The Abdus Salam
International Centre for Theoretical Physics

United Nations
Educational, Scientific and
Cultural Organization

IAEA
International Atomic Energy Agency

2065-21

Advanced Training Course on FPGA Design and VHDL for Hardware Simulation and Synthesis

26 October - 20 November, 2009

Liberto TM Integrated Design Environment

Nizar Abdallah
ACTEL Corp. 2061 Stierlin Court Mountain View
CA 94043-4655
U.S.A.

Strada Costiera 11, 34151 Trieste, Italy - Tel.+39 040 2240 111; Fax +39 040 224 163 - sci_info@ictp.it

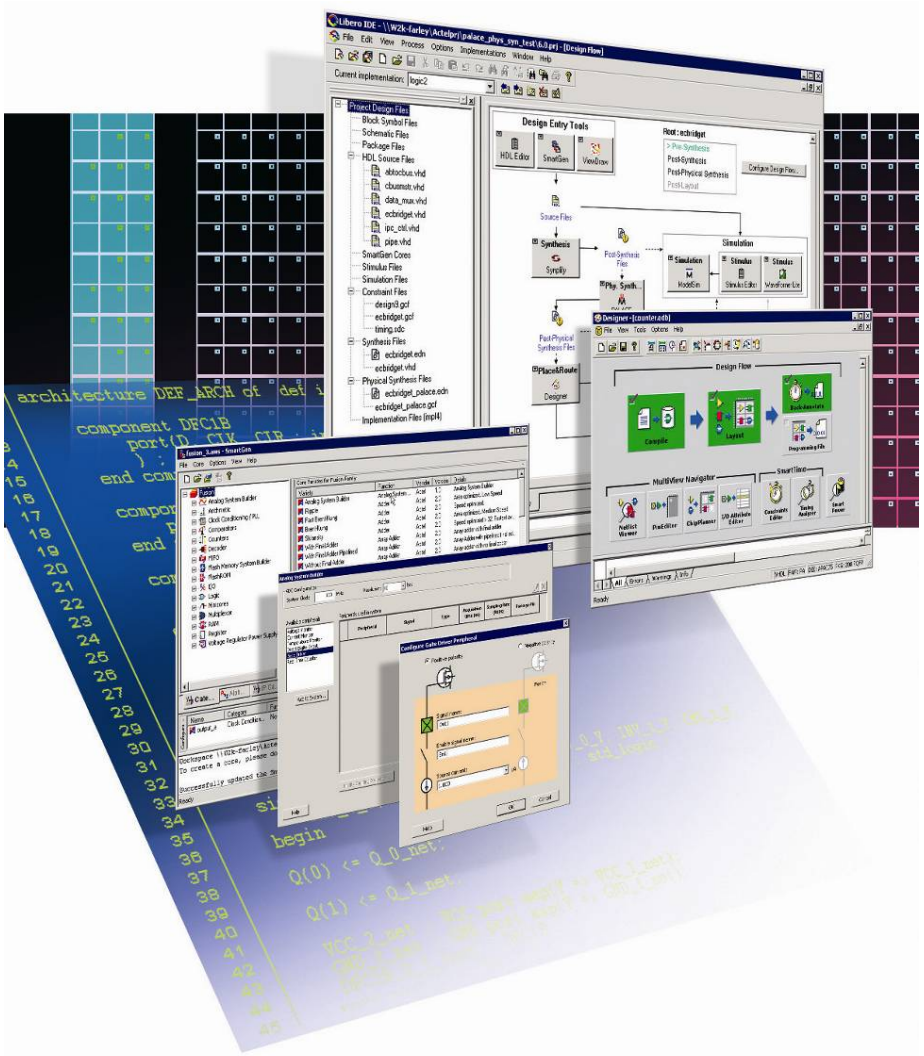
Libero™ Integrated Design Environment

Agenda

- Libero Overview
- 3rd Party Tools
- Designer Overview
- Programming
- Silicon Explorer
- Reference Material

Libero IDE:

A Complete FPGA Design Solution



- **Comprehensive Development Environment**
  - Design, implementation, validation, debug
  - Supports full suite of Actel's power efficient FPGAs
- **Best in Industry Tools**
  - Industry leading synthesis and simulation tools
  - High performance timing and power driven place & route
  - SmartTime timing and bottleneck analysis speeds timing closure
  - Accurate hierarchical power analysis with SmartPower
- **Relentless Pursuit of Ease of Design**
  - Push button flows hide design complexity
  - GUI wizards guide design process
  - Rich IP library & user block support facilitates design reuse

Libero IDE Gold Edition

World Class Fully-Integrated PC Development System

- Targets All Devices up to 1.5M Gates
- Tools:
  - ViewDraw AE v8.5
  - WaveFormer Lite 12.30a
  - ModelSim AE 6.4a
  - Synplify AE 9.4A (1.5M gates) SYNPLIFY-PRO
  - Actel Designer 8.5(1.5M gates)
  - FlashPro 8.5
  - Silicon Explorer 5.2
- PC Platform
  - Licensed to Disk ID (Node-Locked license)
- 1 Year Free License!
  - No Updates (Service Packs Available)

Tool Features

Integrated Design Management	✔
Schematic	✔
VHDL	✔
Verilog	✔
Macro Generation	✔
Testbench Generator	✔
VHDL Simulation Verilog Simulation	✔
Timing Simulation	✔
Static Timing Analysis	✔
Timing Driven Place and Route	✔
Push Button Place and Route	✔
Layout Editor	✔
Silicon Explorer	✔
Standard Industry Interfaces	✔
Third Party Design Libraries	✔

WinXP Pro SP2	✔
Win Vista Business	✔

Libero IDE Platinum Edition

Windows Support

World Class Fully Integrated PC Development System

- Targets *All* Actel Devices
- Tools
  - ViewDraw AE v8.5
  - WFL 12.30a with Reactive Testbench Generation
  - ModelSim AE 6.4a
  - Synplify Pro AE 9.4A (*No* Gate Limit)
  - Actel Designer 8.5 (*No* Gate Limit)
  - FlashPro 8.5
  - Silicon Explorer 5.2
- License Types
  - Node-Locked License Tied to Disk-id or Hardware Keys
  - Floating License
- Price:
  - $2495 First Year / $1995 Renewal
  - Free 45 day Evaluation License

Tool Features

Integrated Design Management	✔
Schematic	✔
VHDL	✔
Verilog	✔
Macro Generation	✔
Testbench Generator	✔
VHDL Simulation Verilog Simulation	✔
Timing Simulation	✔
Static Timing Analysis	✔
Timing Driven Place and Route	✔
Push Button Place and Route	✔
Layout Editor	✔
Silicon Explorer	✔
Standard Industry Interfaces	✔
Third Party Design Libraries	✔

WinXP Pro SP2	✔
Win Vista Business	✔

Libero IDE

Solaris and Linux Support

- Libero Solaris, Libero Linux
  - Tools
    - Project Manager
    - Synplify Pro AE
    - ModelSim AE VHDL & Verilog
    - Designer
  - Supports All Devices
  - License Type: Floating
    - Solaris: Host ID only
    - Linux: requires USB key for Synplify
  - 1 Year License: $2495
    - Renewal: $1995
  - Free 45 day Evaluation License

Libero IDE

Edition Summary

	Libero Gold	Libero Platinum (Windows)	Libero Platinum (Solaris / Linux)	Libero Platinum Eval
Device Support	All devices up to 1.5M Gates	All Devices	All Devices	All Devices
ViewDraw	AE	AE	-	AE
WaveFormer Lite	AE	Reactive Testbench	-	Reactive Testbench
Synthesis	Synplify AE	Synplify Pro AE	Synplify Pro AE	Synplify Pro AE
ModelSim	AE	AE	AE	AE
Designer	Yes	Yes	Yes	Yes
FlashPro	Yes	Yes	-	No
Explorer	Yes	Yes	-	No
License Type	Disk ID	Key or Disk ID	Floating	Disk ID (Windows) Floating (Solaris / Linux)
Price	Free	$2,495	$2,495	Free 45 day

Libero IDE v8.5

Device Support by Edition

Family	Gold	Platinum and Eval
ACT1	1010B, 1020B, 10V10B, 10V20B	
ACT2 / ACT3	1225A, 1240A, 1280A, 415A, 1425A, 1440A, 1460A, 14100A	
1200XL	1225XL, 1240XL, 1280XL	
DX	3265DX, 32100DX, 32140DX, 32200DX, 32300DX	
MX	40MX02, 40MX04, 42MX09, 42MX16, 42MX24, 42MX36	
RH	RH1020, RH1280	
RT	RT1020, RT1425A, RT1280A, RT1460A, RT14100A, RTSX32SU, RT54SX72SU, RTAX250S, RTAX1000S	RTAX2000S, RTAX4000S
RTAX-DSP		RTAX2000D, RTAX4000D
SX / SX-A	SX08, SX16, SX16P, SX32, SX08A, SX16A, SX32A, SX72A	
eX	eX64, eX128, eX256	
Axcelerator	AX125, AX250, AX500, AX1000	AX2000
ProASIC^{PLUS}	APA075, APA150, APA300, APA450, APA600, APA750, APA1000	
ProASIC3	A3P015, A3P030, A3P060, A3P125, A3P250, A3P400, A3P600, A3P1000, A3PE600, A3PE1500	A3PE3000
M7 / M1 ProASIC3	M7A3P1000, M1A3P250, M1A3P600, M1A3P1000, M1A3PE1500	M1A3PE3000
ProASIC3L	A3P250L, A3P600L, A3P1000L	A3PE3000L
ProASIC3 nano	A3PN030	
IGLOO (1.2V / 1.5V)	AGL015, AGL030, AGL060, AGL125, AGL250, AGL600, AGL1000, AGLE600	AGLE3000
IGLOO PLUS	AGLP030, AGLP060, AGLP125	
M1 IGLOO (1.2V / 1.5 V)	M1AGL250, M1AGL600, M1AGL1000	
IGLOO nano (1.2V / 1.5V)	AGLN030	
Fusion	AFS090, AFS250, AFS600, AFS1500	
M7 / M1 Fusion	M7AFS600, M1AFS250, M1AFS600, M1AFS1500	

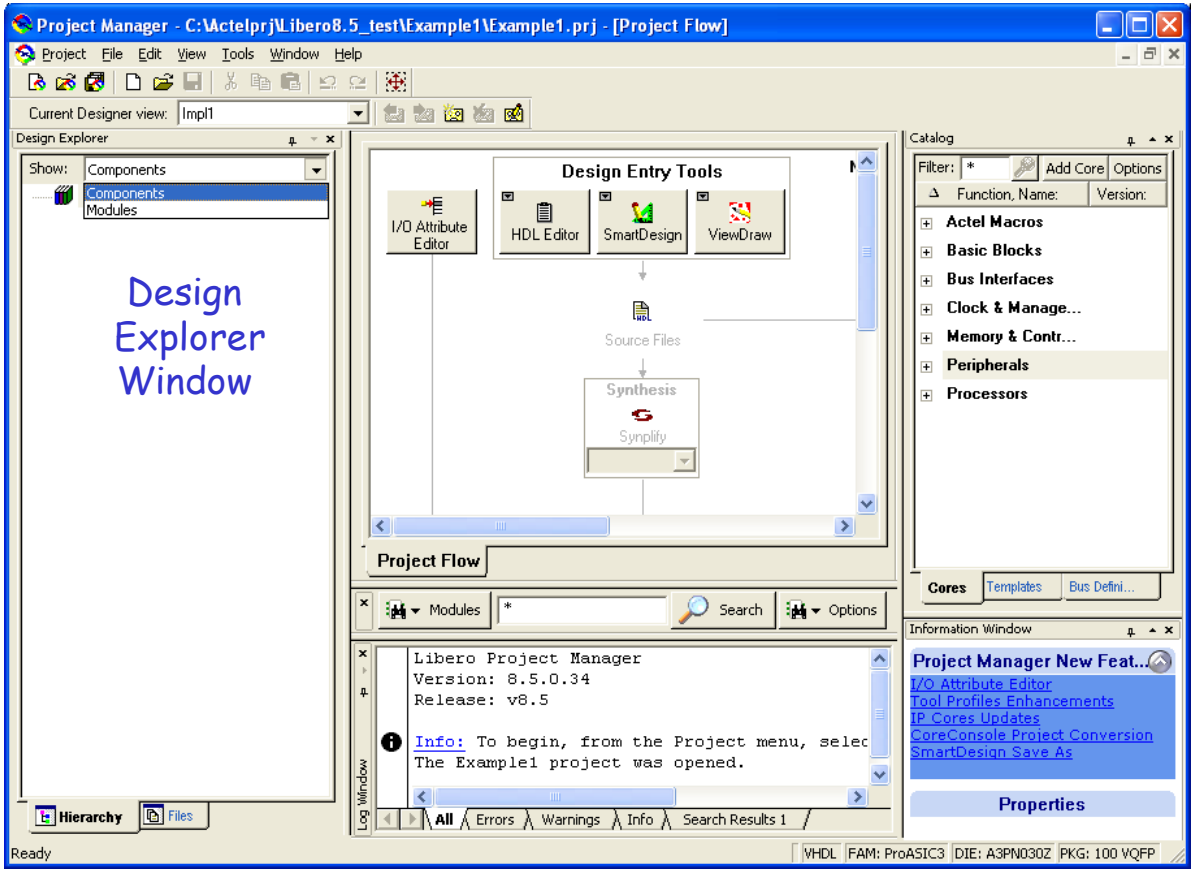
Libero IDE Installation

Libero IDE Project Manager

Libero IDE Design Explorer Window

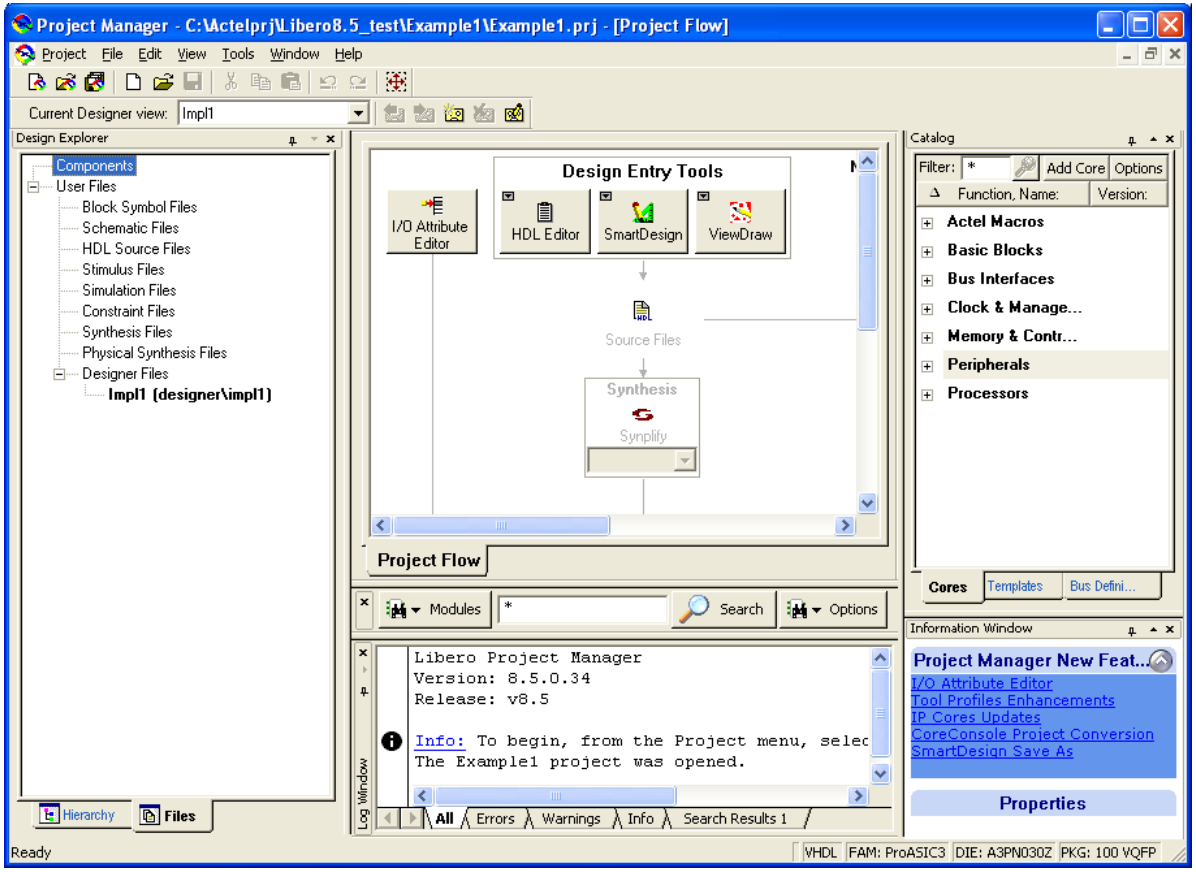
- Hierarchy Tab Displays Hierarchical Representation of Source Files and Components in Project
  - Libero Automatically Updates Design Hierarchy

Display Modules or Components



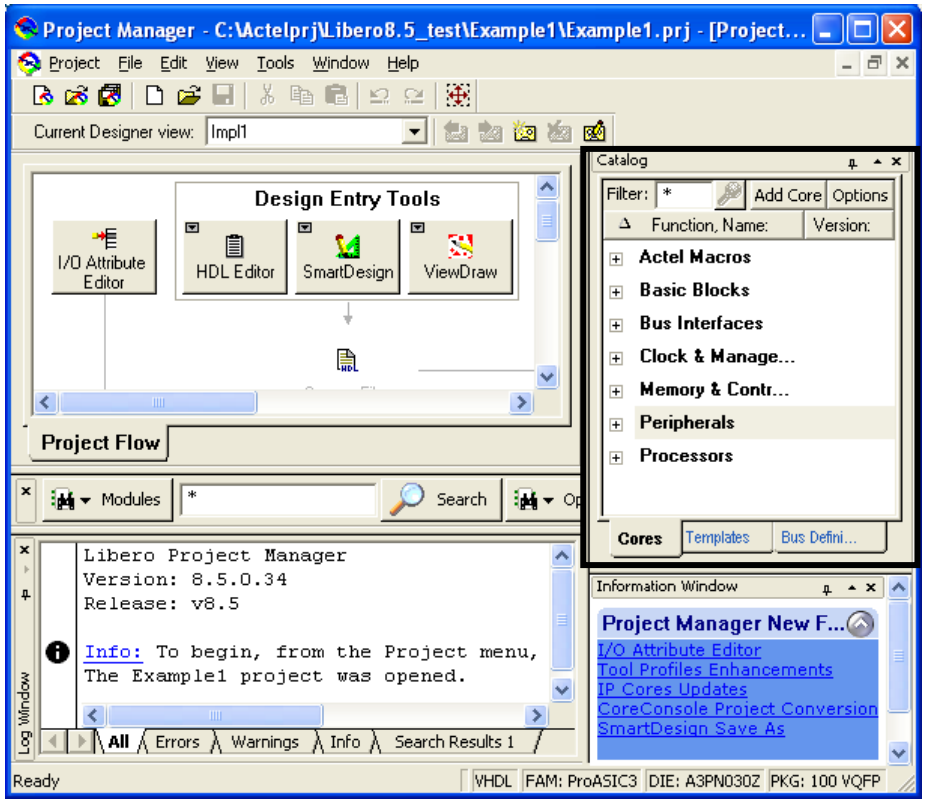
Libero IDE Design Explorer Window

- Files Tab Displays All Files in Project Grouped by Type
  - Components and user created files are displayed separately
  - Sort components by name or by type



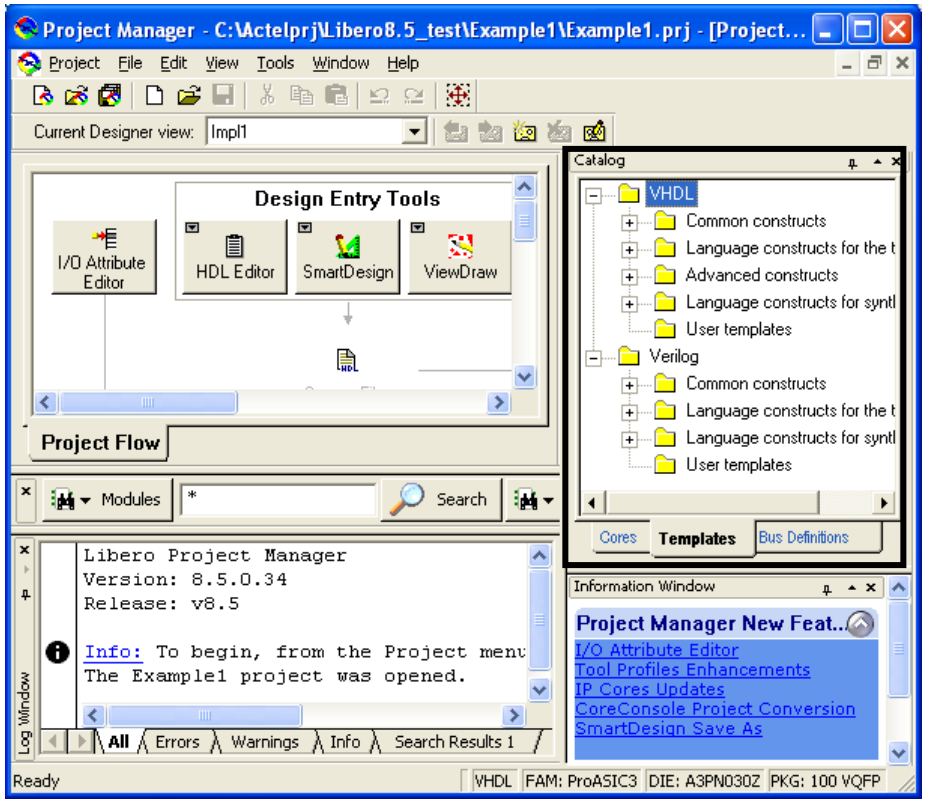
Libero IDE Catalog Window: *IP Cores Tab*

- IP Based Design Flow
  - Access to a variety of IP Cores
- Configure and Instantiate Directly from Libero
  - No need to launch separate tools
  - No need to re-enter common information
  - No need to import generated cores
- Modify Generated Cores Directly from Libero
  - Change Configurations
  - Change Versions



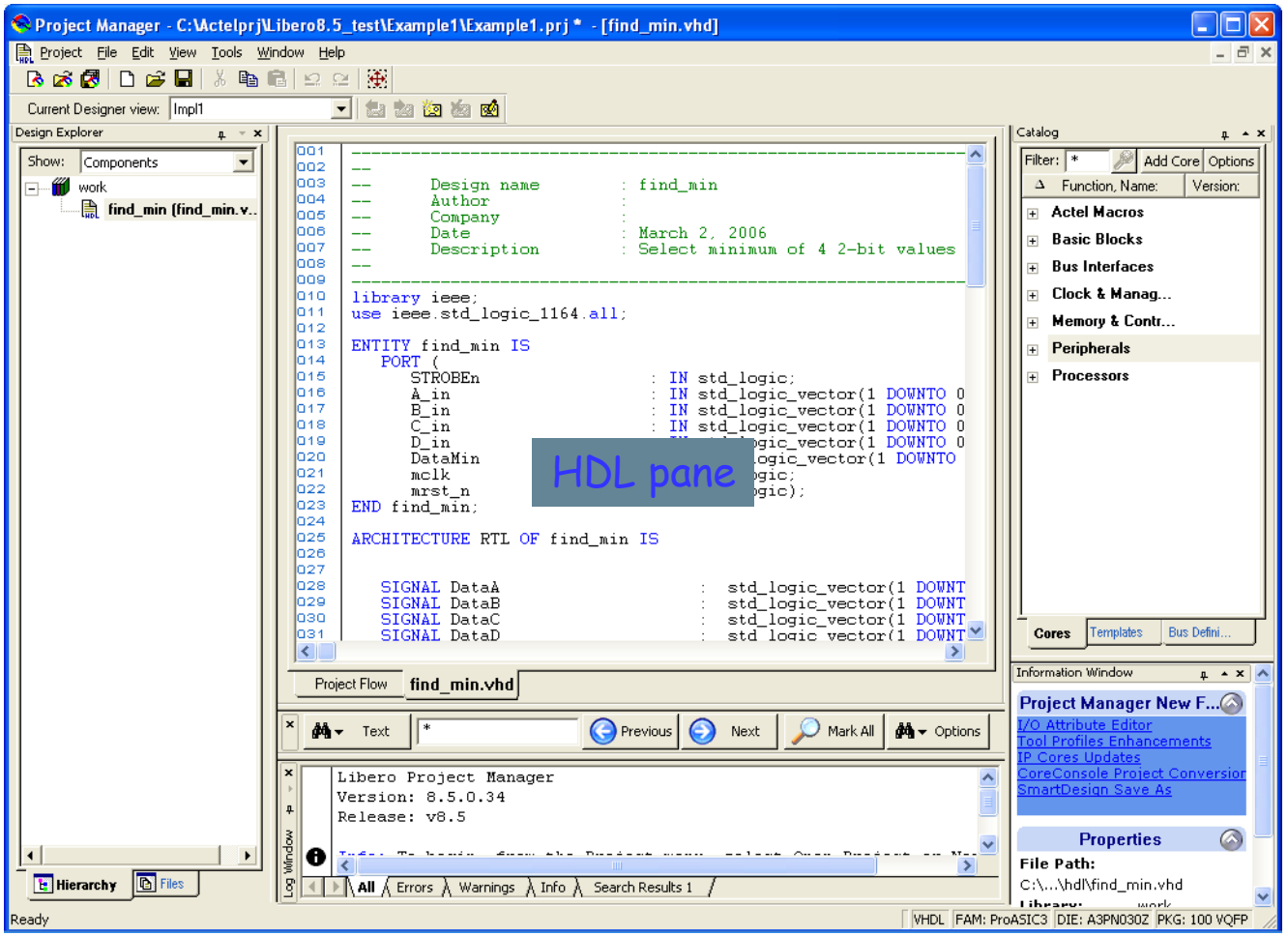
Libero IDE Catalog Window: *HDL Templates Tab*

