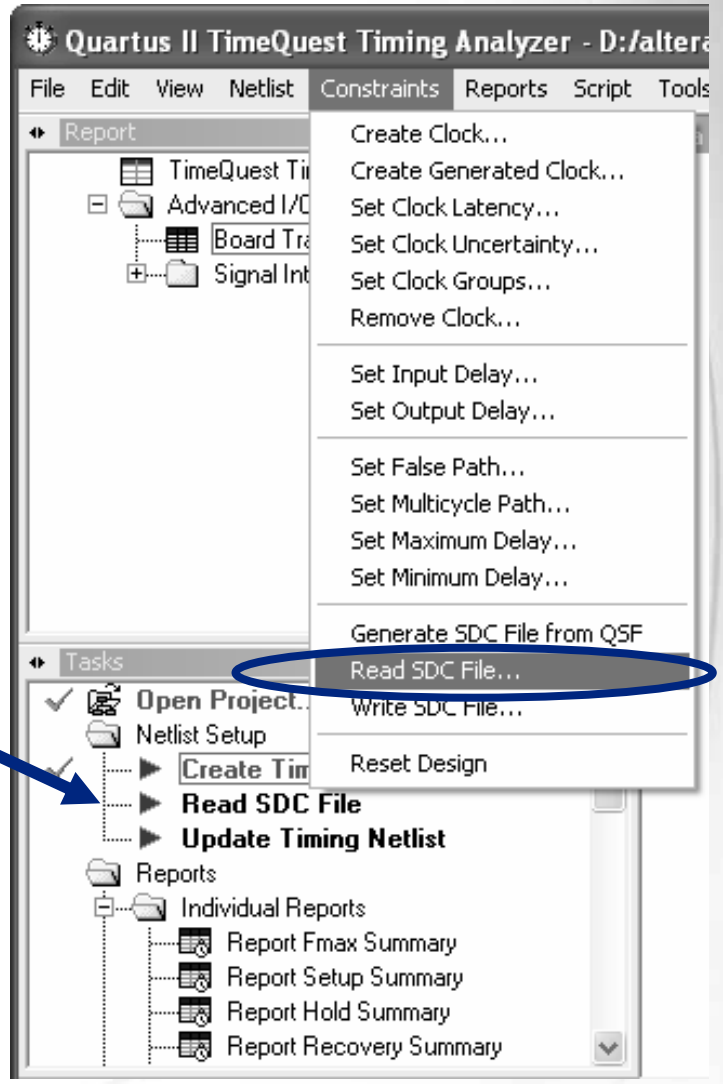
1) Generate Timing Netlist

- Create a timing netlist (i.e. database) based on compilation results
  - Post-synthesis (mapping) or post-fit (if design already fully compiled)
  - Worst-case (slow; max. temp.), best-case (fast; min. temp.) timing models
  - Set custom operating conditions
- To execute:



2) Create or Read in SDC File

- Create SDC file using SDC file editor
  - Don't enter constraints using **Constraints** menu
- Read in constraints & exceptions from existing SDC file
  - Skip if no SDC file
- Execution
  - **Read SDC File** (Tasks pane or **Constraints** menu)
- File precedence (if no filename specified)
  - Files specifically added to Quartus II project
  - <current_revision>.sdc (if it exists in project directory)