- VHDL and Verilog Templates
  - Facilitate the Writing of HDL Files by Inserting Predefined Language Constructs
  - Create User-defined Template Files



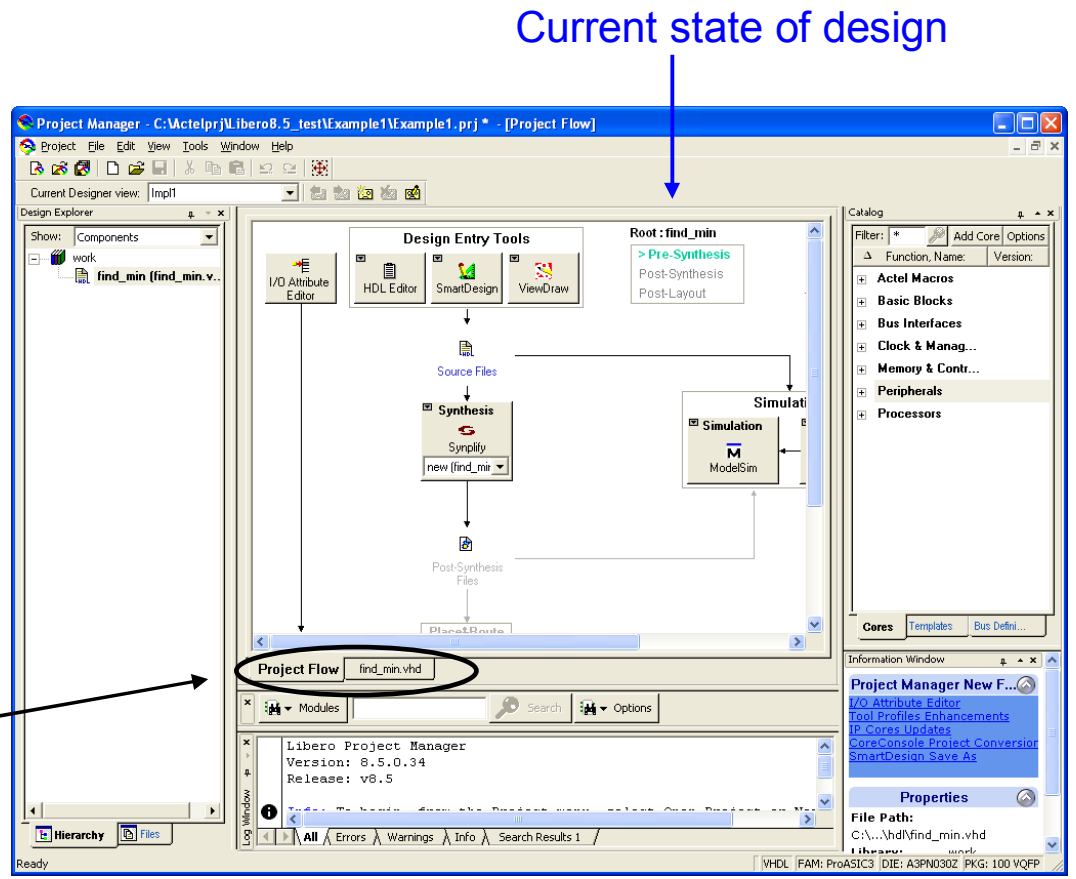
Libero IDE Project Manager: *HDL Editor*

- Language-Sensitive HDL Editor
  - Verilog 2001 or VHDL 93



Libero IDE Project Manager: *Project Flow Window*

- Tools Can Be Launched from Project Flow Window or Menus
- Step-by-Step Design flow Decreases Development Time
- Project Flow Window Displays:
  - Tools
  - Files
  - Transitions
  - Current State
  - Tool Tips
- Display Changes Dynamically Based on Target Family
- Tabs Switch Between Flow Window and HDL Window



Libero IDE Project Flow Window: *File Status*

- Group of Files Can Be …
- … Missing
  - If ANY Are Missing, Block Is Shadowed Out
- … Available and Current
  - Green Check Mark Is Shown
- … Available, but Not Current
  - If at Least One is Not Current, Warning Icon Is Displayed

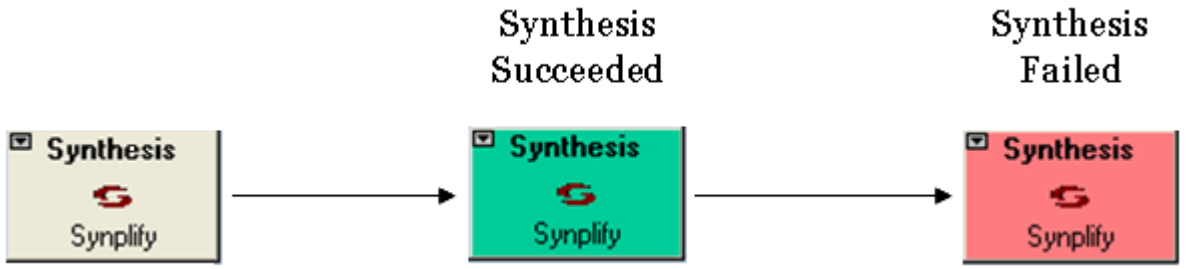
Synthesis Incomplete

Synthesis Complete

Source Modified – EDIF Netlist not current

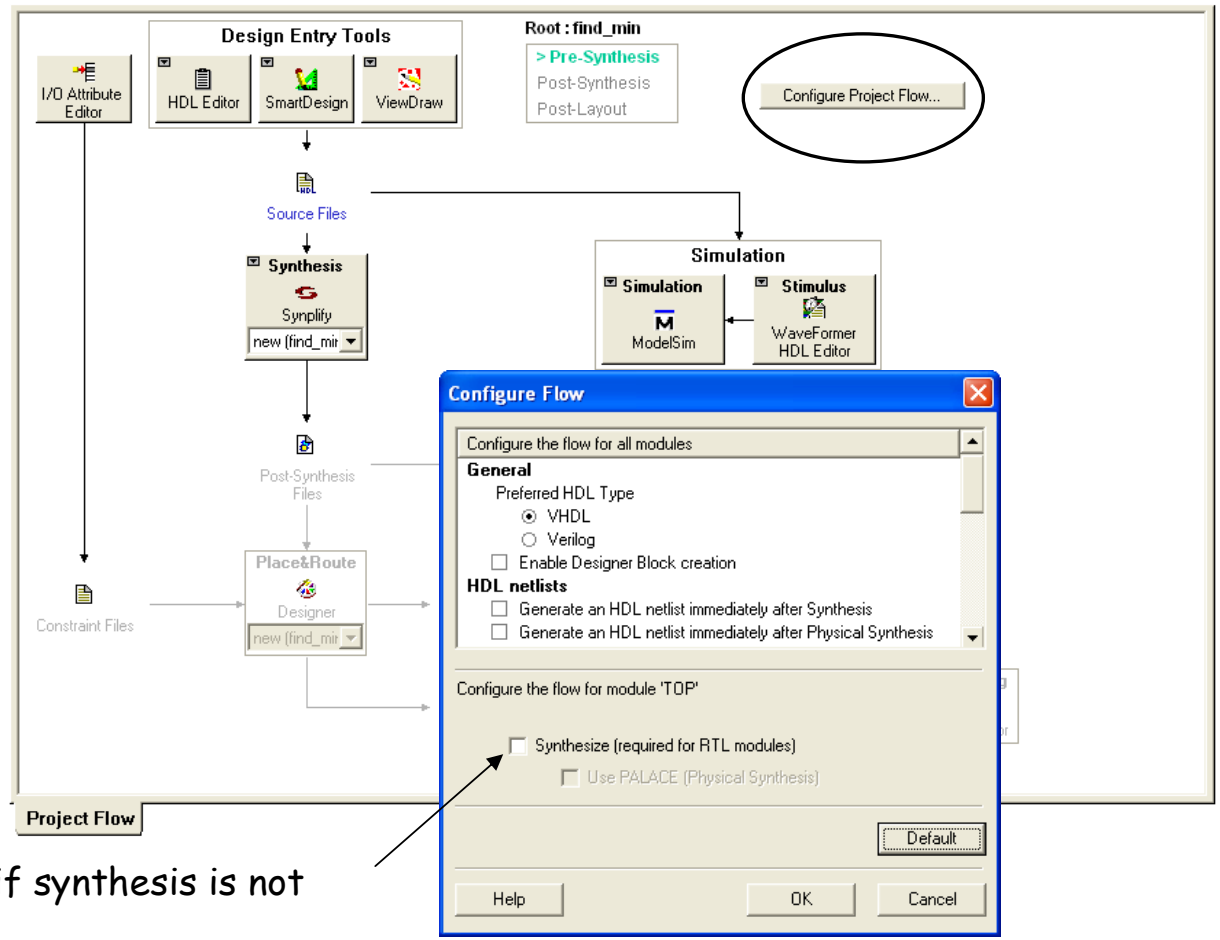
Libero IDE Project Flow Window: *Tool States*

- Disabled => Button Is Shadowed
- White => Available, but Not Yet Used
- Green => Completed Successfully
- Red => Error in Running Tool



Libero IDE Project Flow Window: *Config Options*

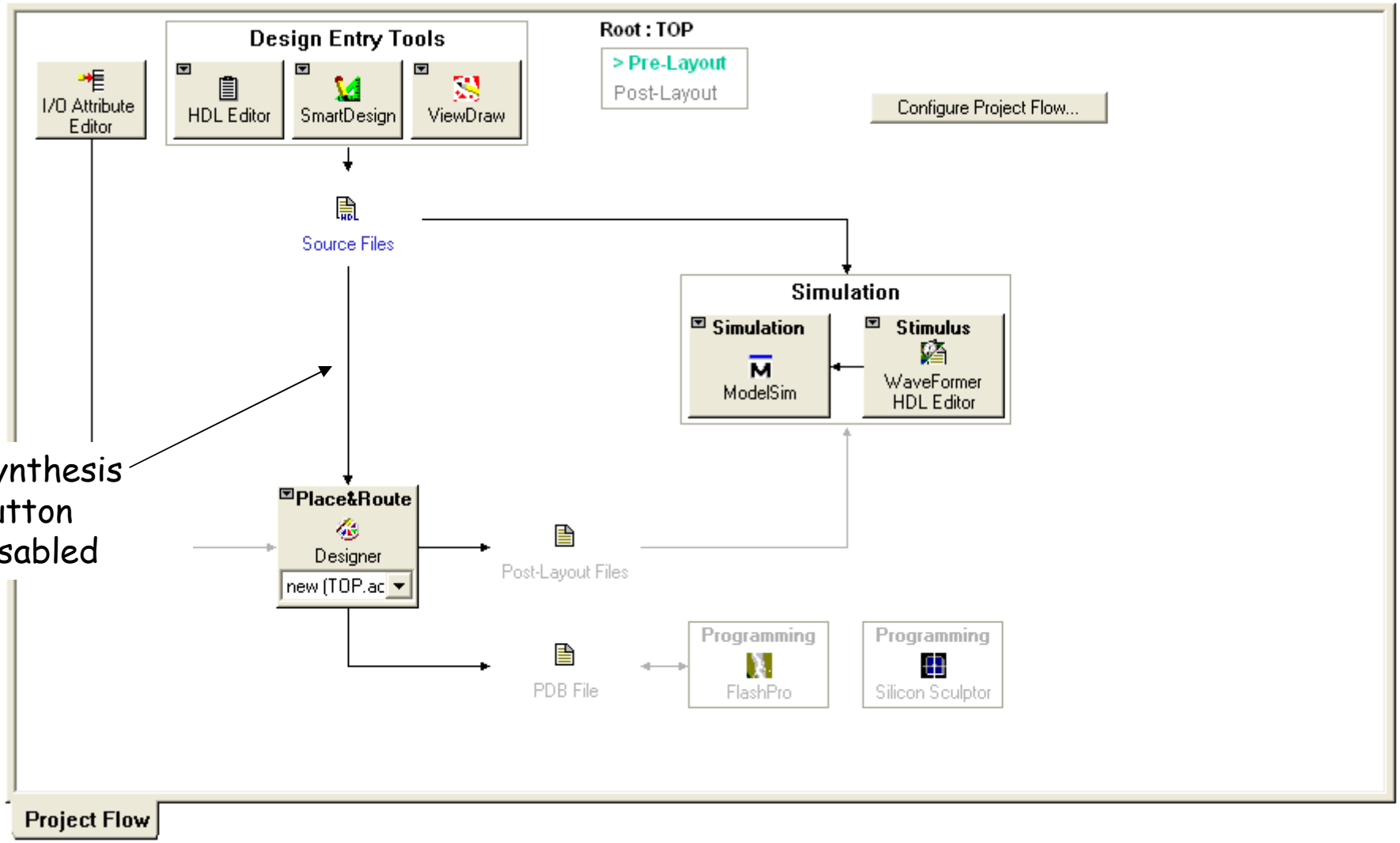
- Project Flow Window Can be Configured for Designs that do not Require Synthesis



Un-check if synthesis is not required

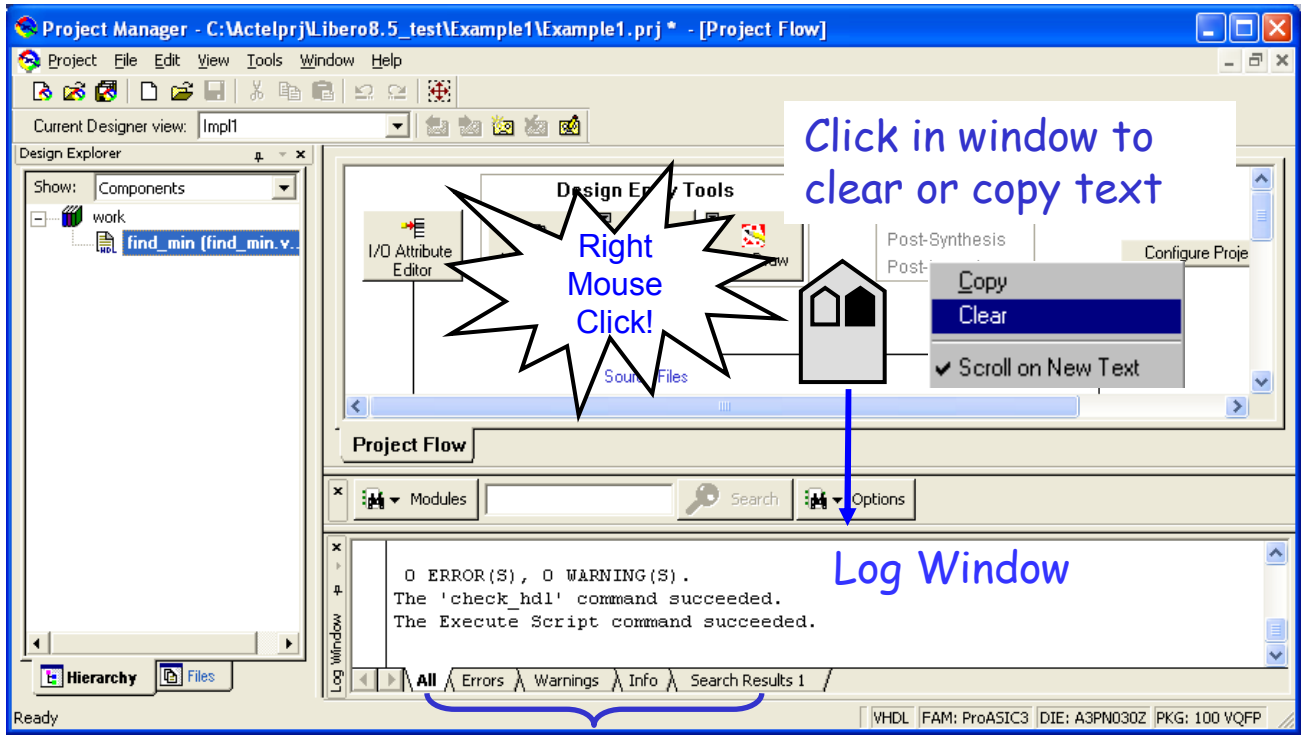
Libero IDE Project Flow Window

Synthesis Not Required



Libero IDE Log Window Error Manager

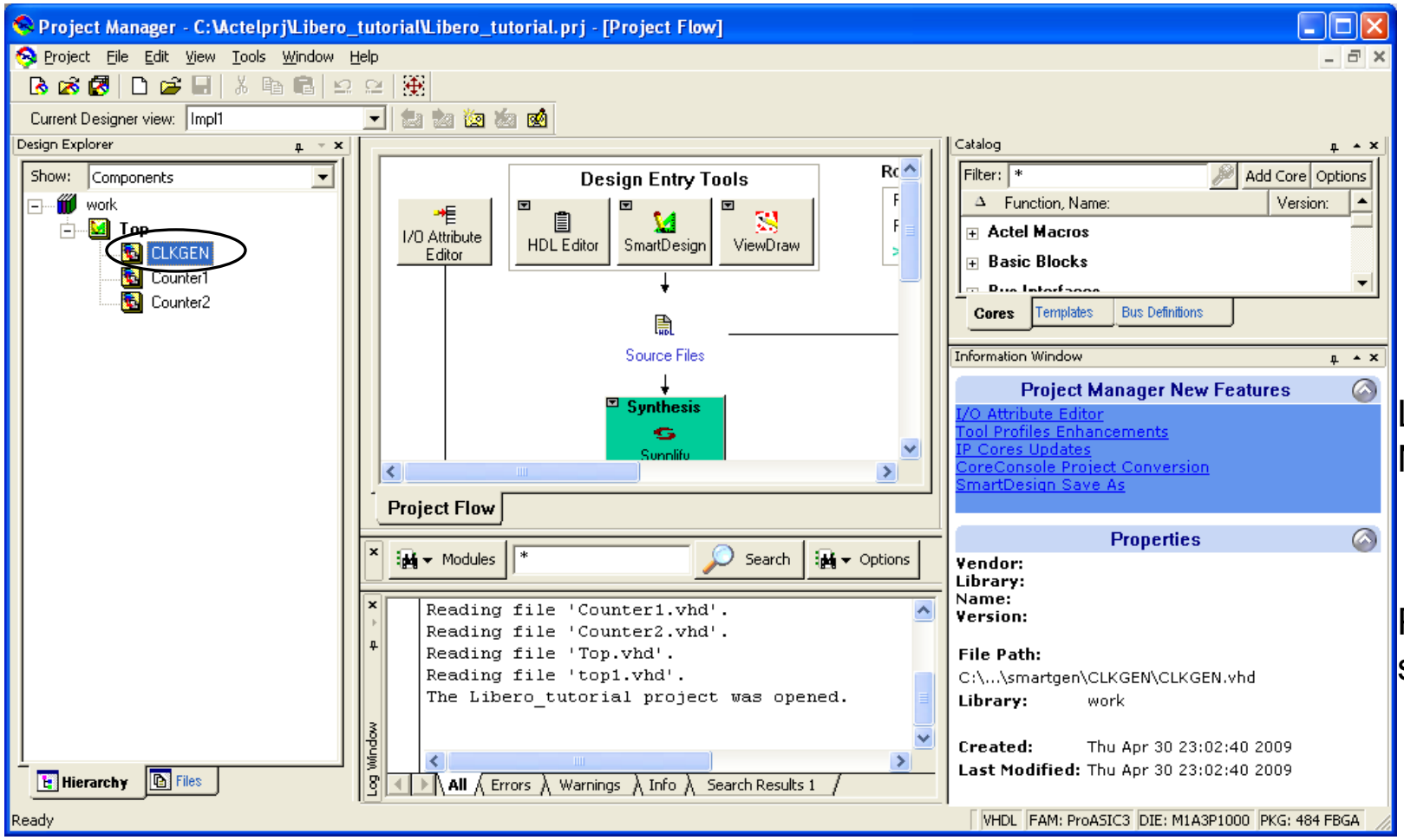
- Error Manager Consists of Tabs in Log Window:
  - All: Displays All Messages
  - Errors: Displays Error Messages
  - Warnings: Displays Warning Messages
  - Info: Displays Information Messages
- Default Error Message Colors can be Changed



Libero error manager tabs

Libero IDE Information Window

- Displays New Features in Libero and File Properties



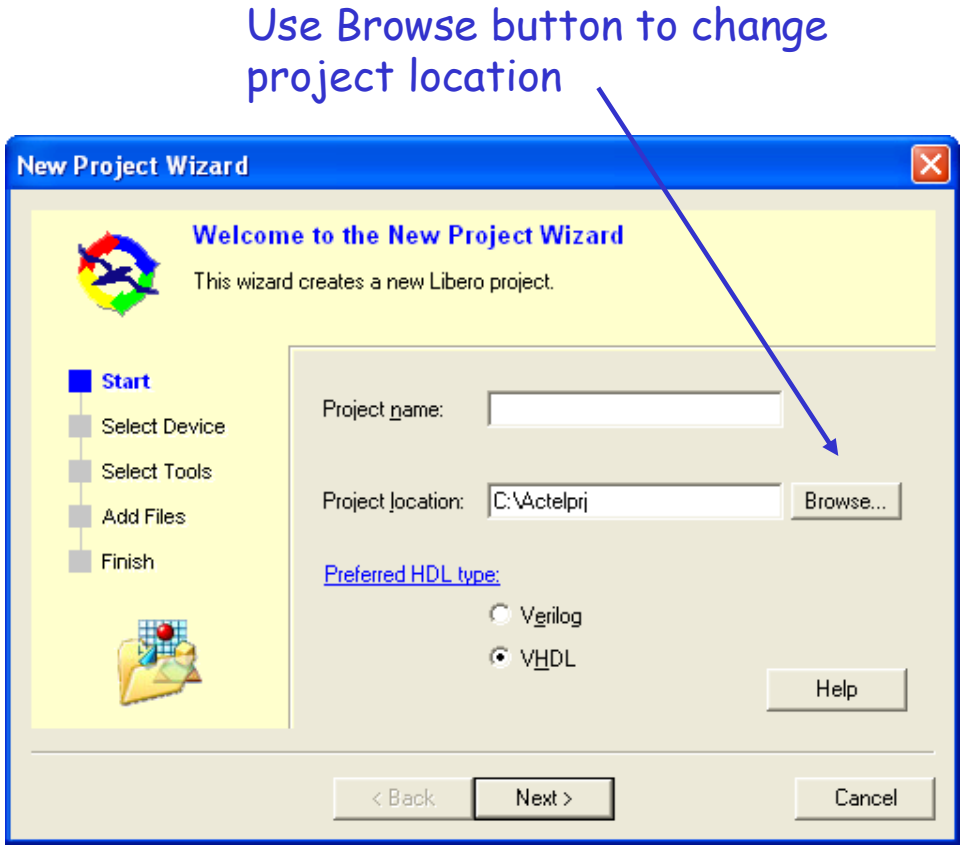
Libero 8.5 New Features

Properties for selected file

Creating a Libero Project

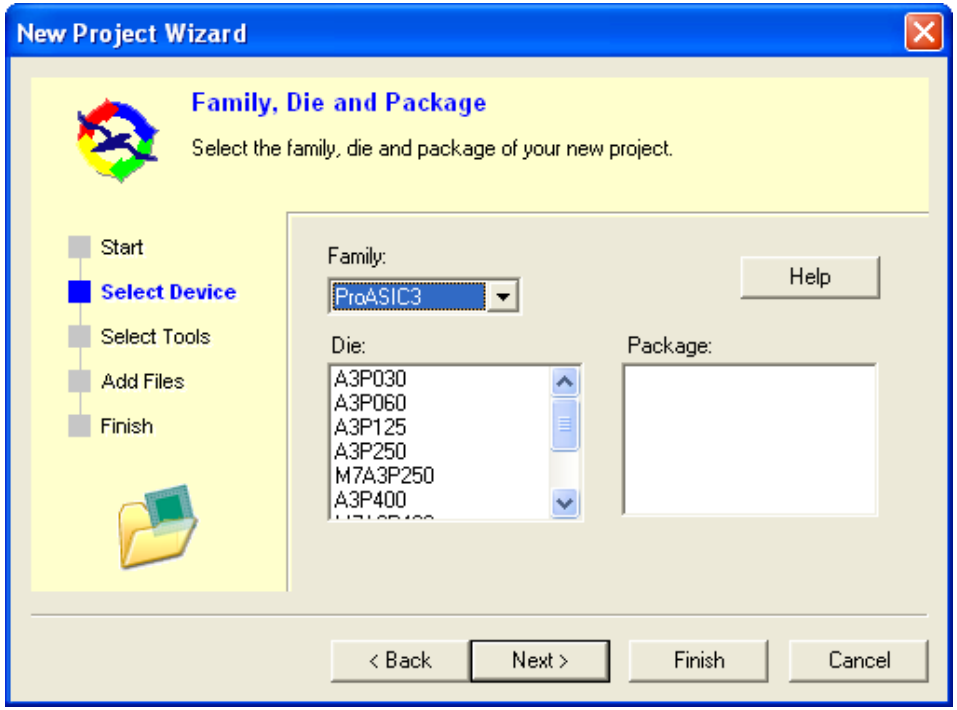
Libero IDE New Project Wizard

- Menu-Driven Wizard
  - Project > New Project
- Status Guide Shows Current State
  - All Fields Must Be Filled in to Continue
- HDL Type Must Be Consistent with License
- Next Button Goes to Next Wizard Screen
- Finish Button Finishes/Closes Wizard after Making Changes and Saves All Selections



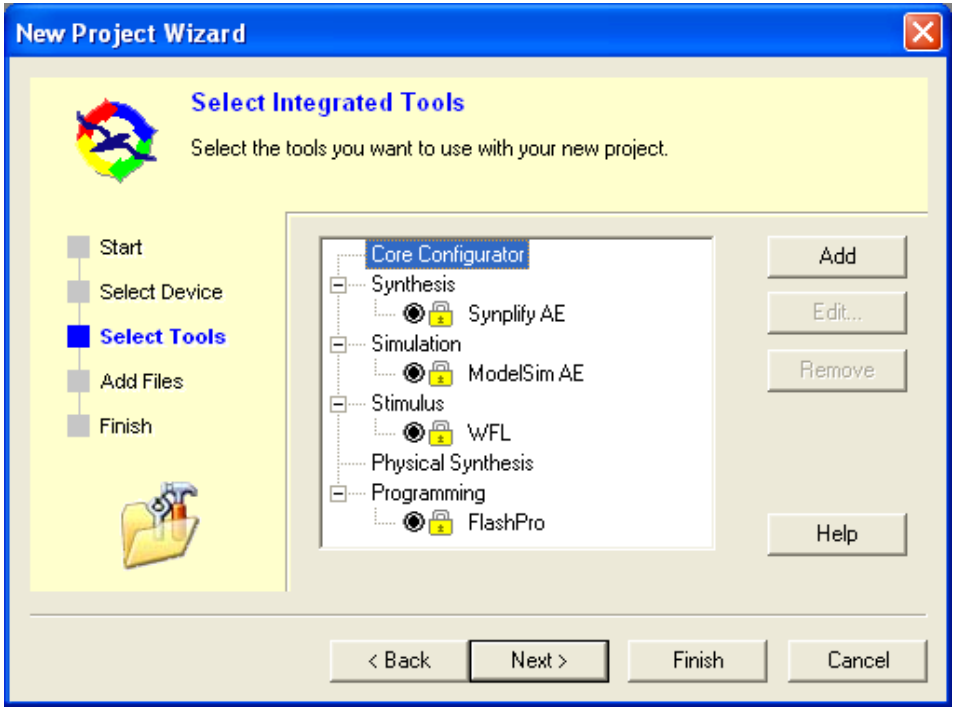
Libero IDE New Project Wizard
Select Device

- Select Family
- After Family Is Selected, Devices from that Family Are Displayed
- After Device Is Selected, Available Packages for Device Are Displayed



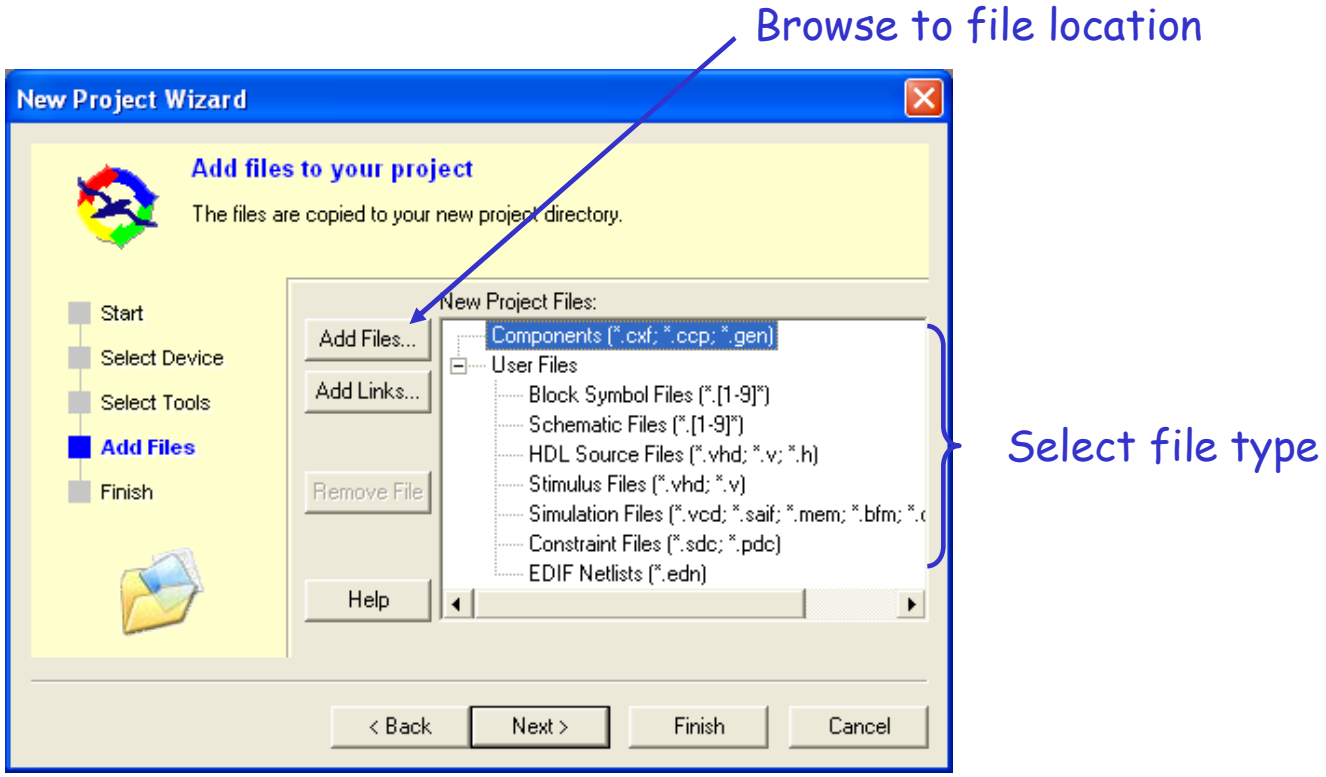
Libero IDE New Project Wizard
Select Tools

- Synthesis
  - Vendor (e.g. Synplify)
    - Version
- Stimulus
  - WaveFormer Lite
    - Select Version
- Simulation
  - ModelSim
    - Select Version
- Support for Mentor Graphic's Leonardo Spectrum and Precision
  - Standard Tools Direct from Mentor
    - No Actel OEM Versions



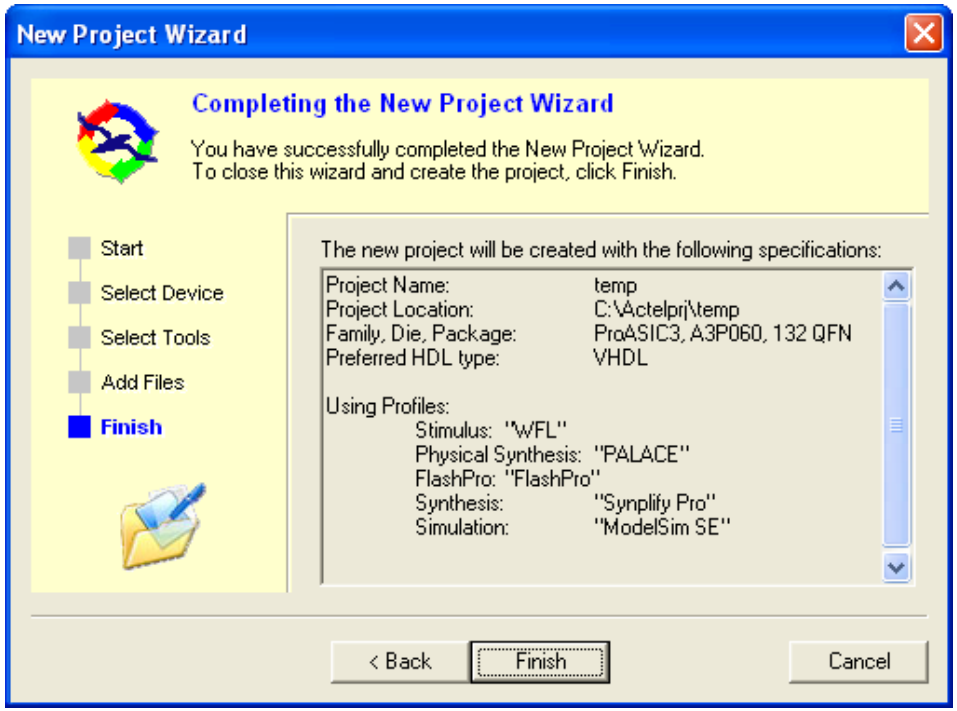
Libero IDE New Project Wizard
Add Files

- Add Existing Design Files to Project or Create a Link
  - Schematics, Symbols, HDL (VHDL or Verilog), Stimulus (VHDL or Verilog), SmartGen Macros or EDIF Netlists



Libero IDE New Project Wizard
Finish

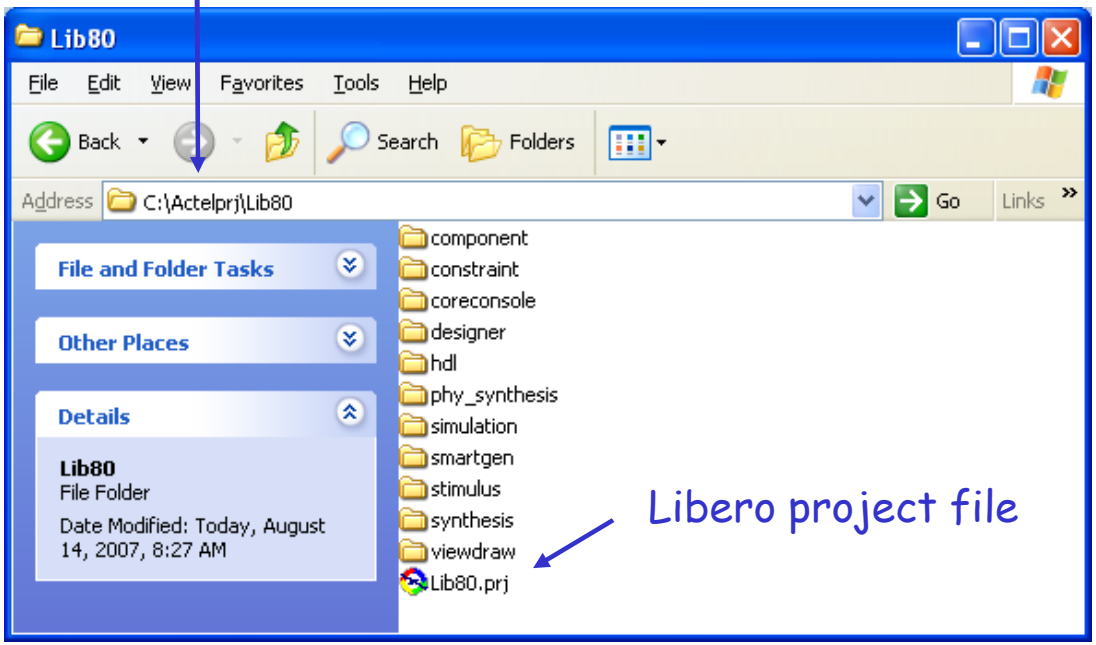
- Project Information Listed in Dialog Box
  - Click “Finish” to Complete Project Creation or “Back” to Make Corrections or Additions