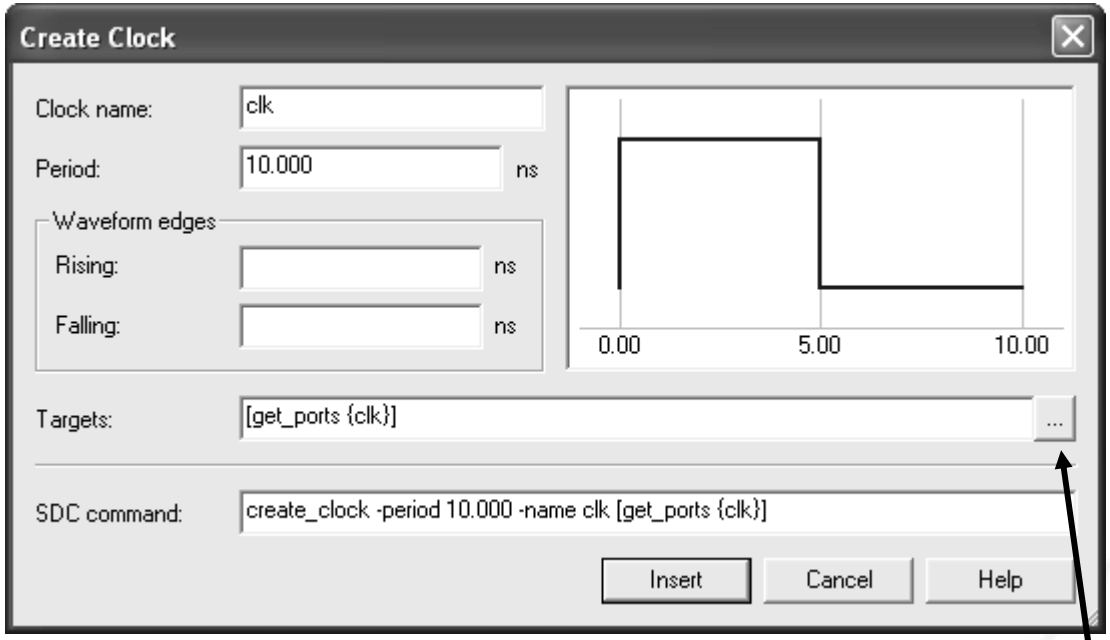


Tcl: read_sdc [<filename>]

Create Clock

In SDC File Editor,
Edit menu ⇒ Insert Constraint

Create Clock fields:
- **Clock Name – Assign name to clock setting; defaults to target node name**
- **Period – Clock period in nanoseconds**
- **Waveform edges – Use for non-50% duty cycle clocks**
- **Targets – Port or pin to which clock setting is being applied**



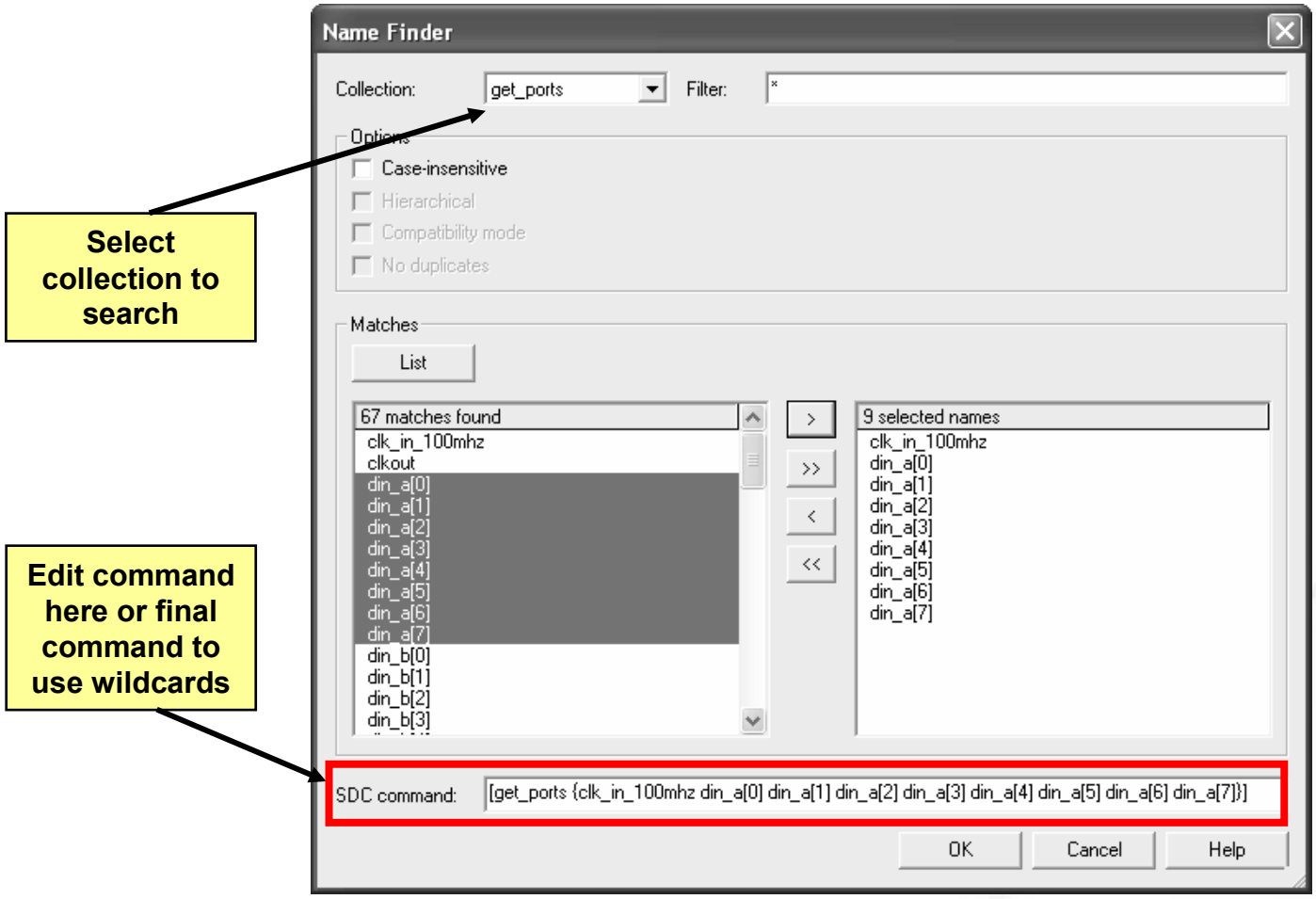
***Important Note*: All design clocks are related by default. This means the timing analyzer will analyze paths between clock domains whether you have specifically related them or not.**

Name Finder (next slide)

SDC: create_clock

Name Finder

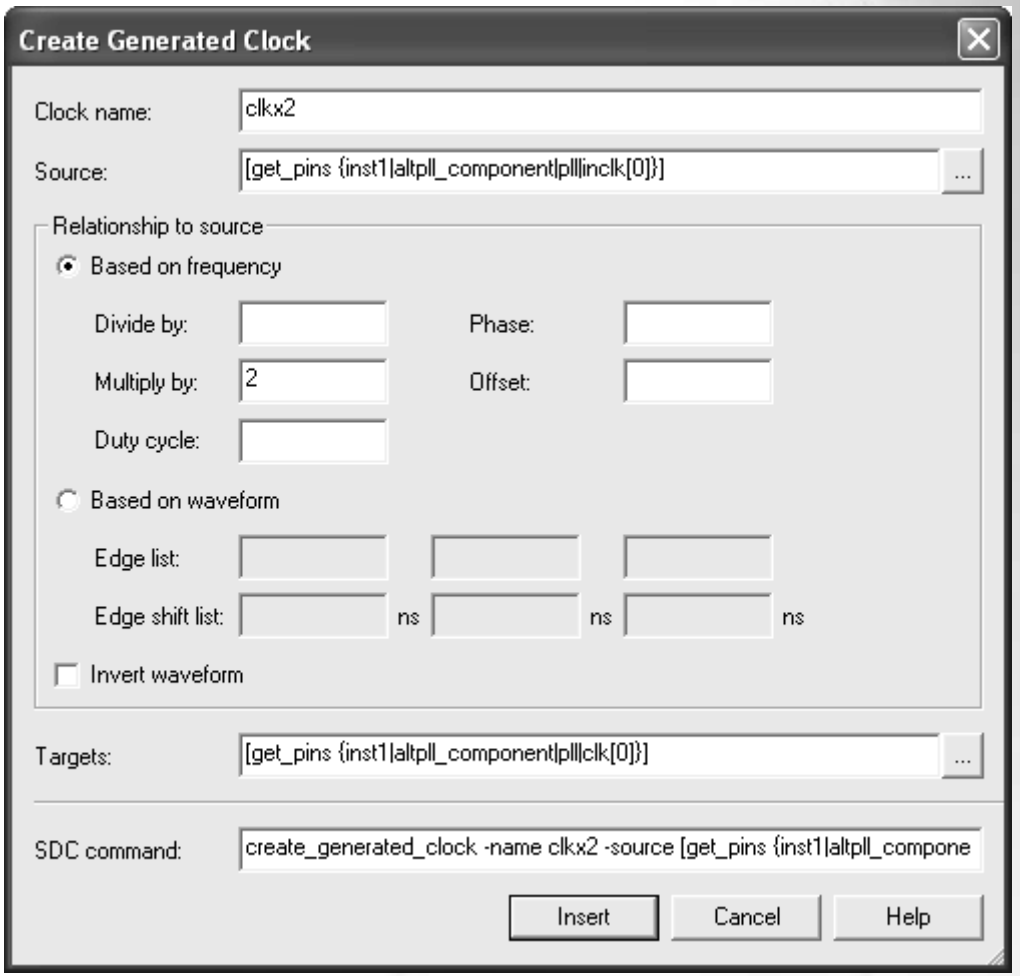
- Search the SDC netlist for node names
  - Similar to the Quartus II Node Finder