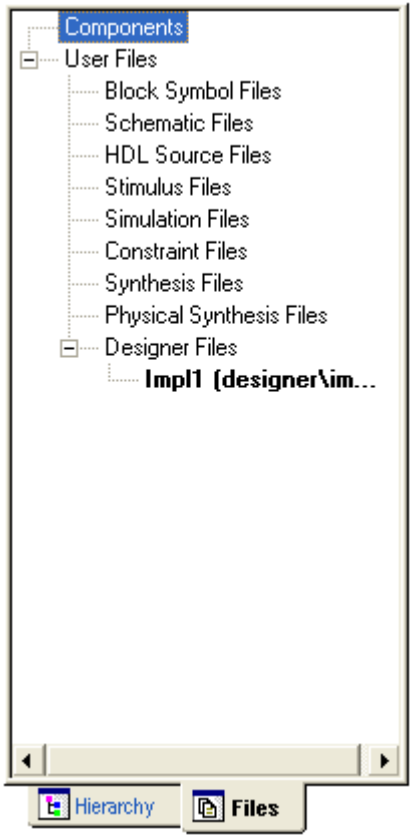


Project summary shown in window

Organization of Libero IDE Project Files

Actelprj is the Default folder for Libero projects



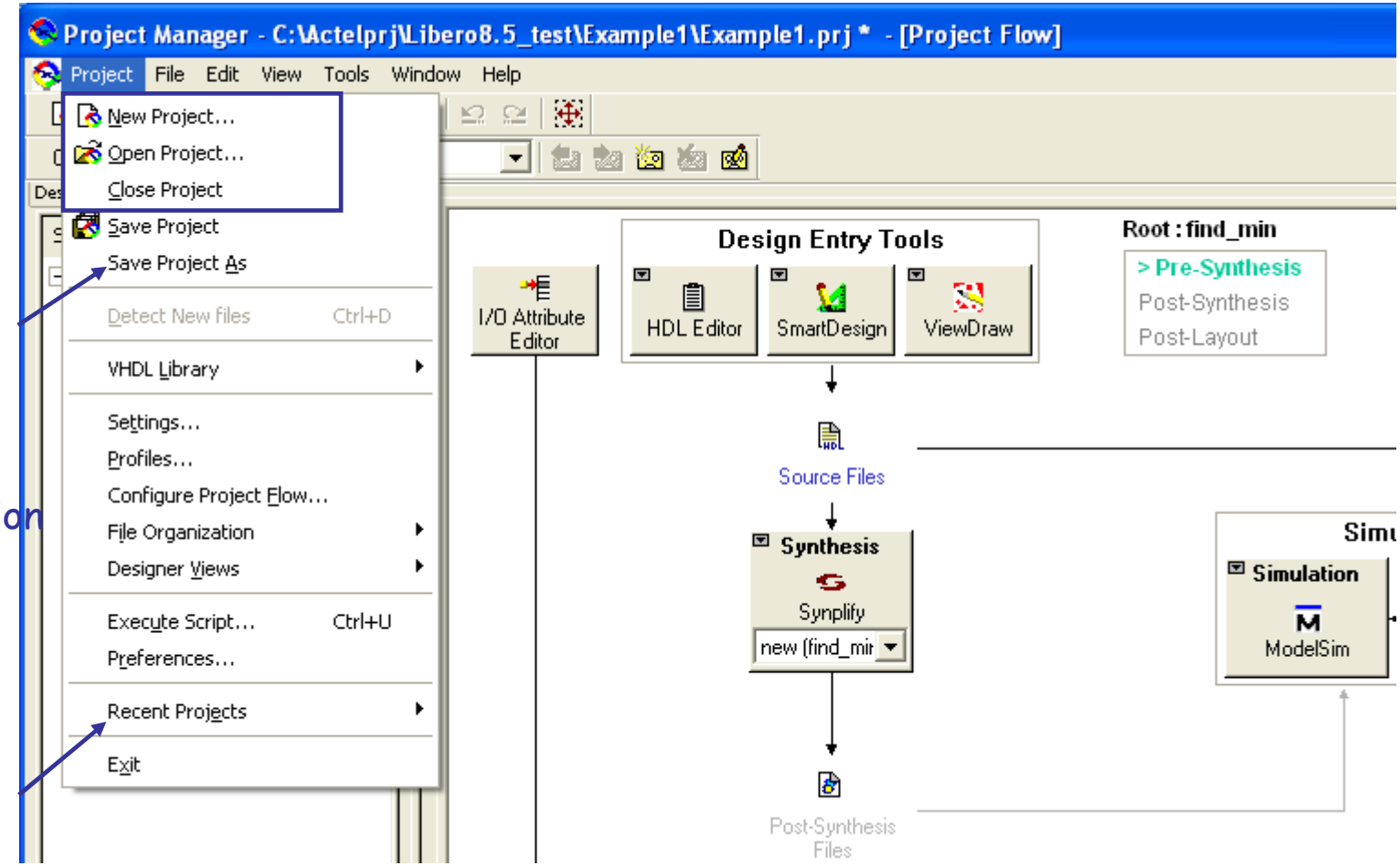


Libero IDE Project Menu Options

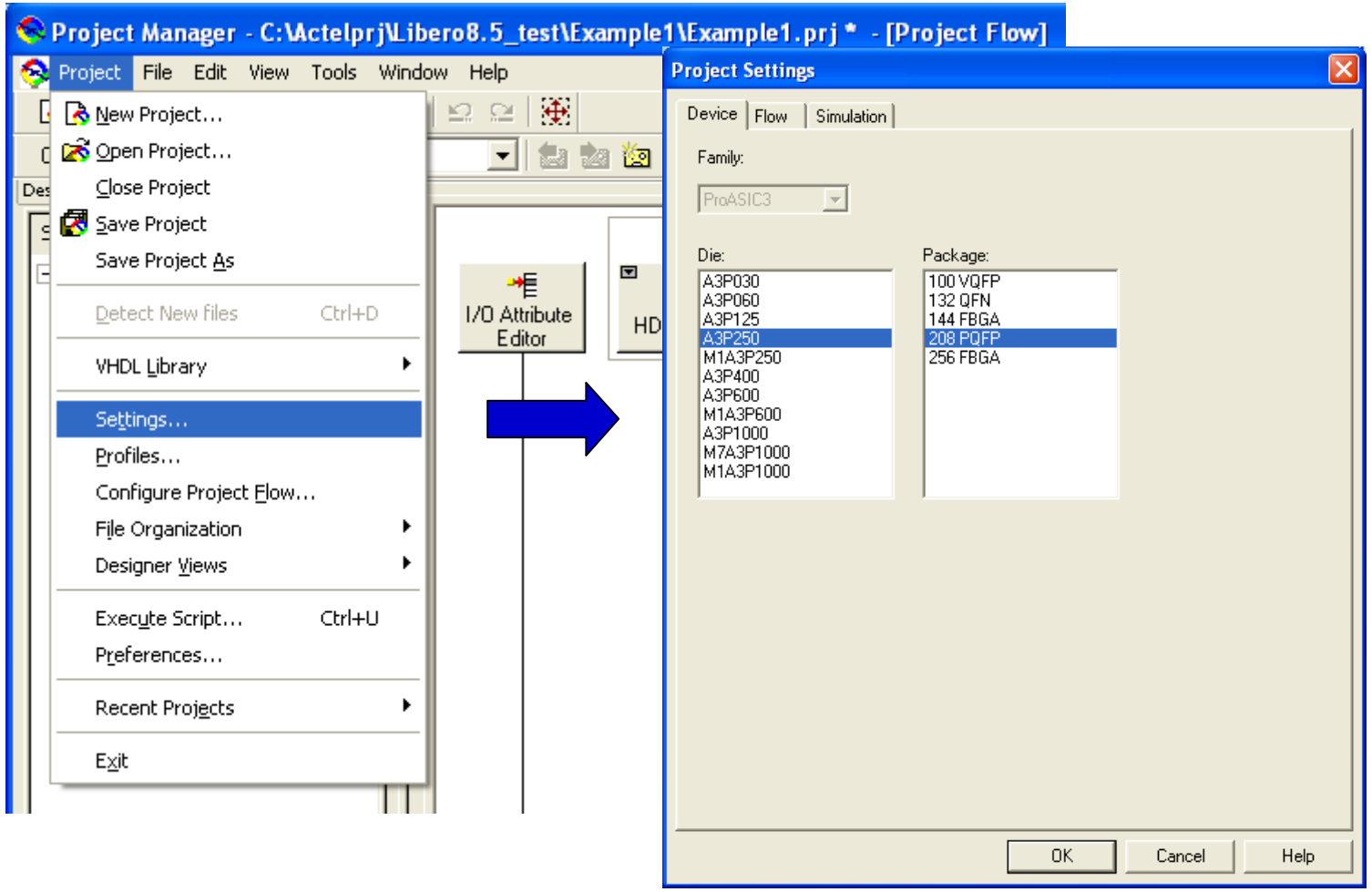
Create, open, close and save projects

"Save-As": Save Project with Different Name and/or in Different Location

List of Recent Projects

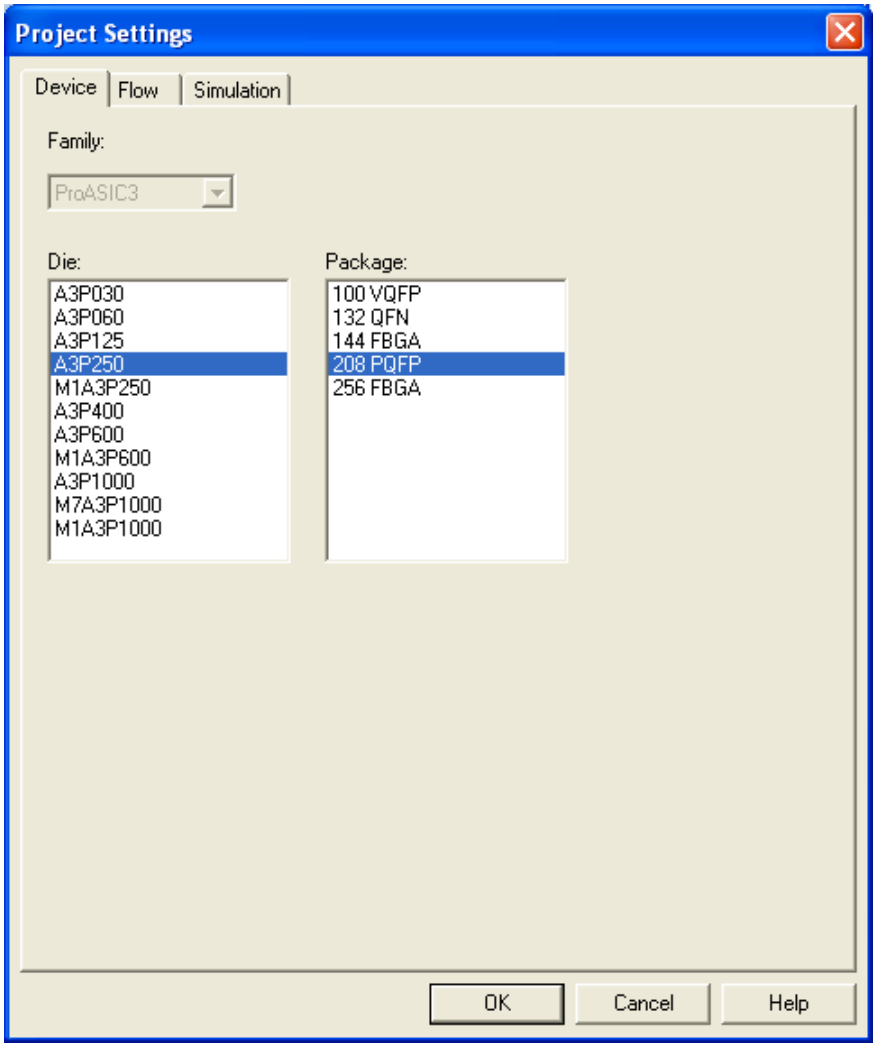


Libero IDE Project Menu
Project Settings



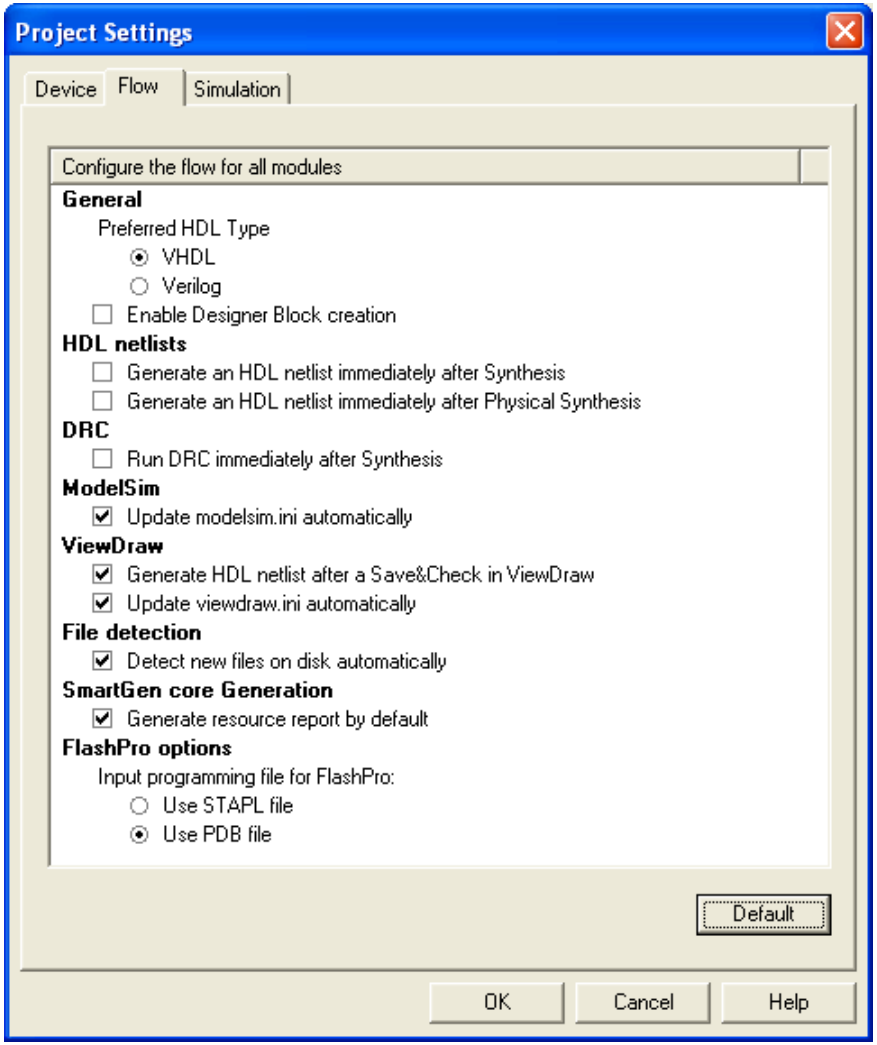
Libero IDE Project Settings

- Device - Change FPGA die or Package
  - Family cannot be changed



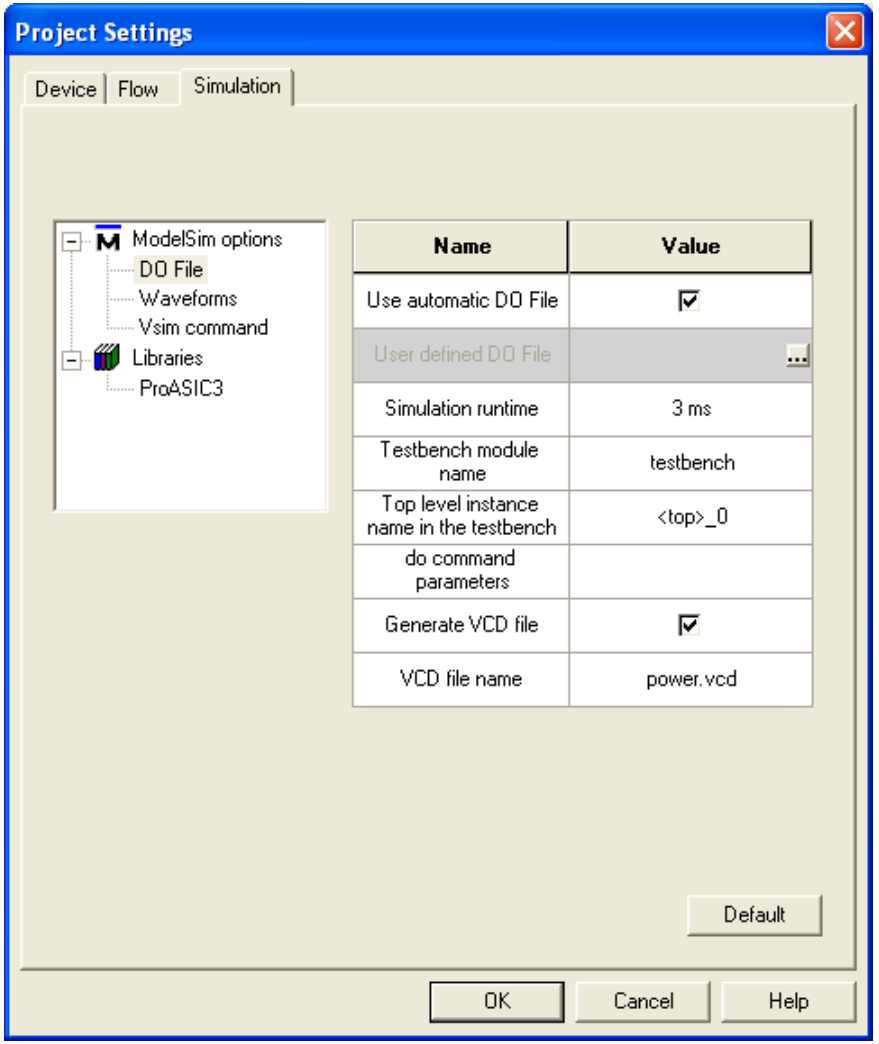
Libero IDE Project Settings

- Device - Change FPGA die or Package
  - Family cannot be changed
- Flow - Configure HDL Netlist and DRC Options



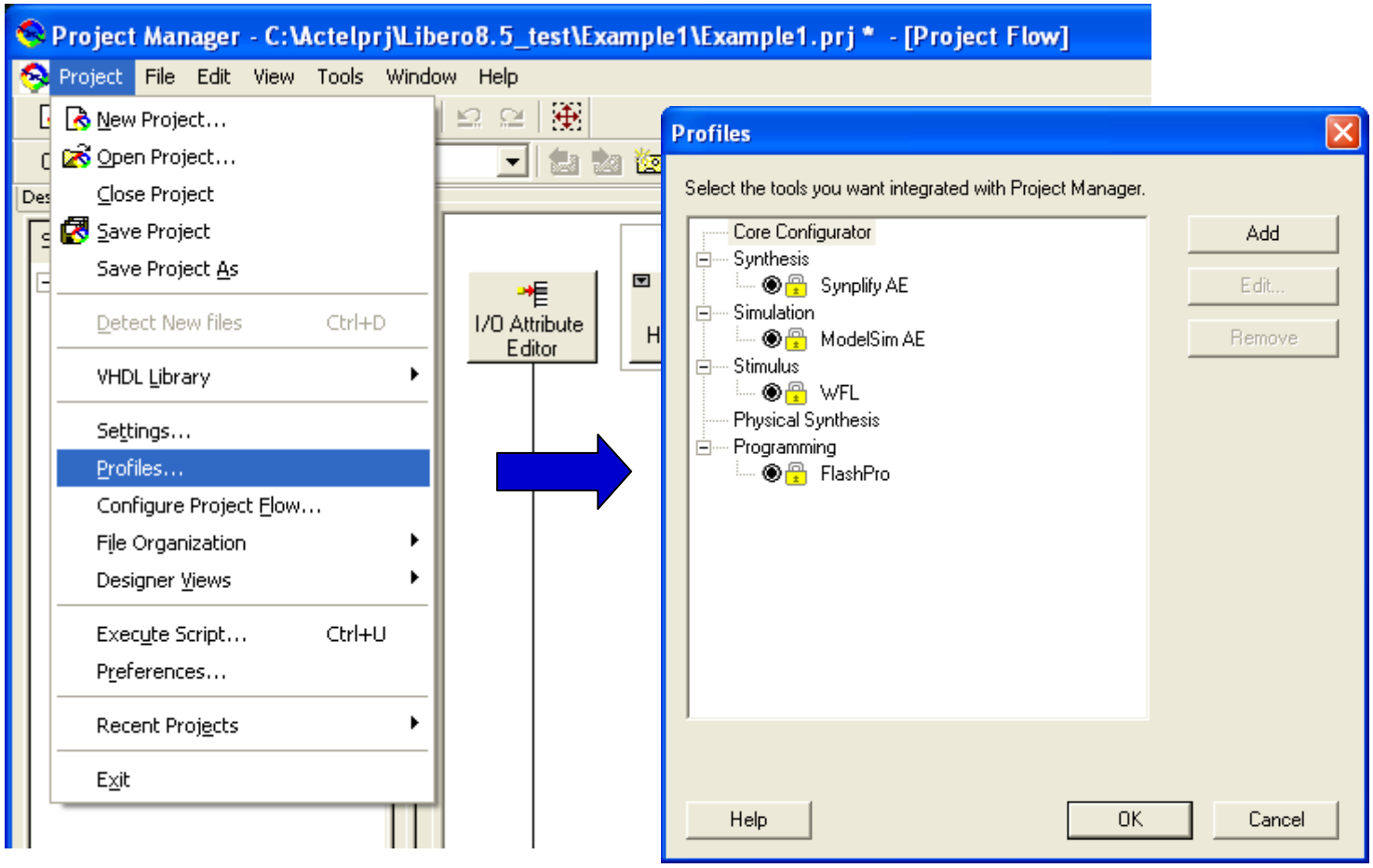
Libero IDE Project Settings

- Device - Change FPGA die or Package
  - Family cannot be changed
- Flow - Configure HDL Netlist and DRC Options
- Simulation - Specify Simulation Options
  - Simulation run time, testbench name location of do files, etc.



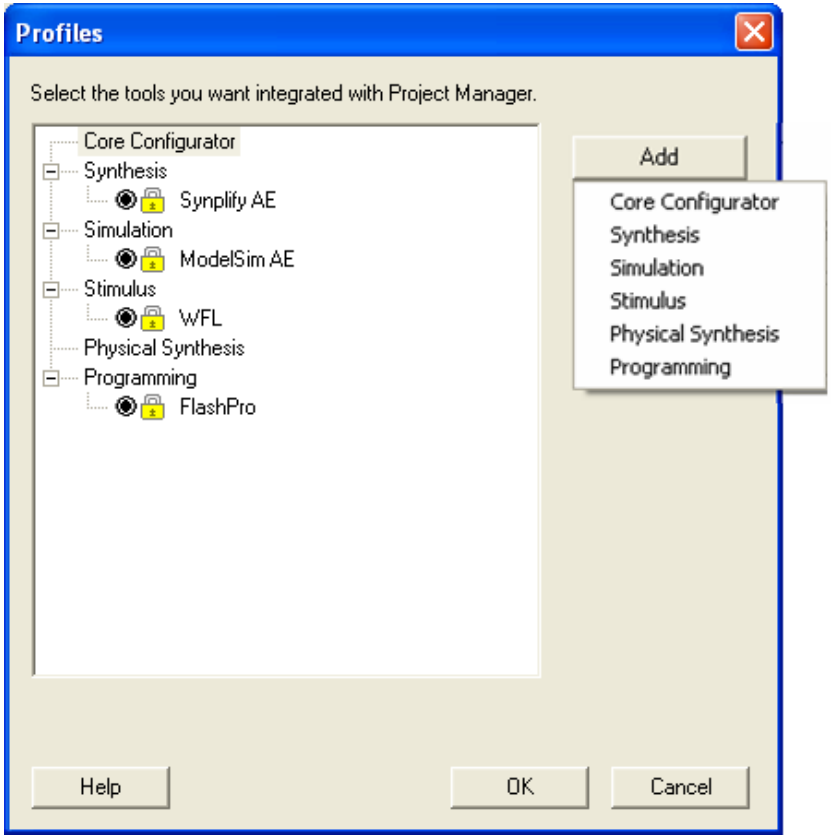
Libero IDE Project Menu
Tool Profiles

- Specify Tools to Use for Project



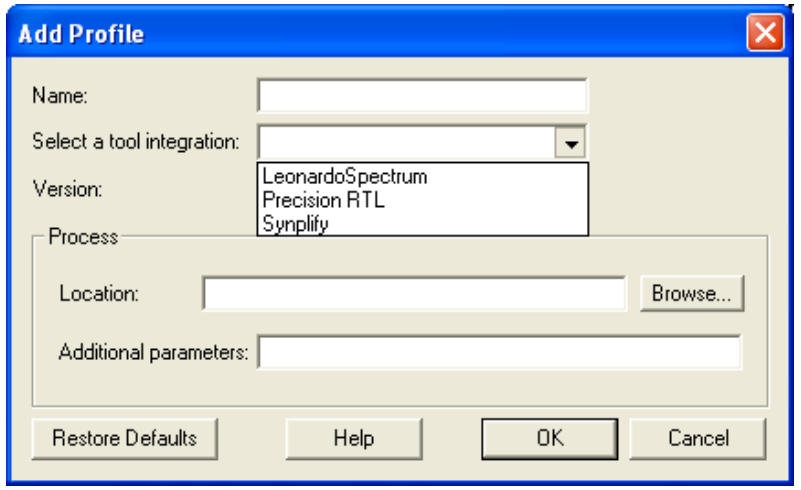
OEM Tools Support
Tool Profiles

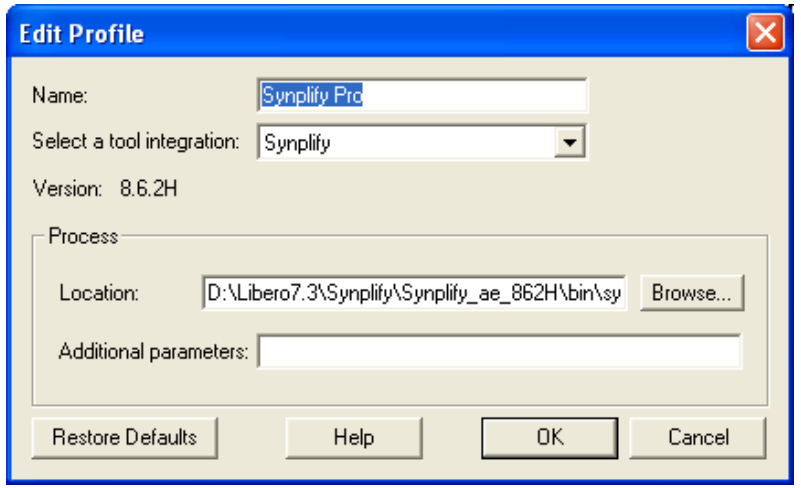
- Create or Edit Tool Profile for Project
  - Project > Profile
  - Select Third-Party Tools & Versions
    - Synthesis
      - Vendor and Version
        - Support for Synplify, Leonardo Spectrum and Precision
    - Simulation
      - ModelSim
        - Select Version
    - Testbench Generation
      - WaveFormer Lite
        - Select Version
  - Name Profile and Save
  - Edit or Add Profiles As Needed



Add / Edit Tool Profiles

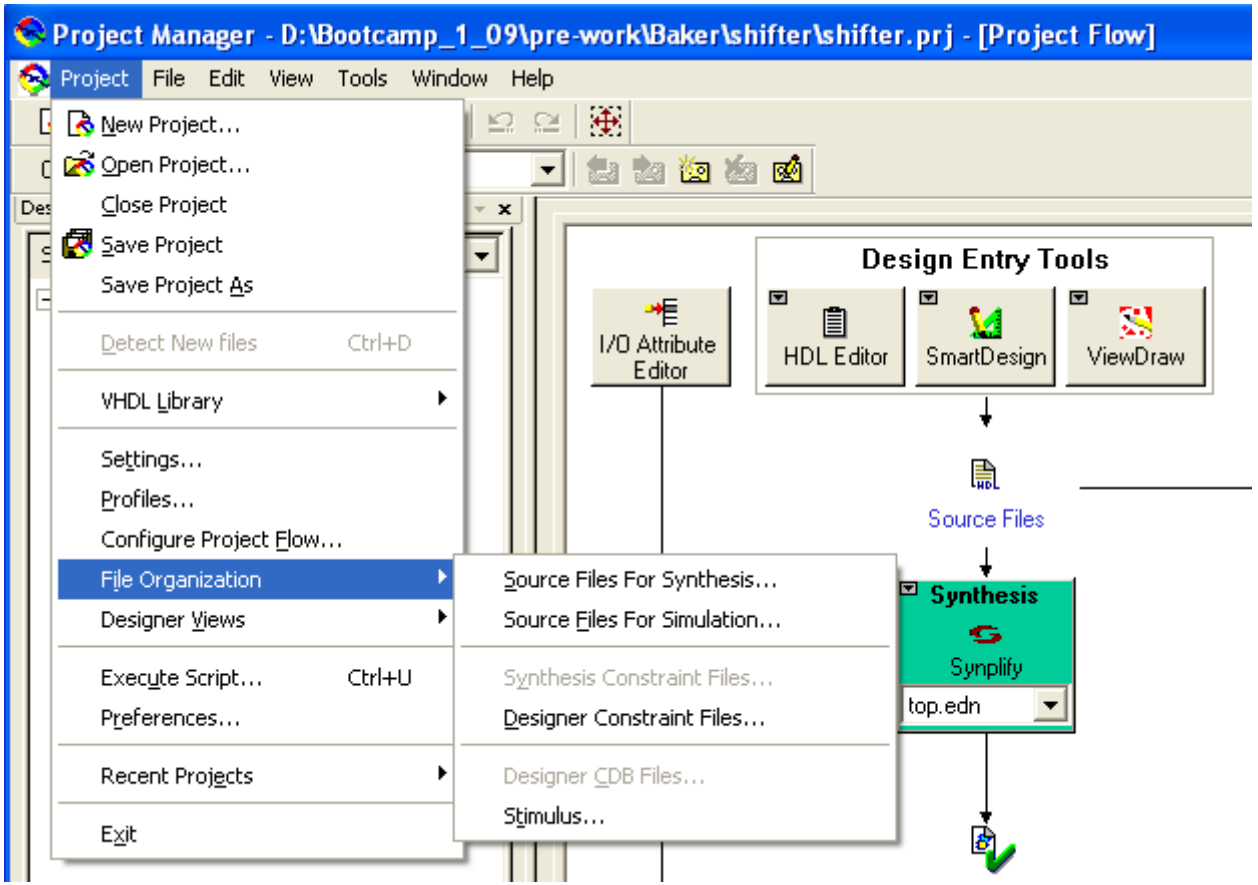
- Add/Edit Profile Requires
  - … Name of Profile
  - … Choosing Tool
    - From Drop-down Menu
    - Choose Version
  - … Choosing Tool Location
    - Browse for Location
    - Specify Location





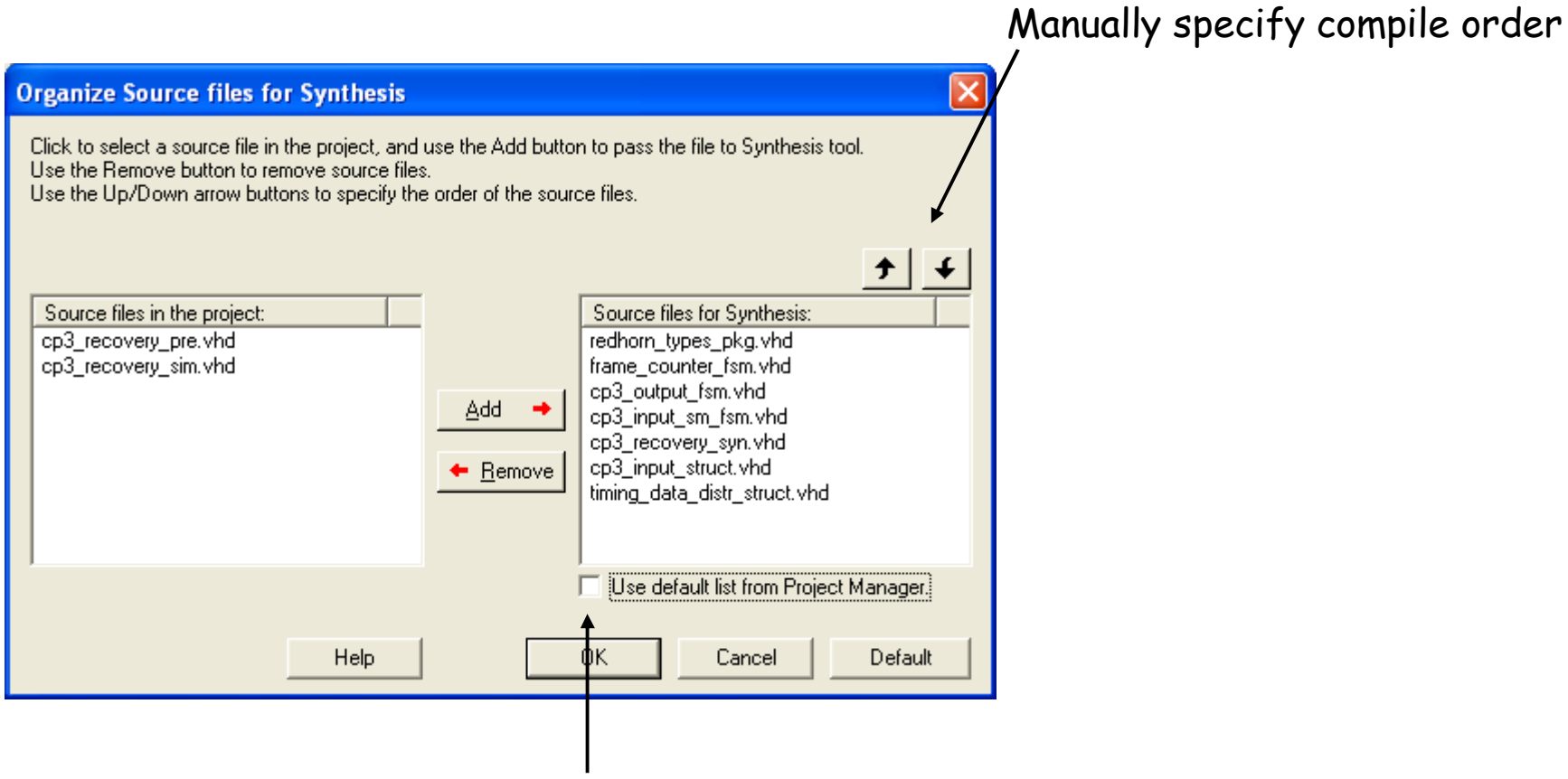
Libero IDE Project Settings
Files Organization

- Specify Package or Header file Compile Sequence and Stimulus Files and Designer Constraint Files



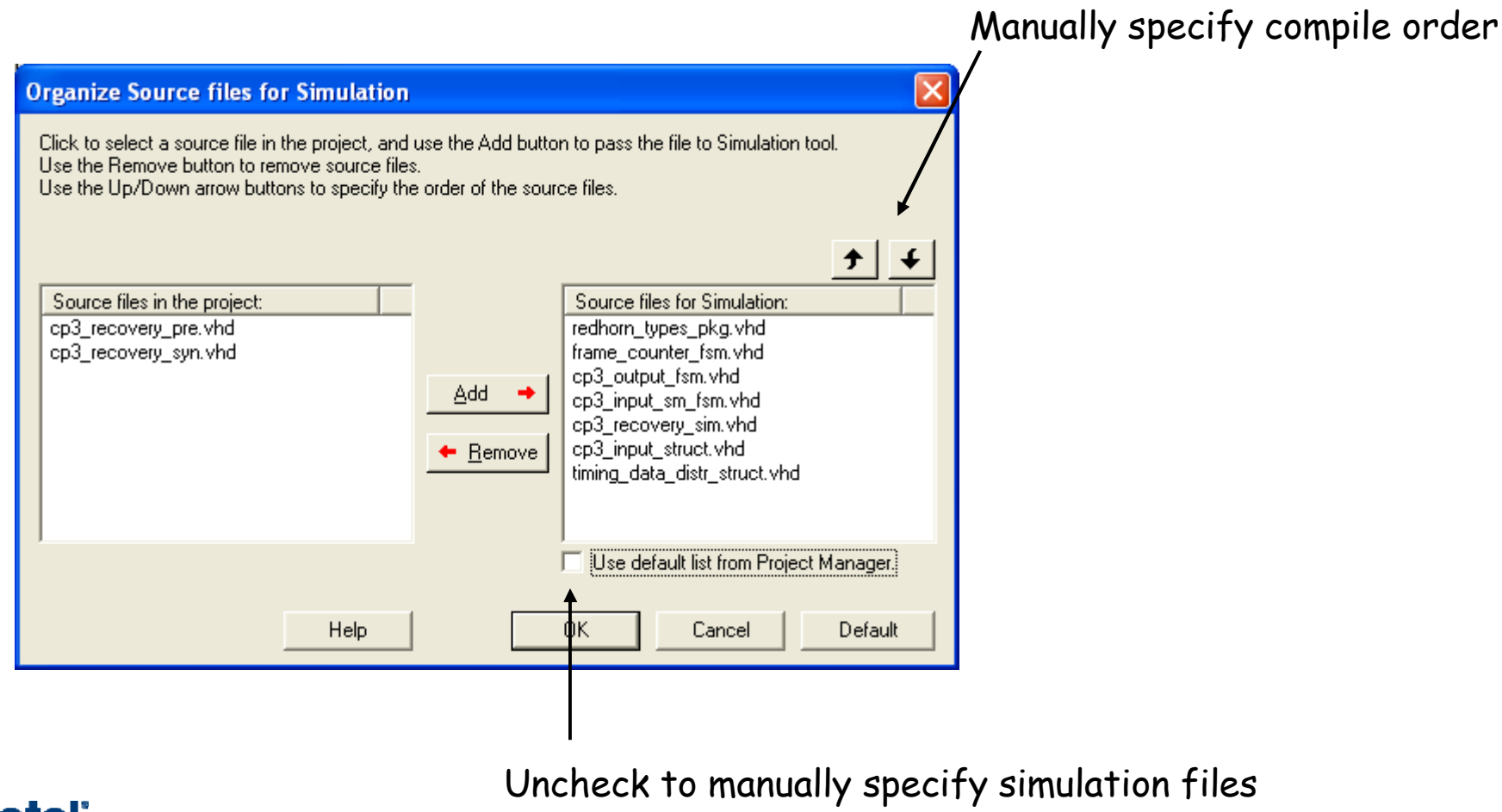
File Organization
Source Files for Synthesis