Generated Clocks

In SDC File Editor,
Edit menu ⇒ Insert Constraint

- Clock signals derived from a previously created clock
  - E.g. clock dividers, ripple clocks, PLLs
  - Must be defined by a constraint



Create Generated Clock fields:
- **Clock Name – Assign name to clock setting**
- **Relationship to source – Specify how generated clock is related to base clock. The Based on waveform section allows for more complexity in the relationship to the base clock (not discussed)**
- **Targets – Port or pin to which clock setting is being applied**

SDC: create_generated_clock

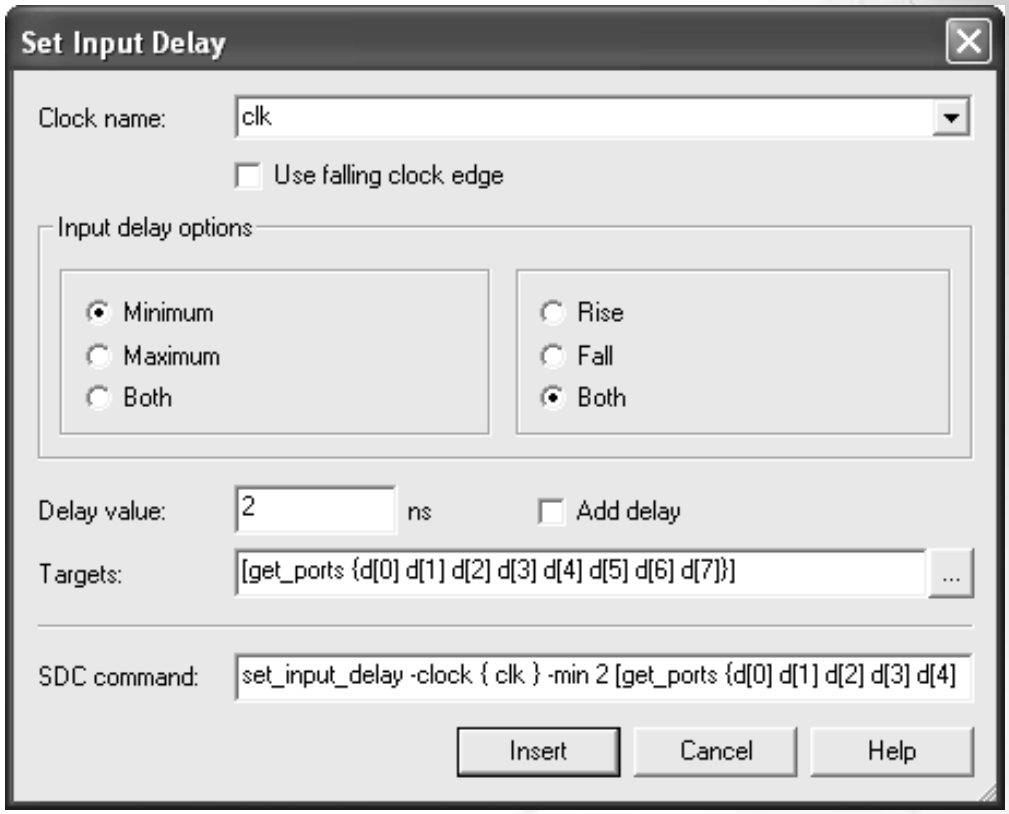
Synchronous I/O Constraining

- Specify system-level timing constraints
- Settings
  - Input/output maximum delay
    - **Maximum** amount of time a signal can take to arrive and still meet **setup** timing
  - Input/output minimum delay
    - **Minimum** amount of time a signal can remain active and still meet **hold** timing
- Pairs of max/min I/O delay constraints specify range of valid delay values for meeting FPGA and third-party device timing

See Appendix for details on I/O timing delays.