- Project > File Organization > Source Files for Synthesis
- Specify the Order of Synthesis Files in the Project Manager



File Organization
Source Files for Simulation

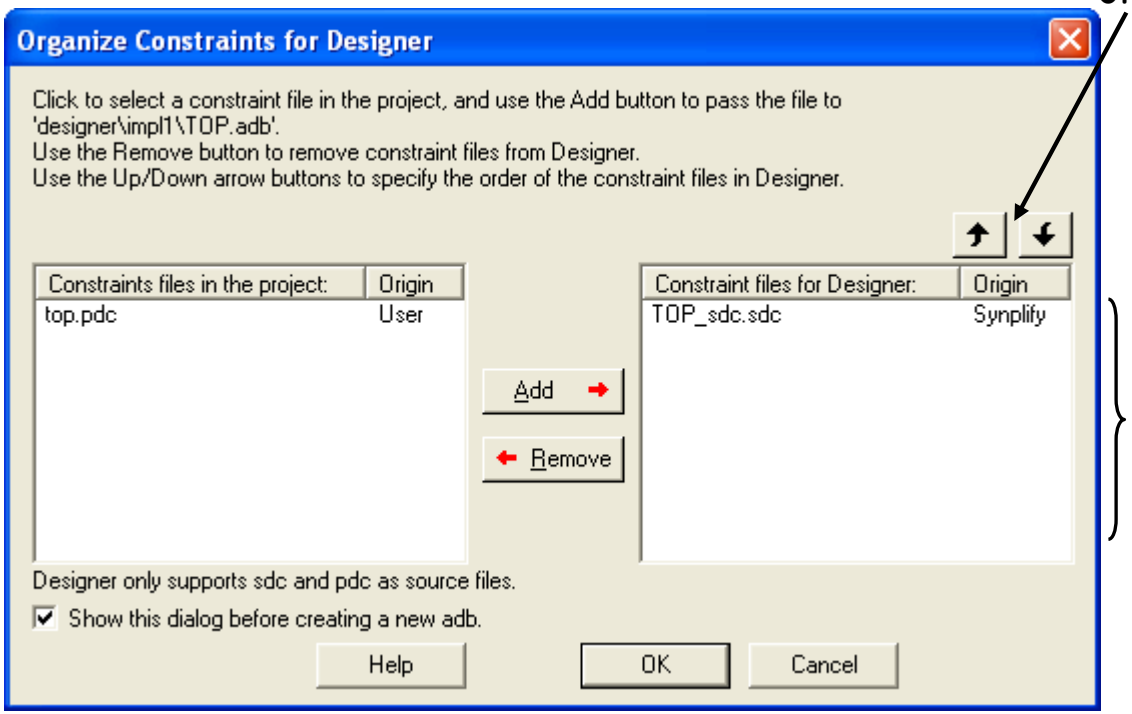
- Project > File Organization > Source Files for Simulation
- Specify the Order of Project Files for Simulation



Files Organization
Designer Constraint Files

- Project > File Organization > Designer Constraint Files
- Specify Designer Constraint Files
  - Timing Constraints (.sdc) or Physical Constraints (.pdc and .gcf)
    - Files may come from Synthesis tool or generated by the engineer

Specify file import order

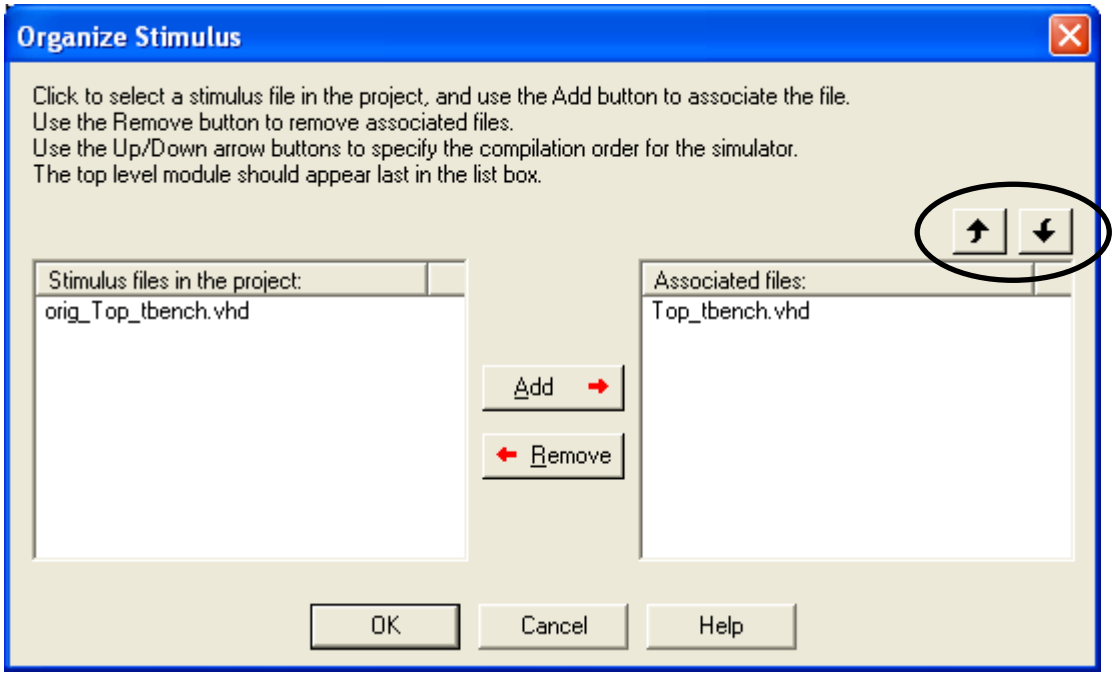


Constraint files in project

Constraint files sent to Designer

File Organization
Stimulus

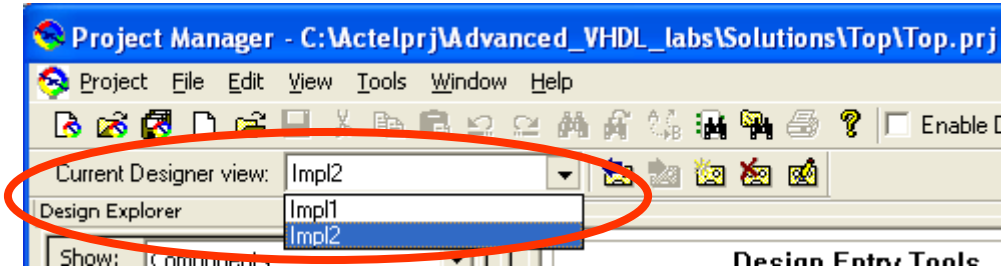
- Specify Stimulus Files for Simulation
- Libero Supports Hierarchical Testbenches
  - Select Multiple Files
  - Libero Does Not Automatically Determine Compile Dependencies of Stimulus Files

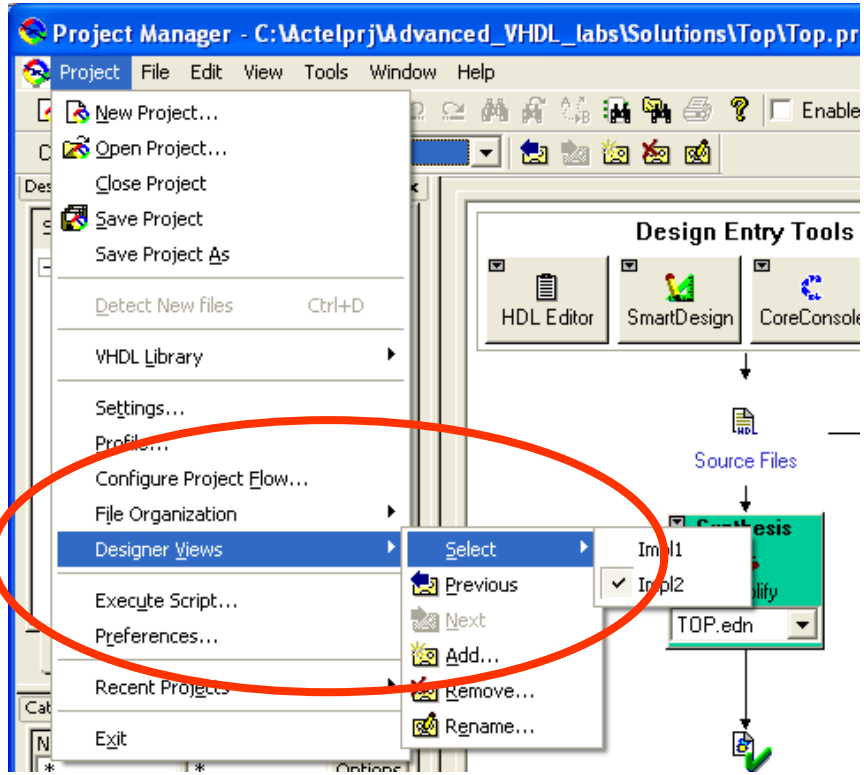


Set compile dependencies when testbench contains multiple files

Libero IDE Project Menu
Designer Views

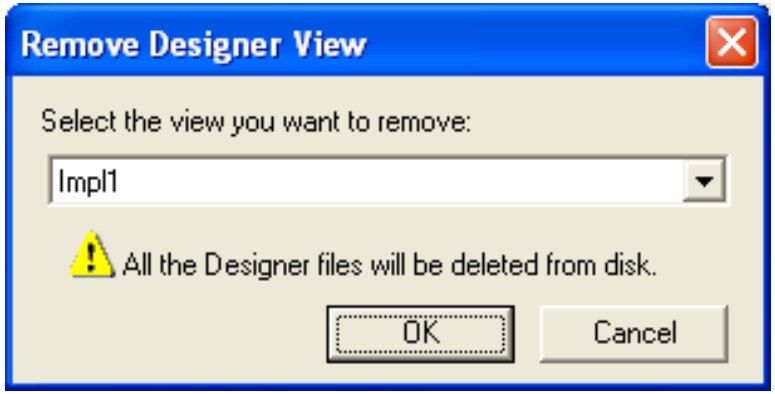
- Select Different Views from Menu or Toolbar
  - Check layout results for each Designer view
  - Select from the Current Designer view drop-down menu and run layout

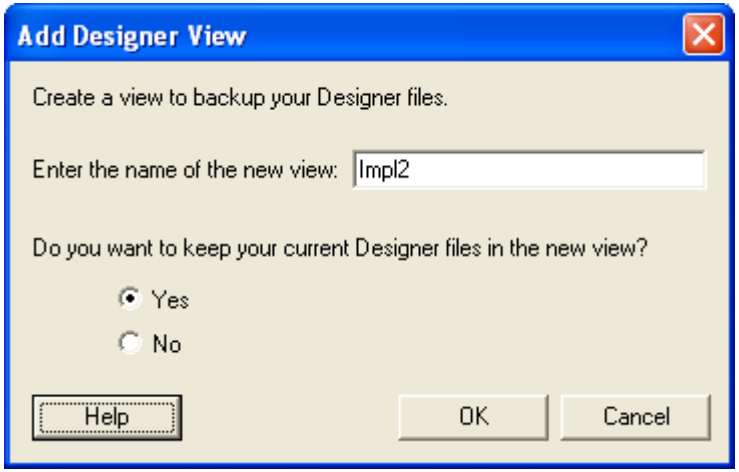


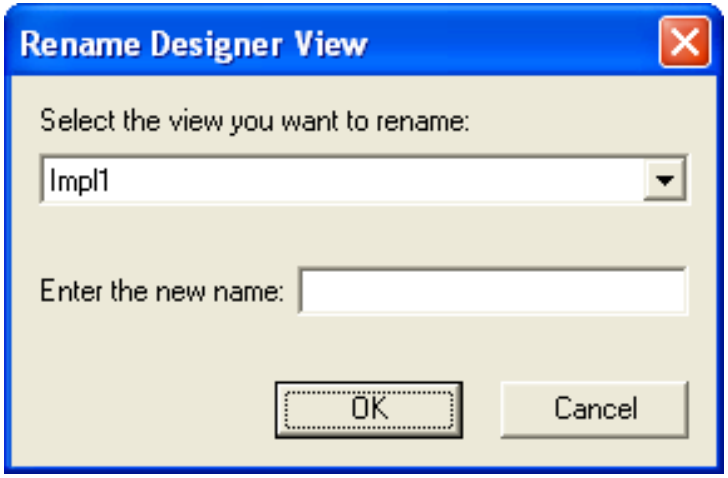


Libero IDE Designer Views
Add, Remove and Edit

- Add, Rename or Remove Views

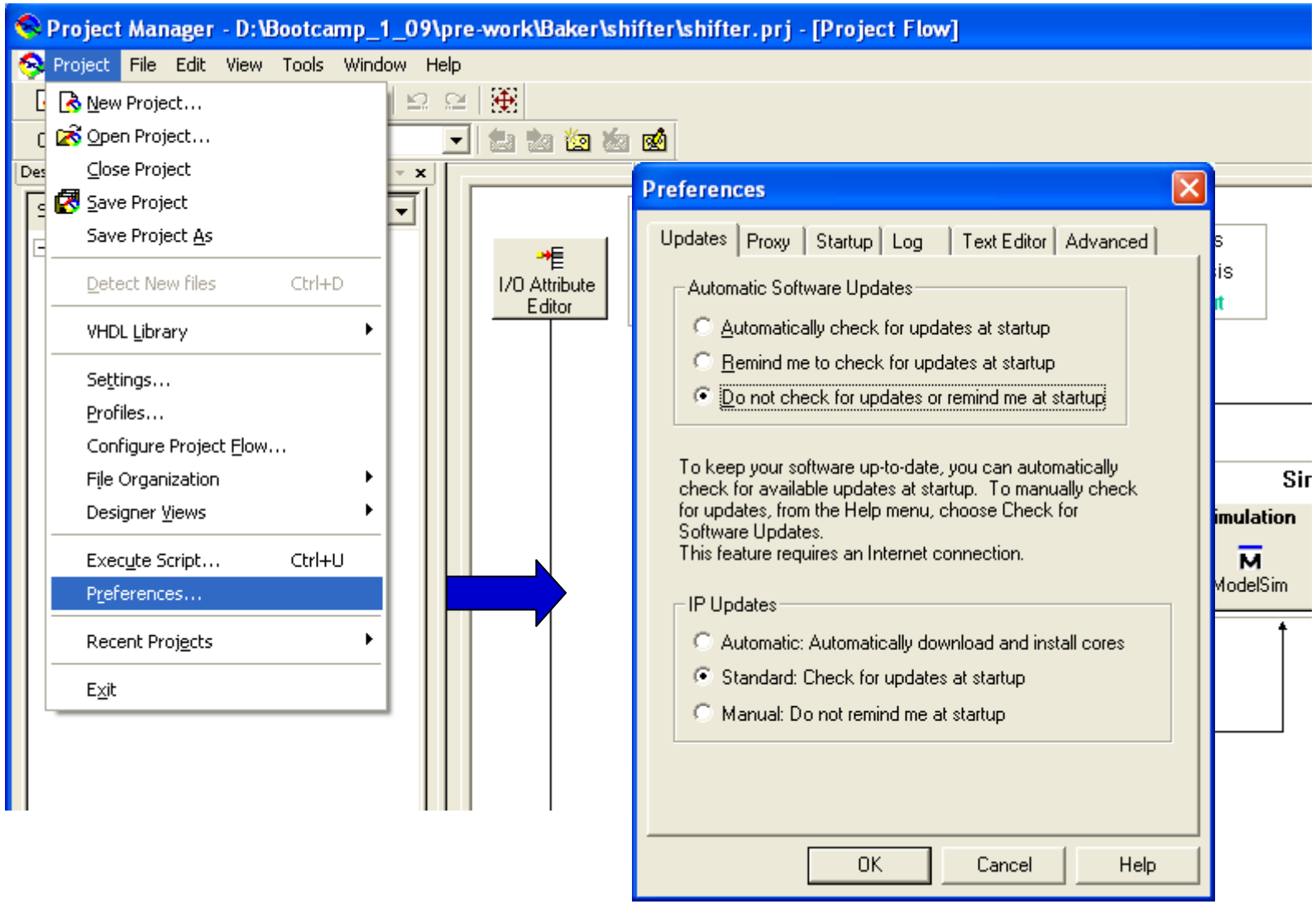






Libero IDE Project Menu

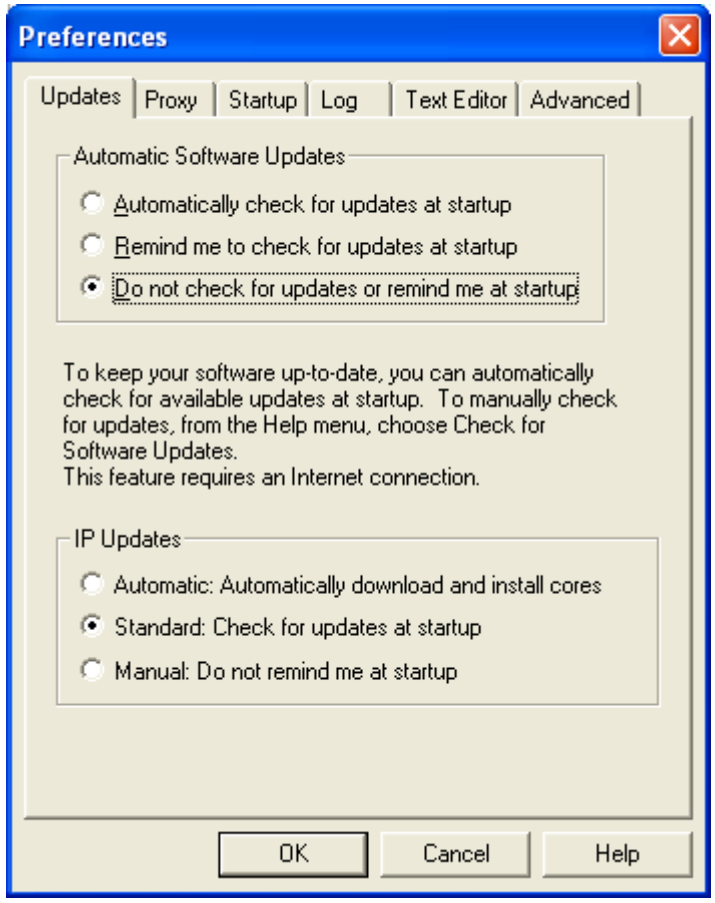
User Preferences



Libero IDE User Preferences

Automatic Check for Updates

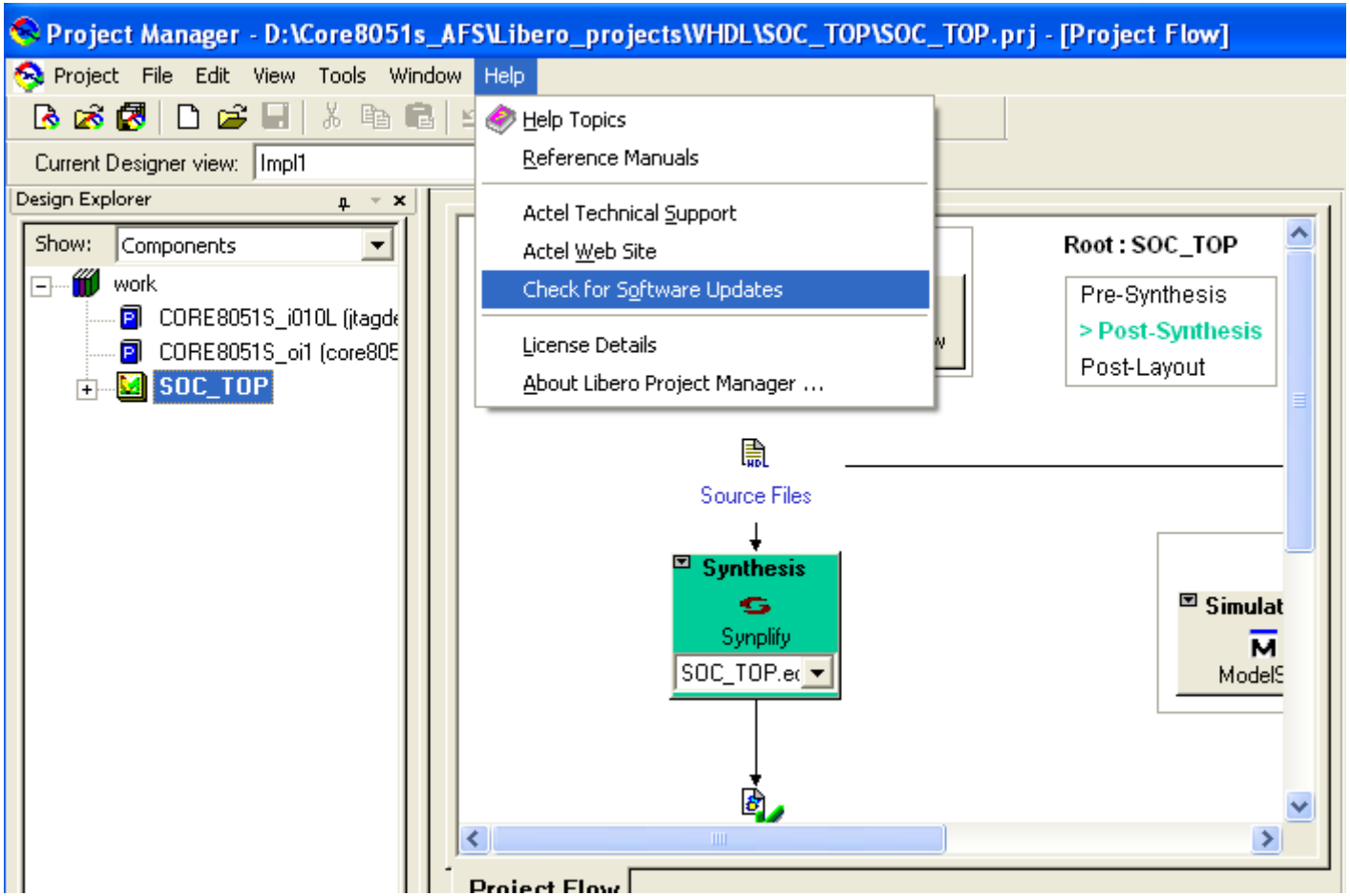
- Enable or Disable Checking for Software and IP Updates When Libero is Launched



Libero IDE

Checking for Software Updates Manually

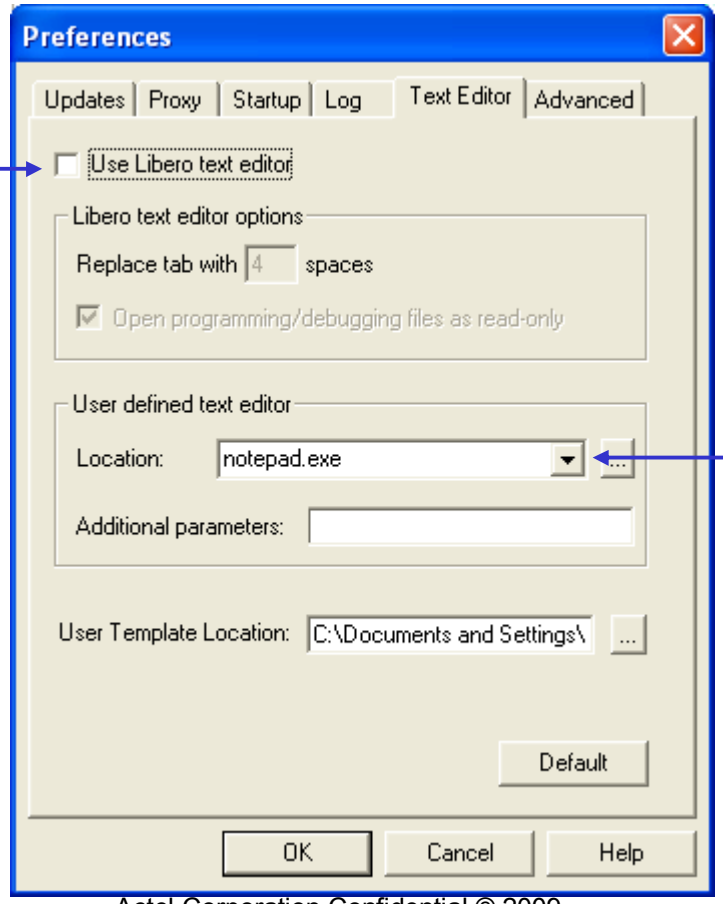
- Manually Check for Software Updates from Help Menu



Libero IDE User Preferences
Text Editor

- Text Editor Selection
  - Use Libero IDE Text Editor or External Text Editor
    - File > Preferences Text Editor Tab

Un-check to use external text editor

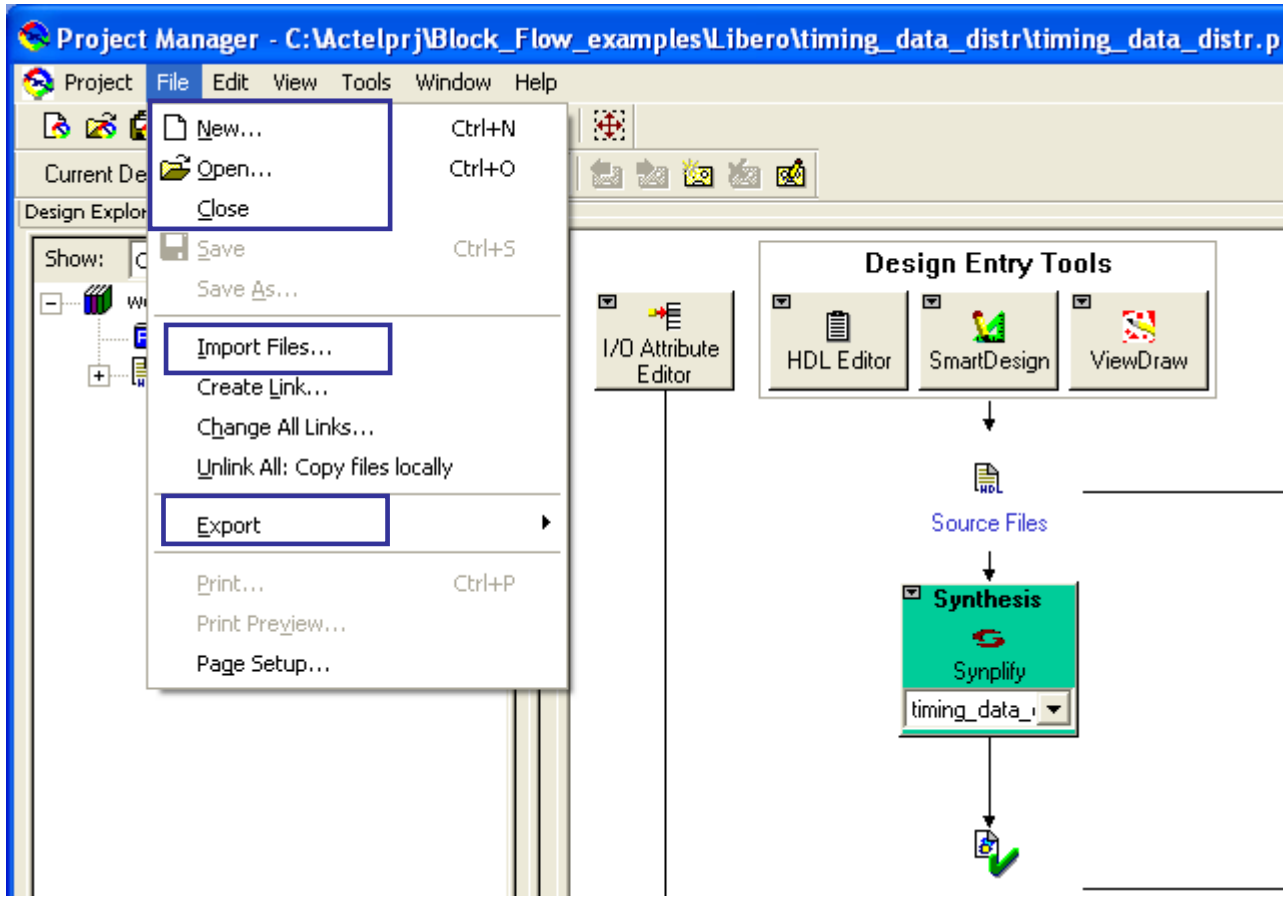


Enter location of external text editor if selected

Libero IDE
File Menu Options

Create, open, save and close files

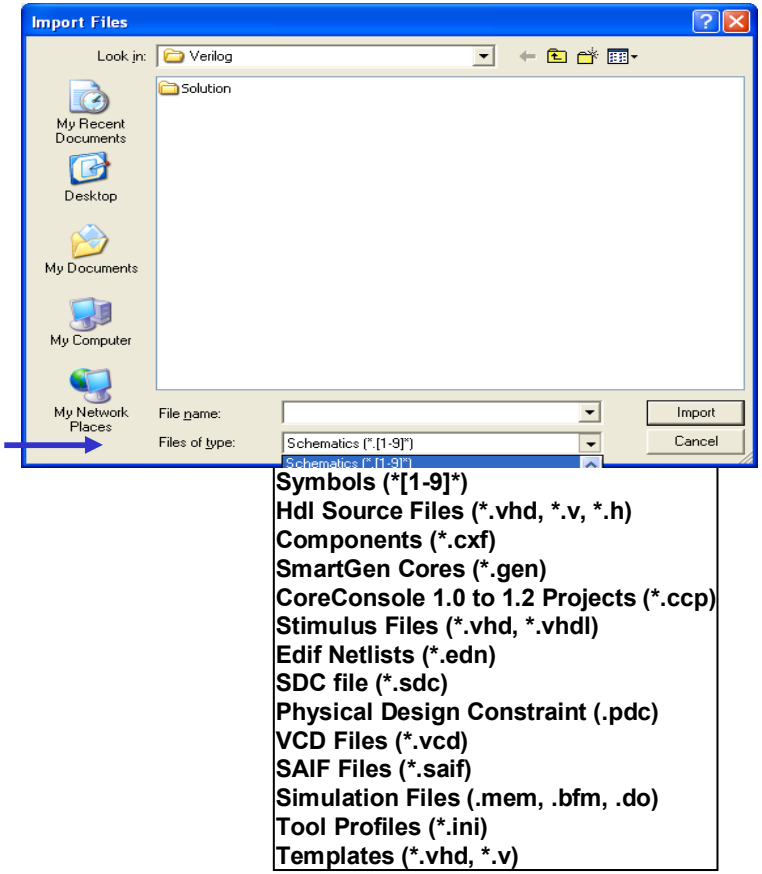
Import and export files

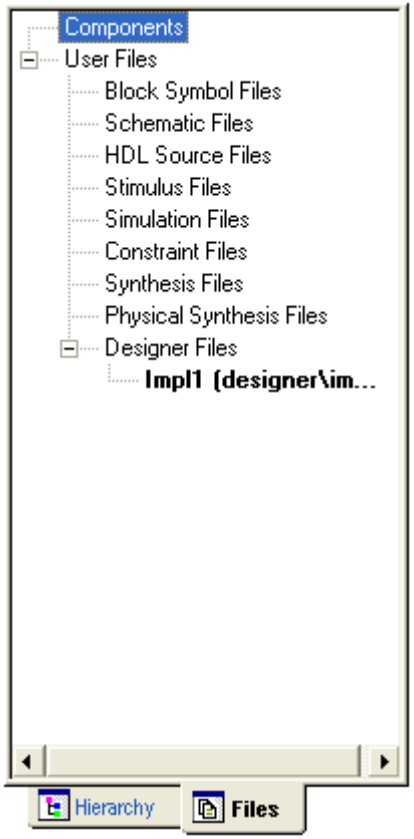


Libero IDE File Menu
Importing Files

- Existing Design Files Can Be Imported into Libero Project
  - Schematics, Symbols, HDL (VHDL or Verilog), Stimulus (VHDL or Verilog), SmartGen Cores, EDIF Netlists, SDC Files, Constraint Files, Tool Profiles

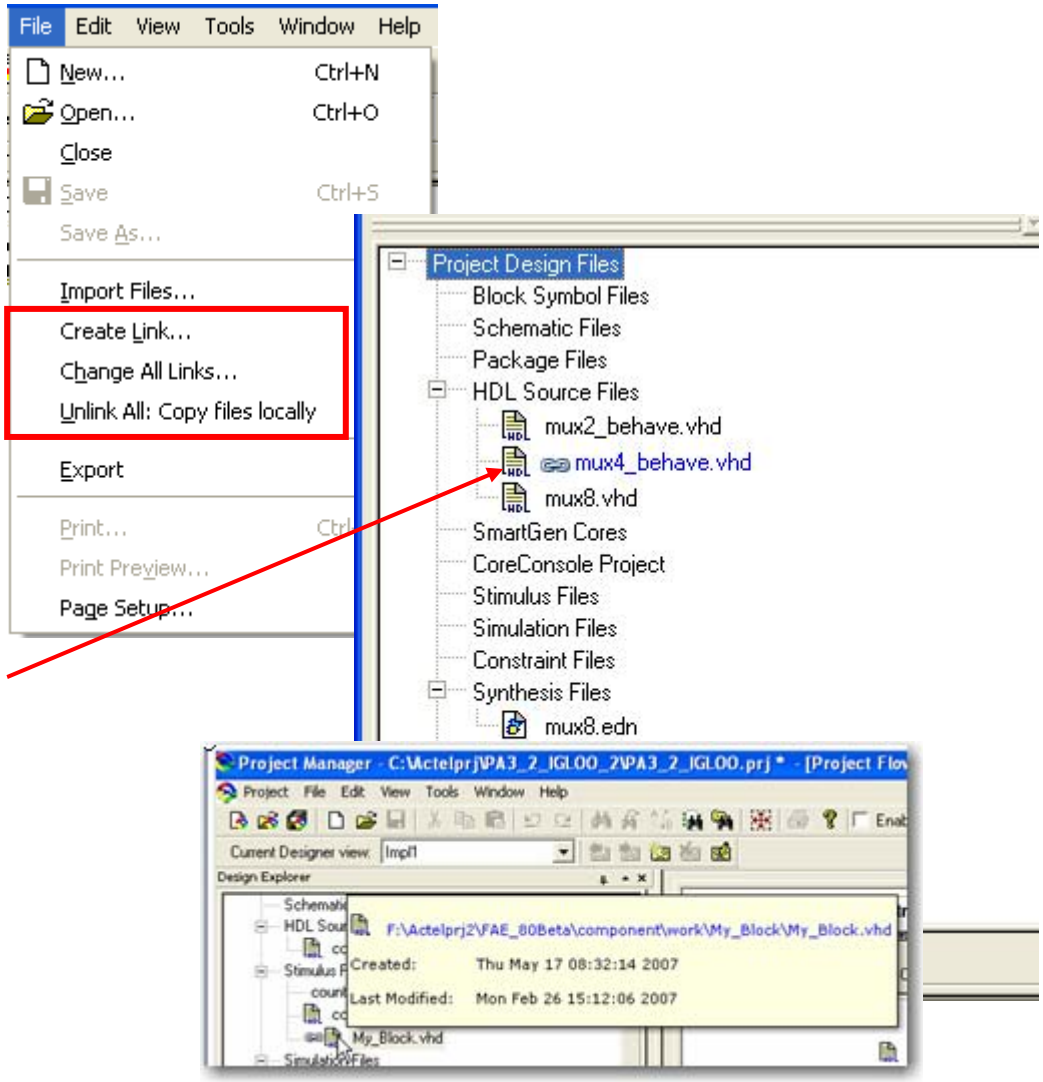
Select file type from pull-down menu





Project Manager File Linking

- Link External Files to Project
  - Package
  - Source HDL
  - Stimulus
  - Constraint
- Linked File Shows as Blue Text With Icon
- Right Click on Link to Remove or Change

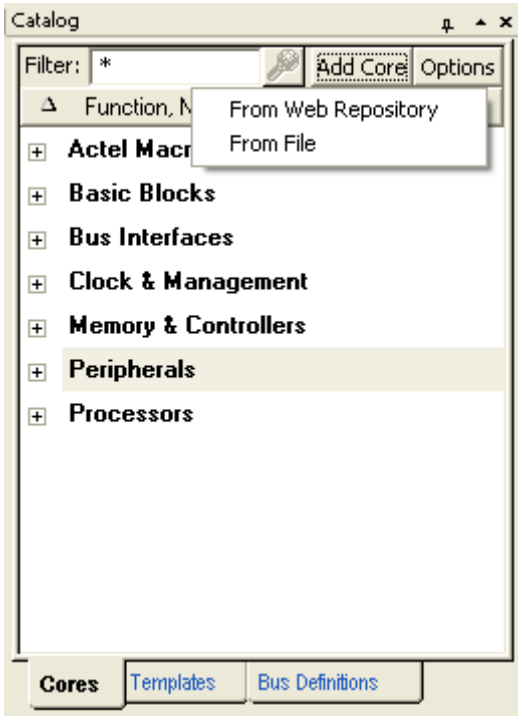


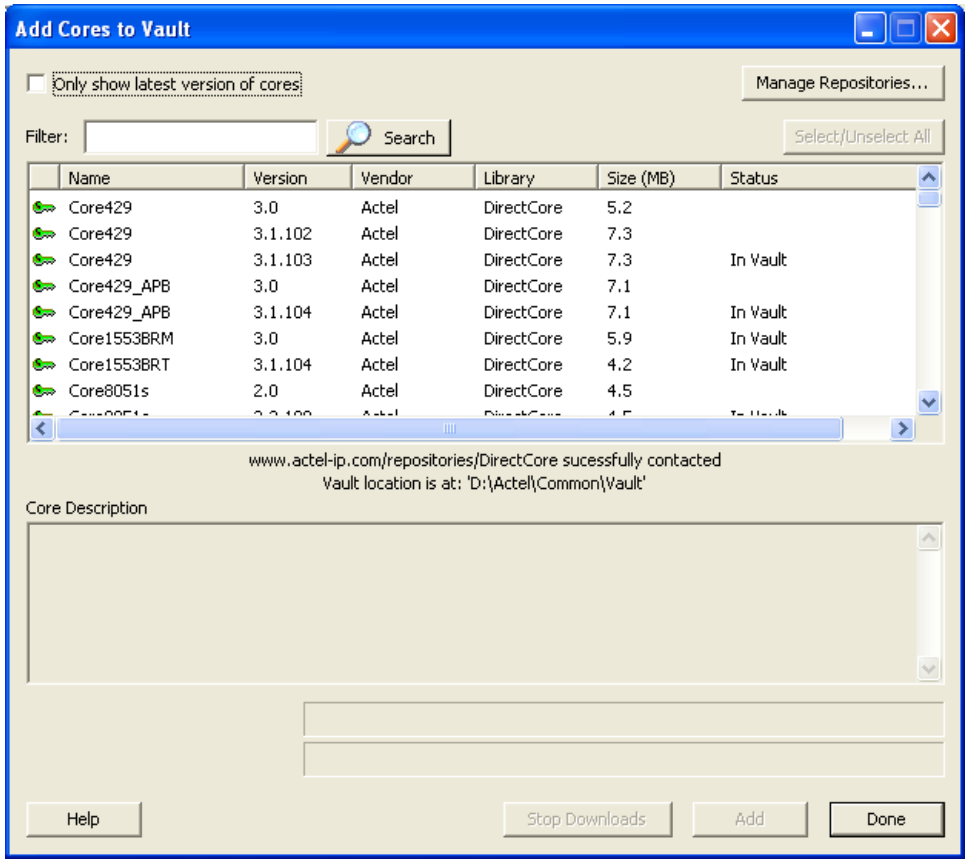
Project Manager File Linking

- Libero Does Not Manage Source File State
  - Linked to file is alterable by project user
  - This may/may not be acceptable by source file owner
- Options:
  - Save the linked source with a new file name
  - Make the Linked to file “read only”
  - Bring (import) the file to the local disk

Project Manager Catalog Options

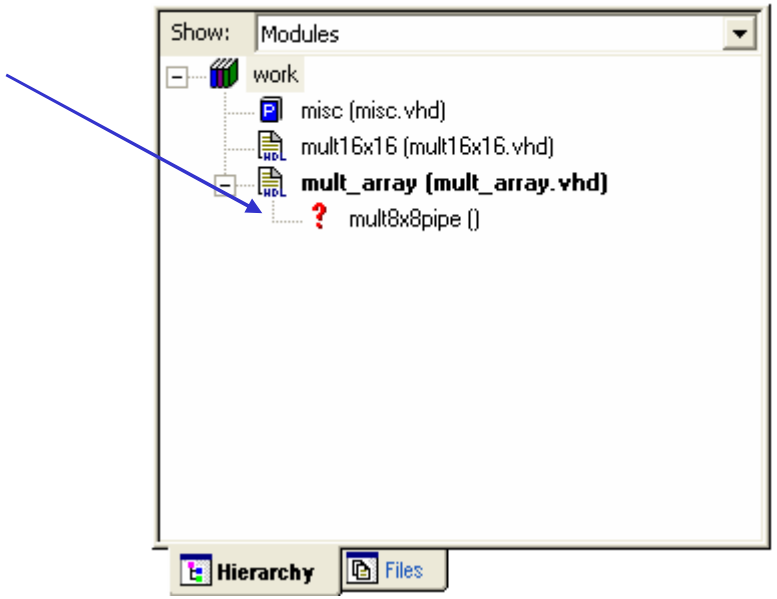
- Download new Cores to IP Catalog





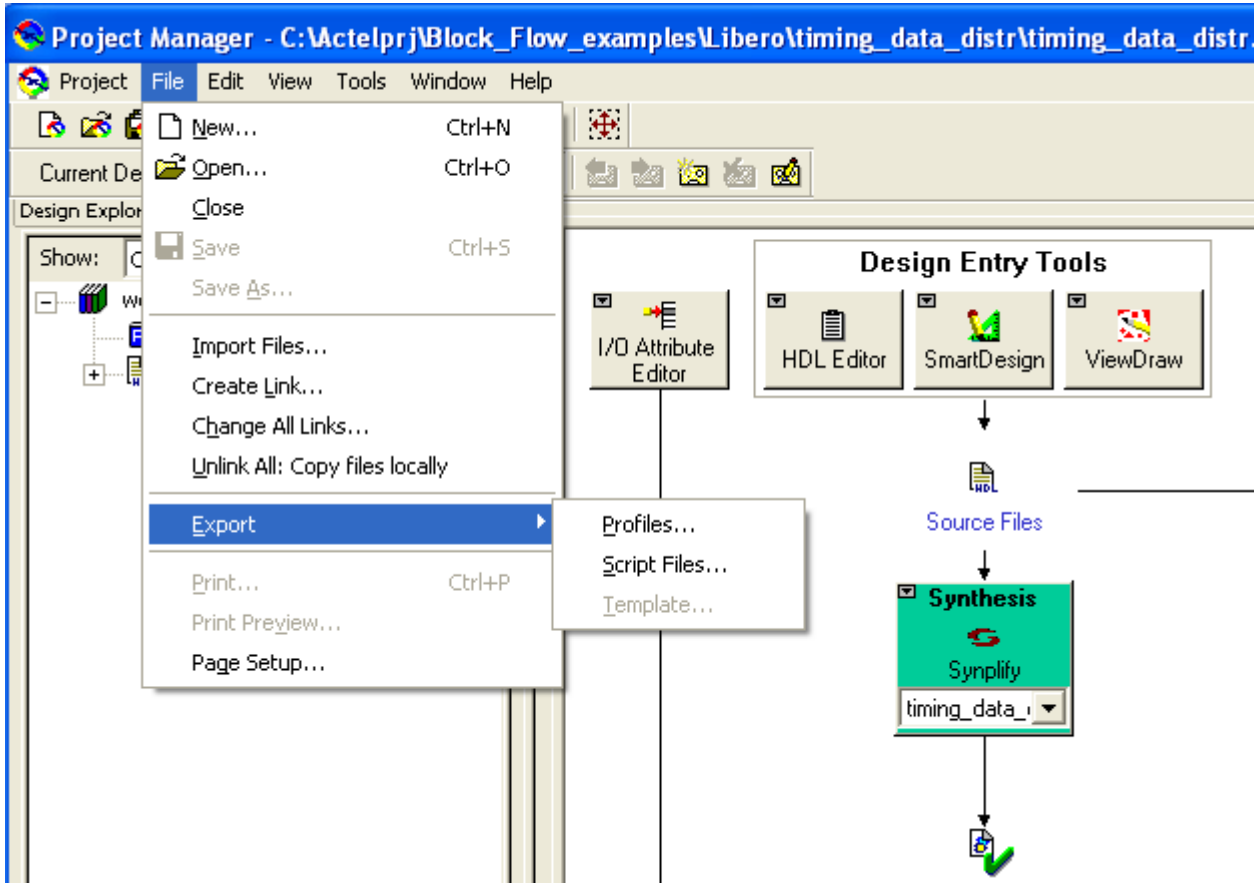
Libero IDE Design Hierarchy

- Libero Displays Design Hierarchy on Hierarchy tab in Design Explorer
- Missing Files Indicated with "?"