Set Input/Output Delay

In SDC File Editor,
Edit menu ⇒ Insert Constraint



Set Input/Output Delay fields:

- **Clock Name – Specify source clock**
- **Input/Output delay options – Choose max or min constraint. Rise/Fall indicates if the constraint applies particularly to a rising or falling edge transition (advanced).**
- **Delay value – Total off chip delay**
- **Add delay – Must use if applying multiple sets of input/output delays to the same port (e.g. input ports feeding multiple internal registers)**
- **Targets – Port to which setting is being applied**

SDC: set_input_delay
SDC: set_output_delay

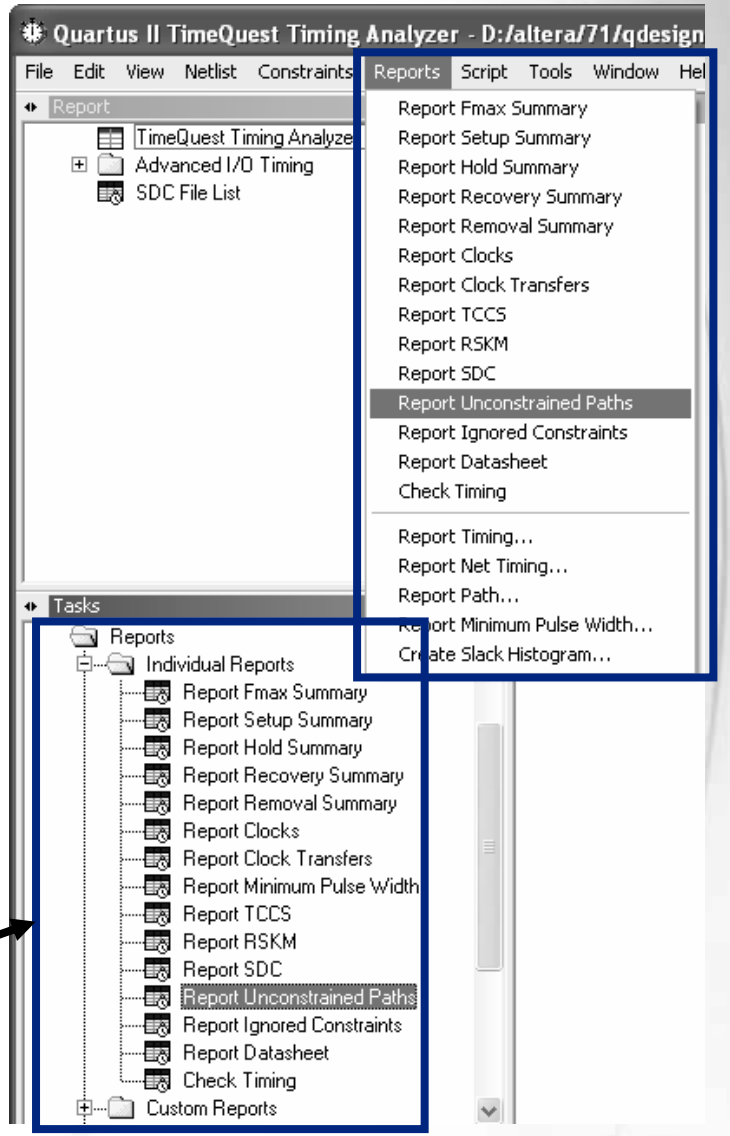
3) Update Timing Netlist

- Apply SDC constraints/exceptions to current timing netlist
- Generates warnings
  - Undefined clocks
  - Partially defined I/O delays
  - Combinatorial loops
- Update timing netlist after adding any new constraint
- Execution
  - **Update Timing Netlist** (Tasks pane or **Netlist** menu)

Tcl: update_timing_netlist

4) Generate Timing Reports

- Verify timing requirements and locate violations
- Check for fully constrained design or ignored timing constraints
- Two Methods
  - Tasks pane
    - Automatically creates/updates netlist & reads default SDC file if needed
  - **Reports** menu
    - Must have valid netlist to access



Double-click on individual report

Useful Reports for Design Constraining

- Report Clocks
  - Use to ensure all clocks have been defined correctly
- Report Unconstrained Paths
  - Use to determine if any constraints are missing
- Report SDC
  - Use to review what constraints have currently been applied to the netlist
- Report Ignored Constraints
  - Use to determine if any constraints being ignored due to possible typos or other errors in constraints

Generating Detailed Reports