Libero IDE File Menu
Exporting Files

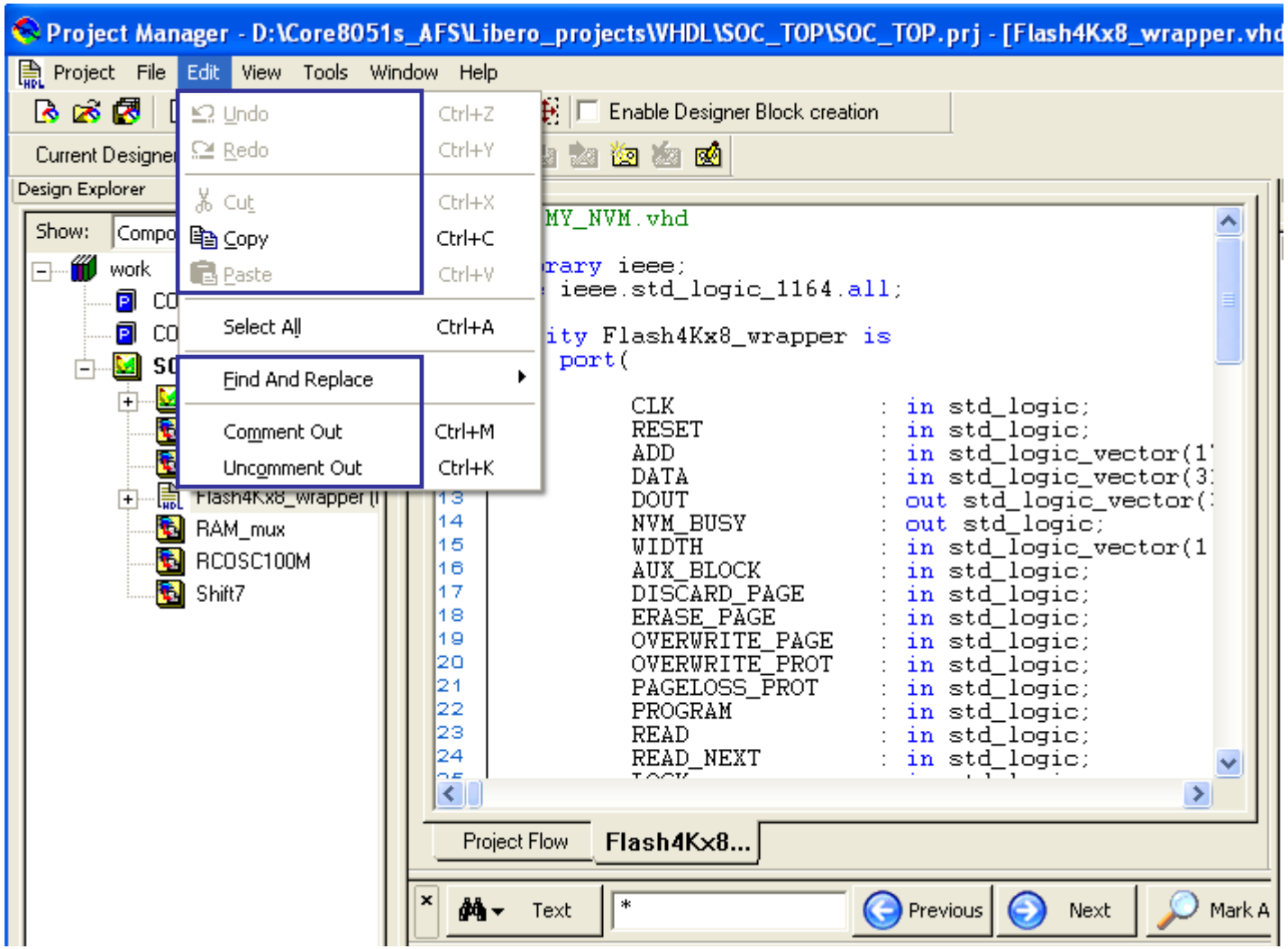
- Export Tool Profiles and TCL Scripts



Libero IDE
Edit Menu Options

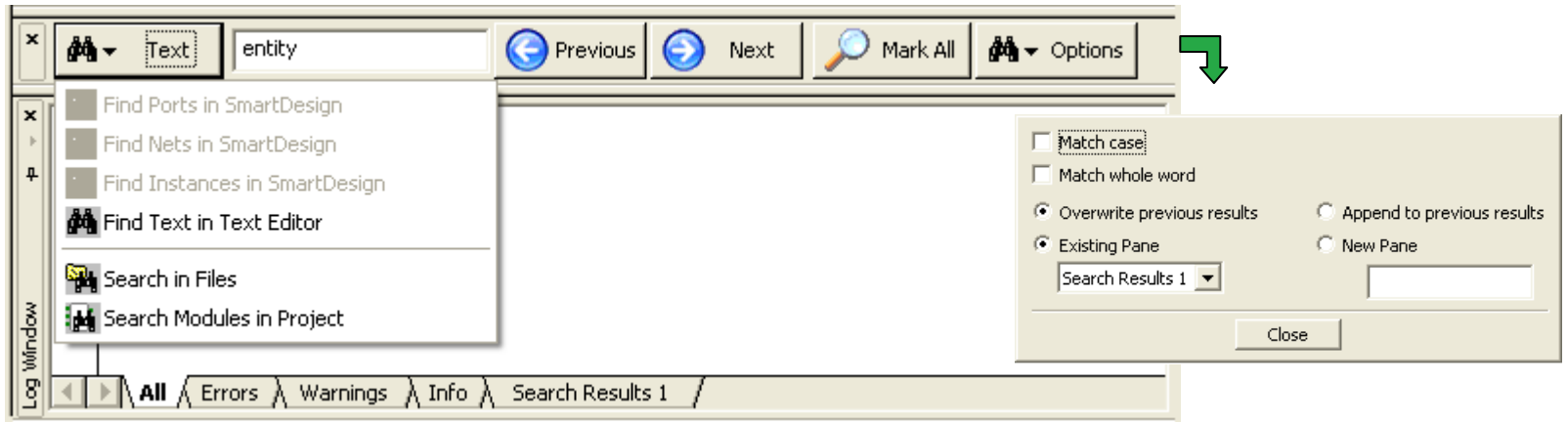
Copy, Paste, Undo

Find and Replace
Add / Remove
Comments



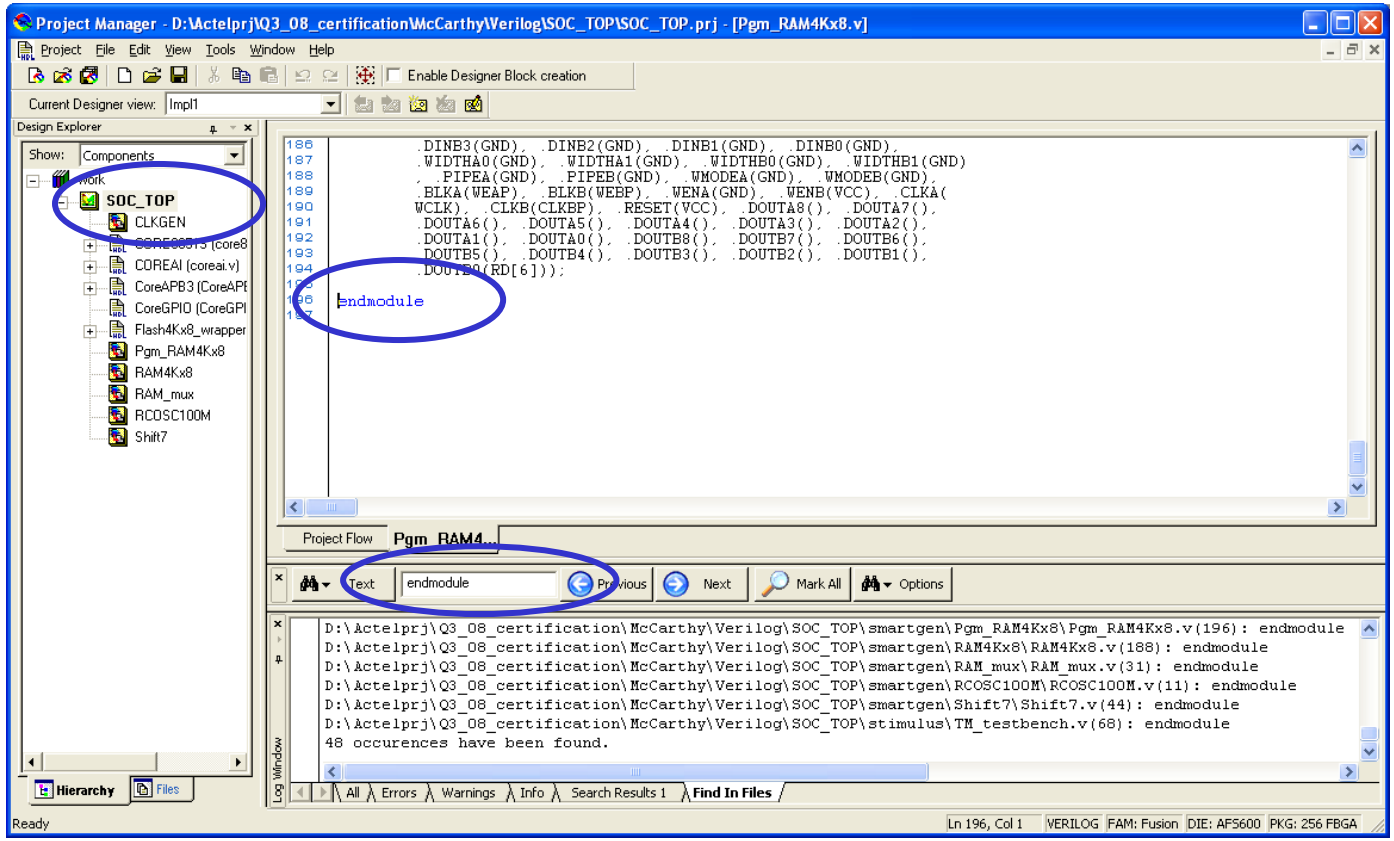
Libero IDE Edit Menu
Find in Files

- Edit => Find and Replace
  - Search for Files, Words, etc
  - Specify by File Types
  - Specify where to Search
    - Text editor, files in project, modules in project
  - Match Whole Word
  - Match Case
  - Regular Expression
- Results Shown in “Search Results” Tab in Log Window



Find in Files
Cross Probing

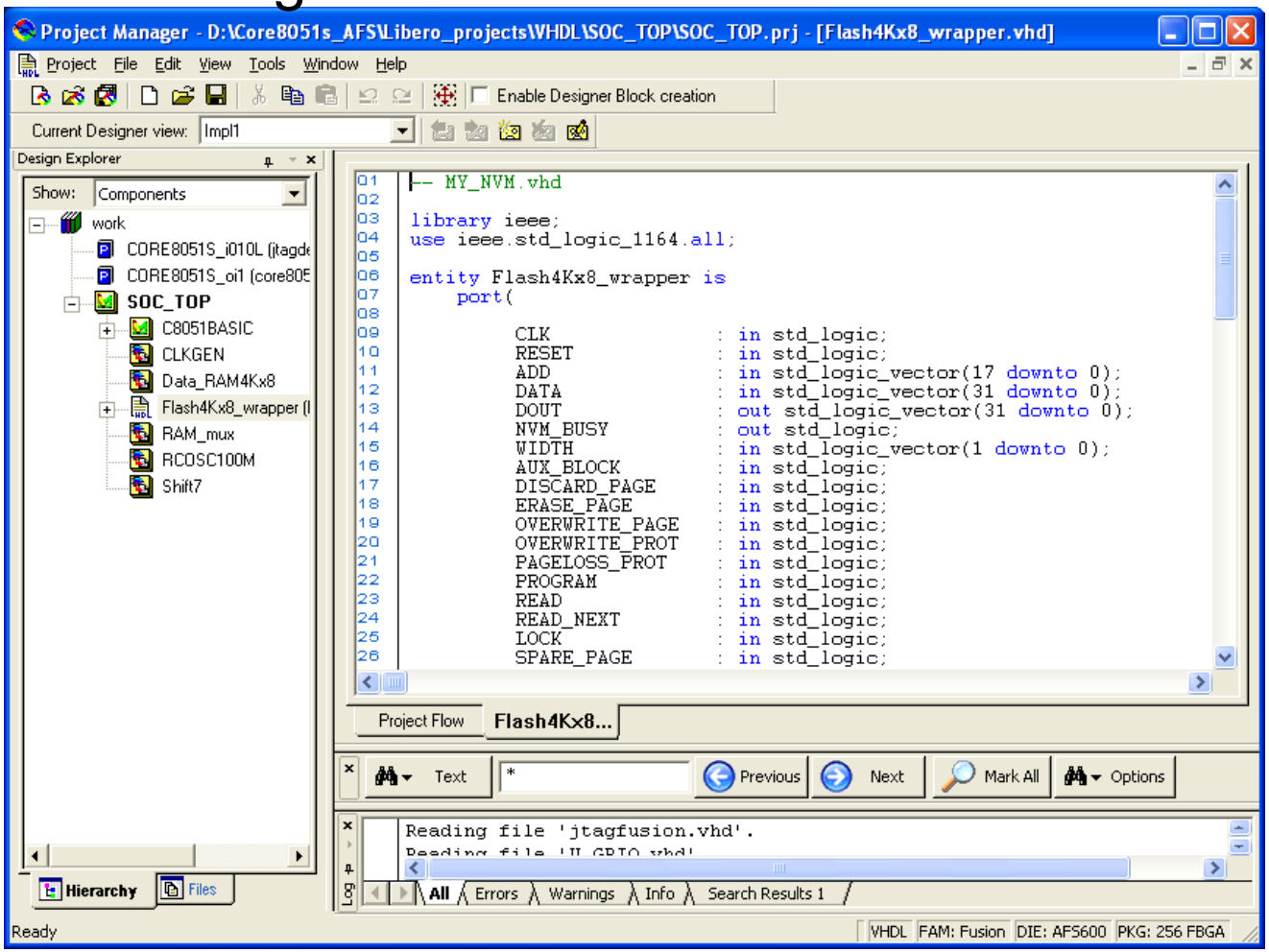
- Selecting File Name Presented in Find in Files Log Window …
  - … Opens Selected File in Libero Text Editor
  - … Highlights Match



Libero IDE Edit Menu

Comment

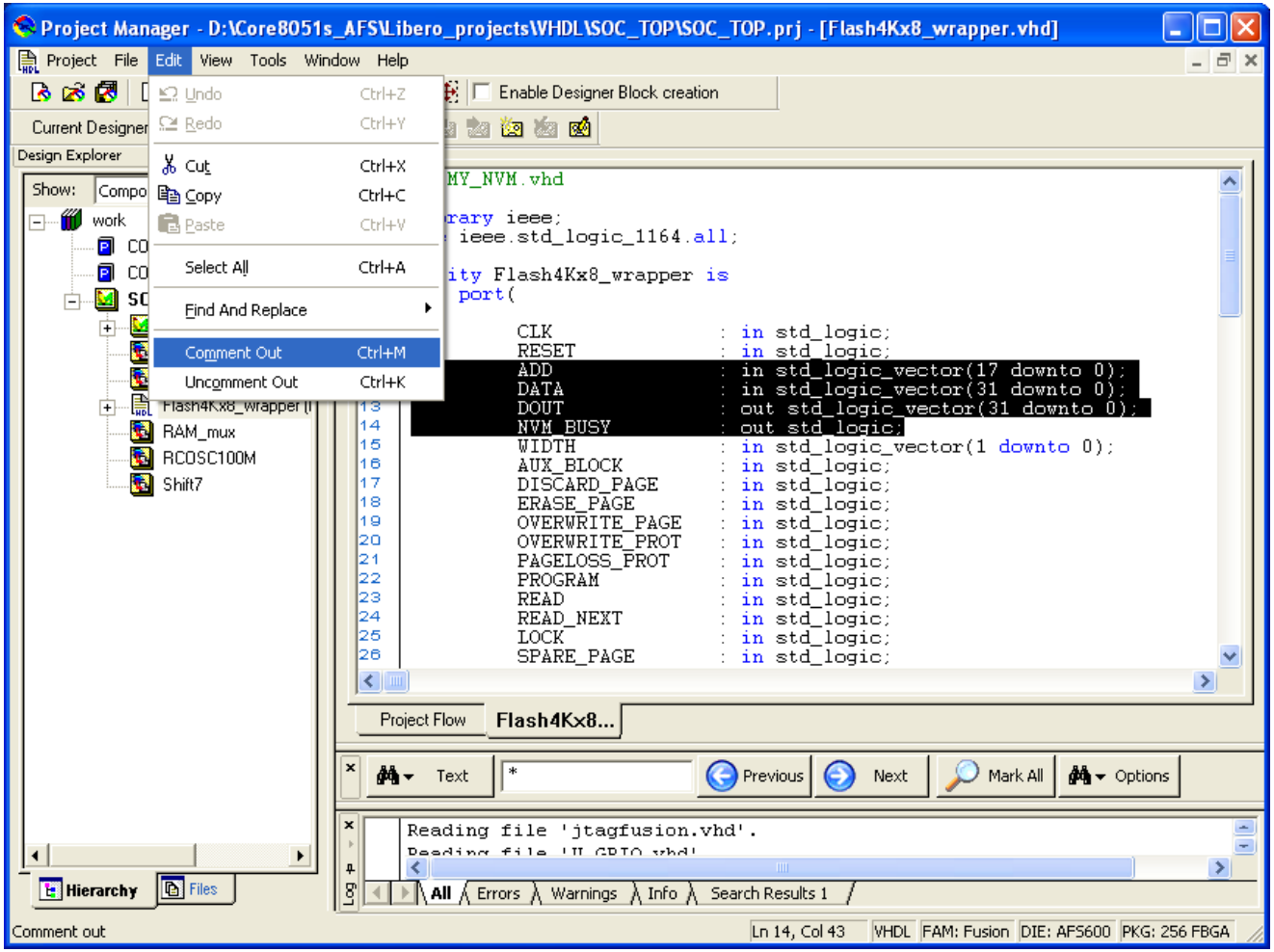
- Comment Command Allows Users to Comment Sections of VHDL or Verilog HDL



Libero IDE Edit Menu

Comment

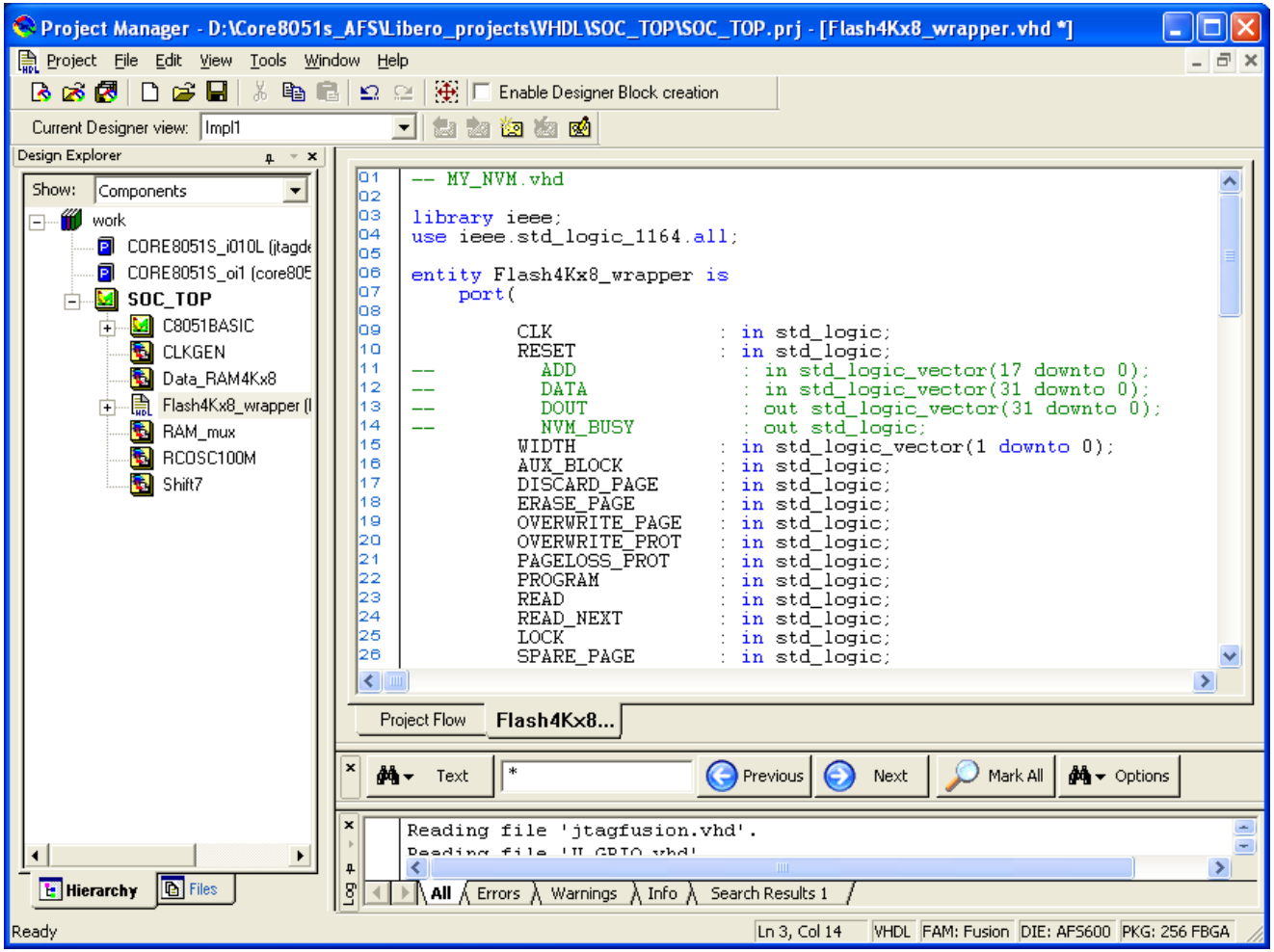
- Comment Command Allows Users to Comment Sections of VHDL or Verilog HDL



Libero IDE Edit Menu

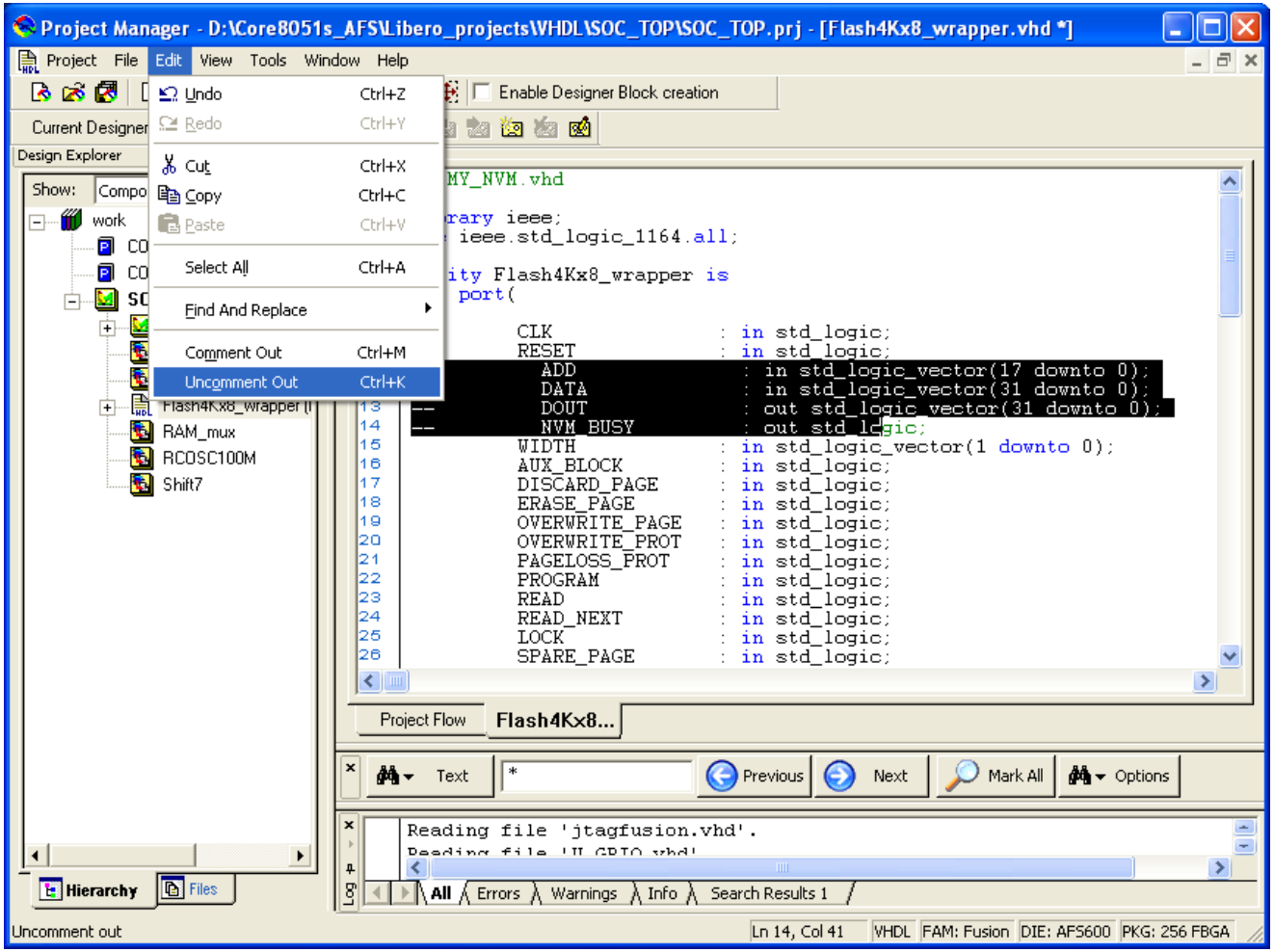
Comment

- Comment Command Allows Users to Comment Sections of VHDL or Verilog HDL



Libero IDE Edit Menu
Uncomment

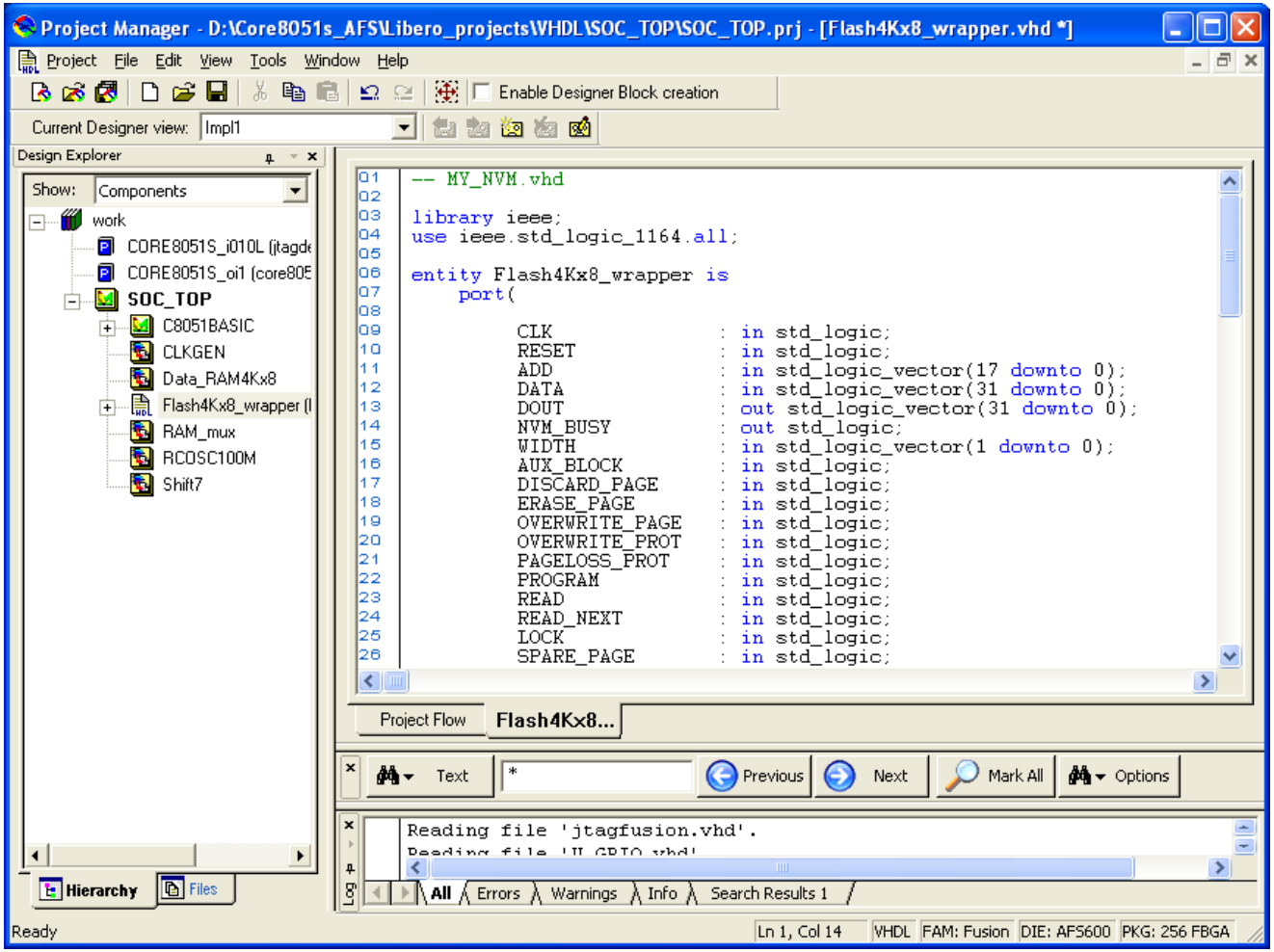
- Uncomment Command Allows Users to Uncomment Sections of VHDL or Verilog HDL



Libero IDE Edit Menu

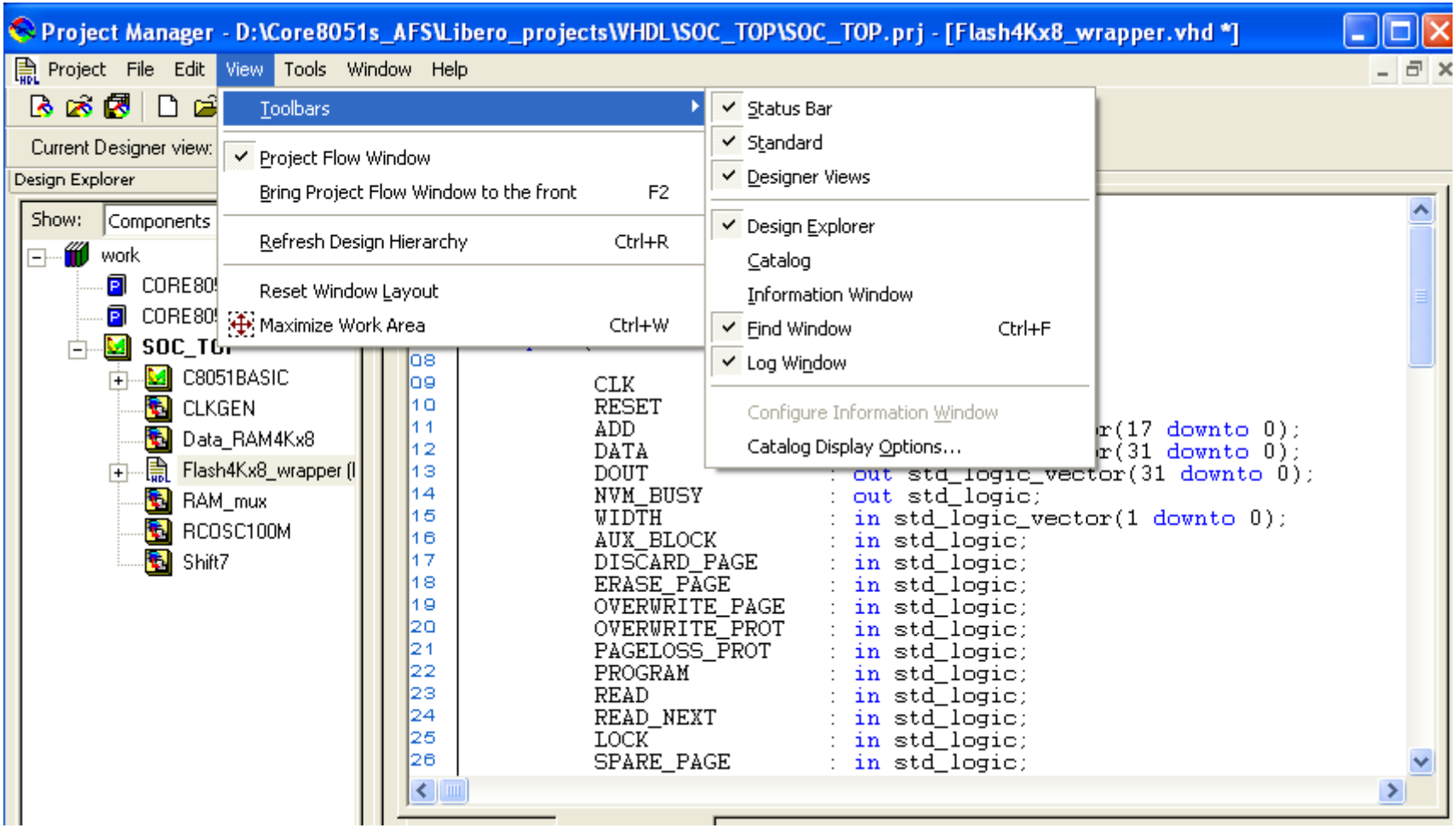
Uncomment

- Uncomment Command Allows Users to Uncomment Sections of VHDL or Verilog HDL



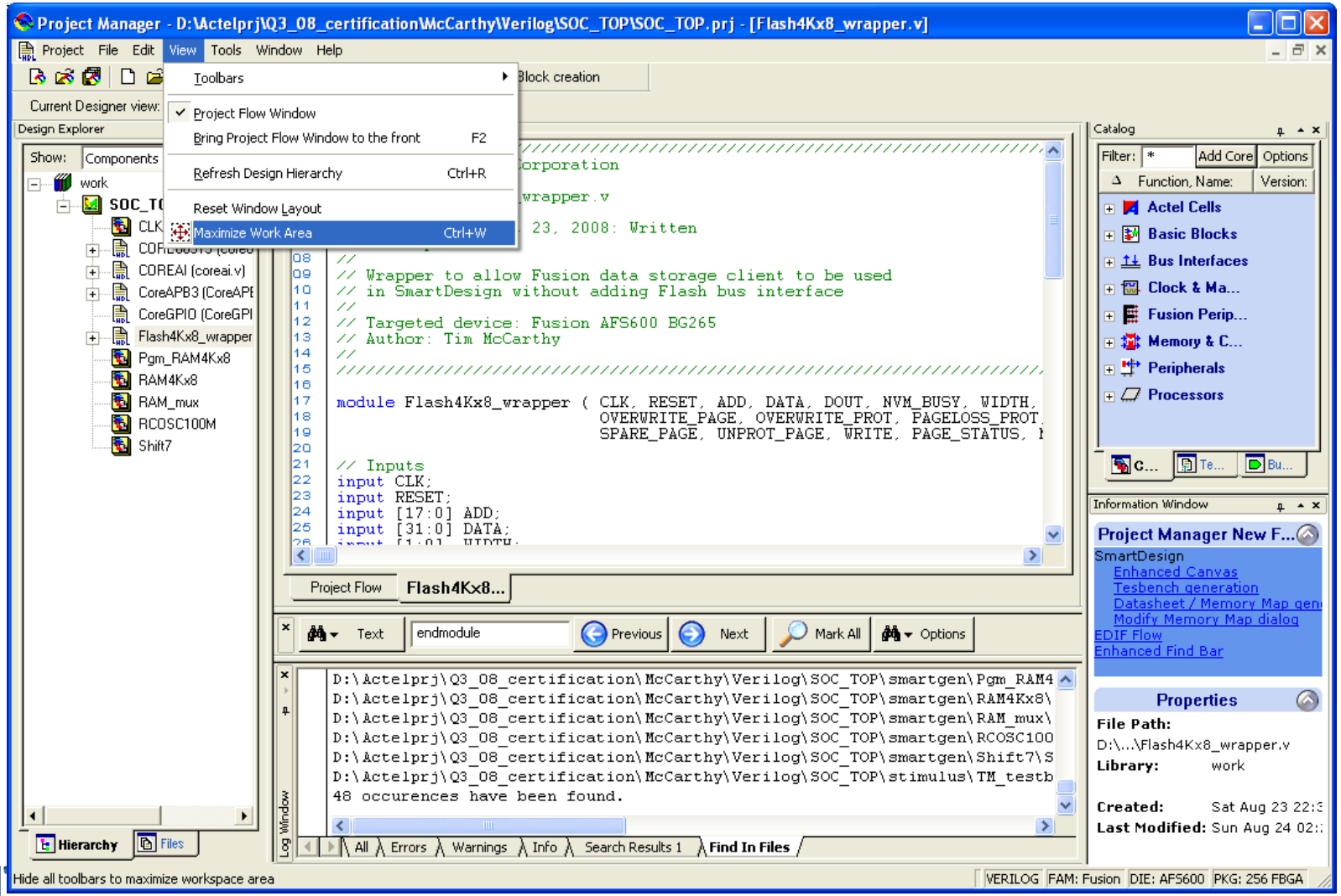
Libero IDE Project Manager
View Options

- Enable / Disable Windows and Docking Bars



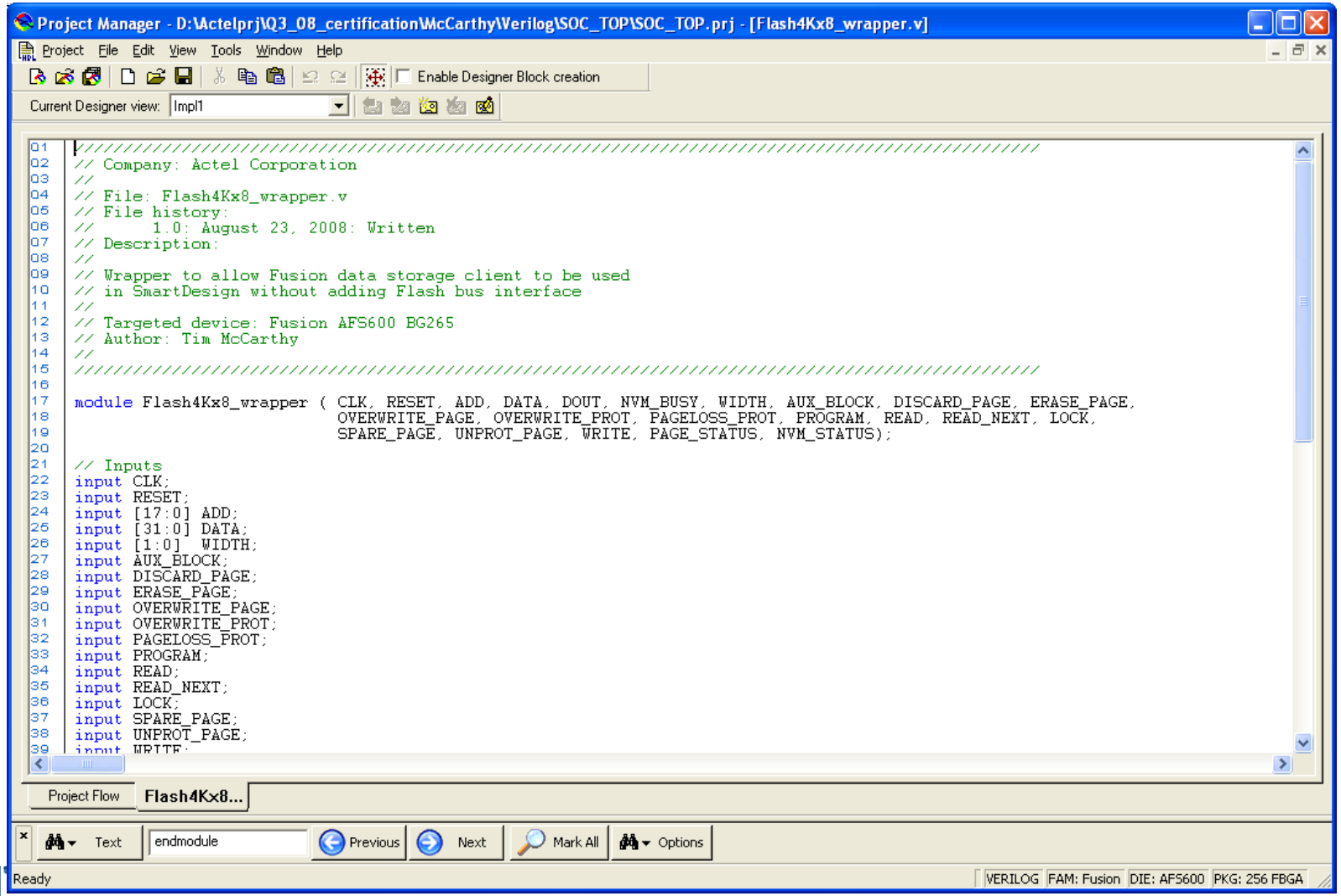
Libero View Options
Maximize Work Area

- Closes all Other Toolbars to Maximize Working Area



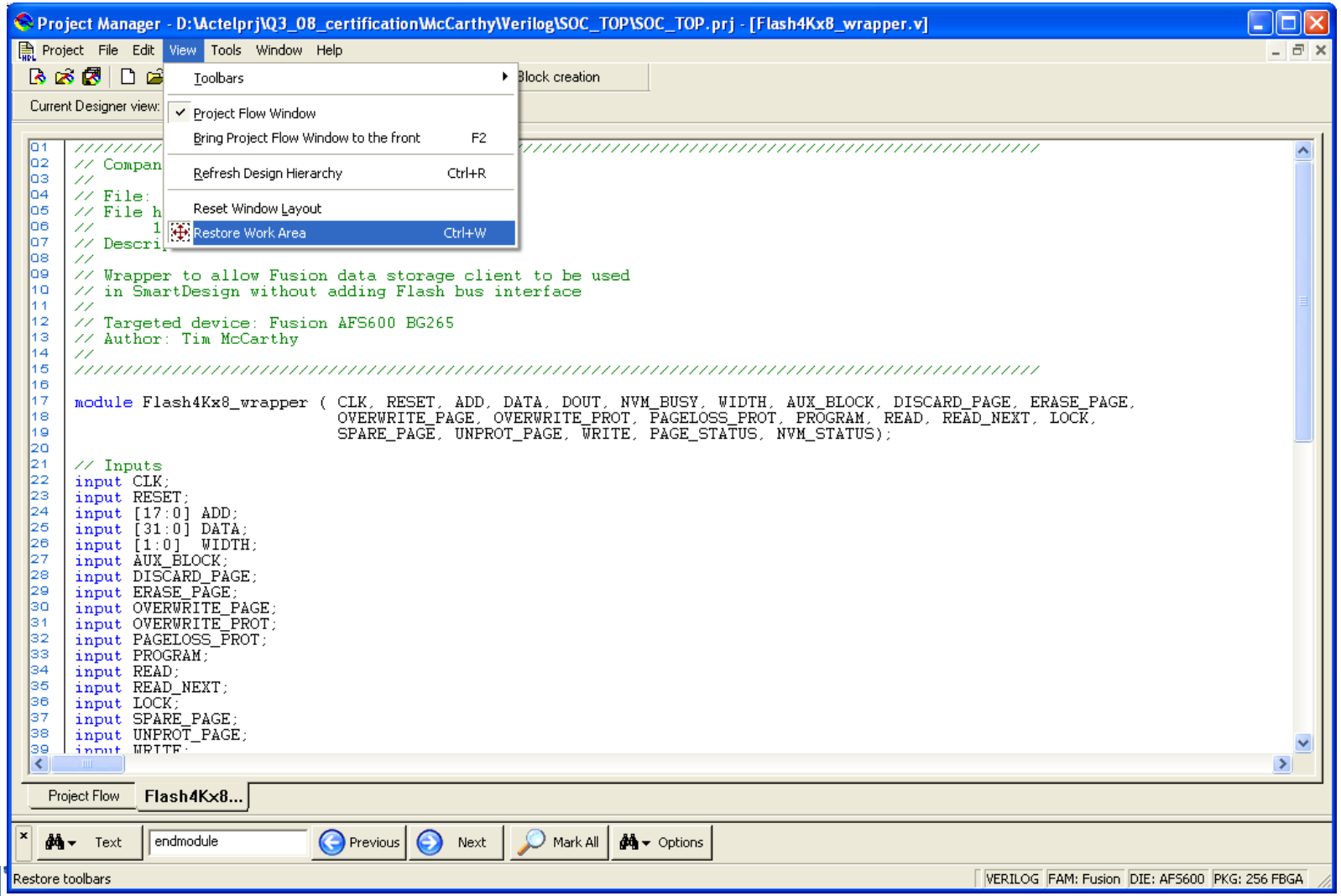
Libero View Options
Maximize Work Area (cont.)

- Editor Maximized



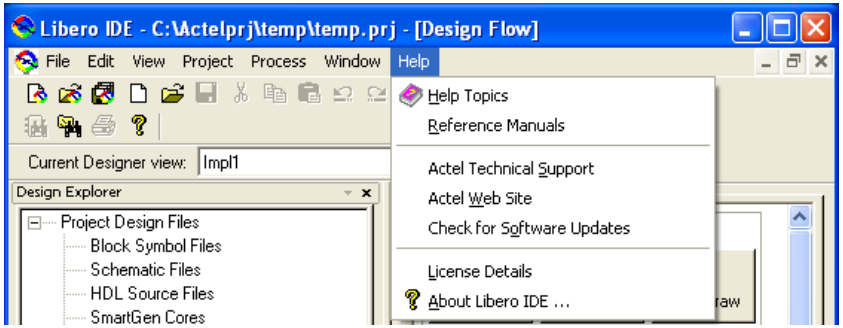
Libero View Options
Restore Work Area

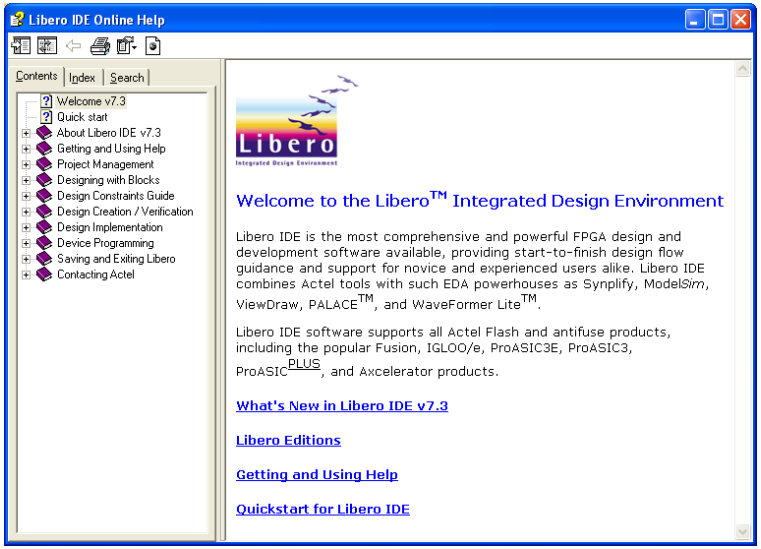
- Restore Toolbars by Selecting “Restore Work Area”



Libero IDE Online Help

- HTML-based Help System
  - Help Available for Error Messages, Specific Screens and Menus
  - Hyperlinks to Application Notes and Actel Web Pages
  - Help Menu Provides Direct Access to All Libero PDF Reference Manuals





Libero IDE Design Flows

Libero IDE Design Flows

- Structural Schematic Flow
  - Contains only Actel ViewDraw Library Components or Mix of Actel ViewDraw Library Components and Structural HDL
  - Top Level *Must* Be Schematic!
  - Synthesis *Optional* before Layout
- Mixed-Mode Flow
  - Schematic and RTL Blocks
    - May also Contain Structural HDL Blocks
  - Top Level *Must* Be Schematic!
  - Synthesis *Required* before Layout
- HDL Flow
  - Mixed VHDL and Verilog Support in Libero 8.5
    - Requires Synplify Pro AE and ModelSim PE/SE
  - May Contain Structural Blocks

SmartGen Macro Builder

SmartGen Macro Builder

- Create Macro Functions from User's Parameters
  - Multiplexors, counters, memory elements, counters, comparators and I/O macros
  - Supports generation and configuration of Fusion peripherals
- Macros are Optimized for Actel Architecture
  - High Speed
  - Small Area
- Rule-based Generation Guarantees Functional Accuracy
- Outputs:
  - Structural VHDL or Verilog (all macros)
  - Behavioral VHDL and Verilog (most macros)

Using SmartGen Macros

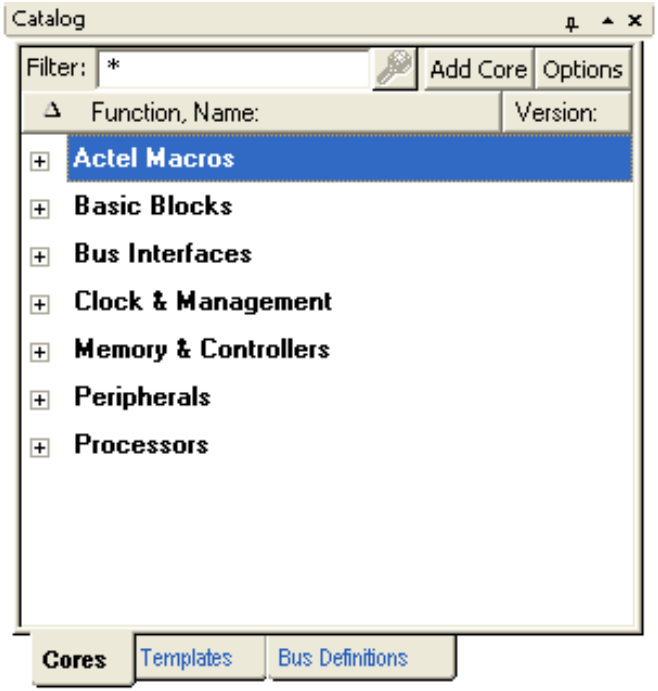
- SmartGen Macros Can Be Used in …
  - … Structural Schematic flow
  - … Mixed-mode Flow
  - … HDL Flow

Using SmartGen within Libero

- Steps:
  - Launch SmartGen from Libero IDE Project Manager
  - Create HDL Structural Implementation
    - VHDL or Verilog
  - HDL Flow
    - Instantiate Macro in Top-level RTL
  - Structural Schematic and Mixed-mode Flows
    - Create ViewDraw Symbol from Libero
    - Instantiate Symbol in Schematic

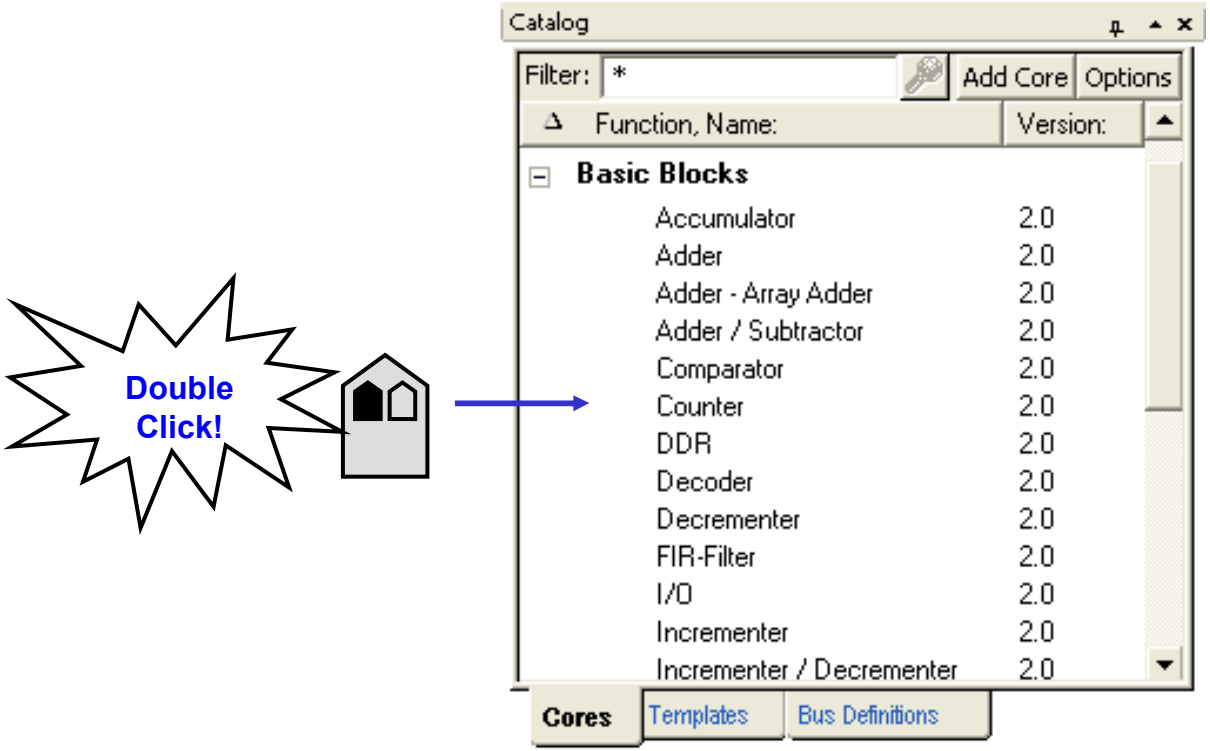
Launching SmartGen from Libero

- SmartGen is Integrated into Libero IDE
  - No Separate SmartGen tool in Libero
  - SmartGen Catalog Embedded in Libero Catalog Windows
  - SmartGen Generated Components Managed by Libero Seamlessly



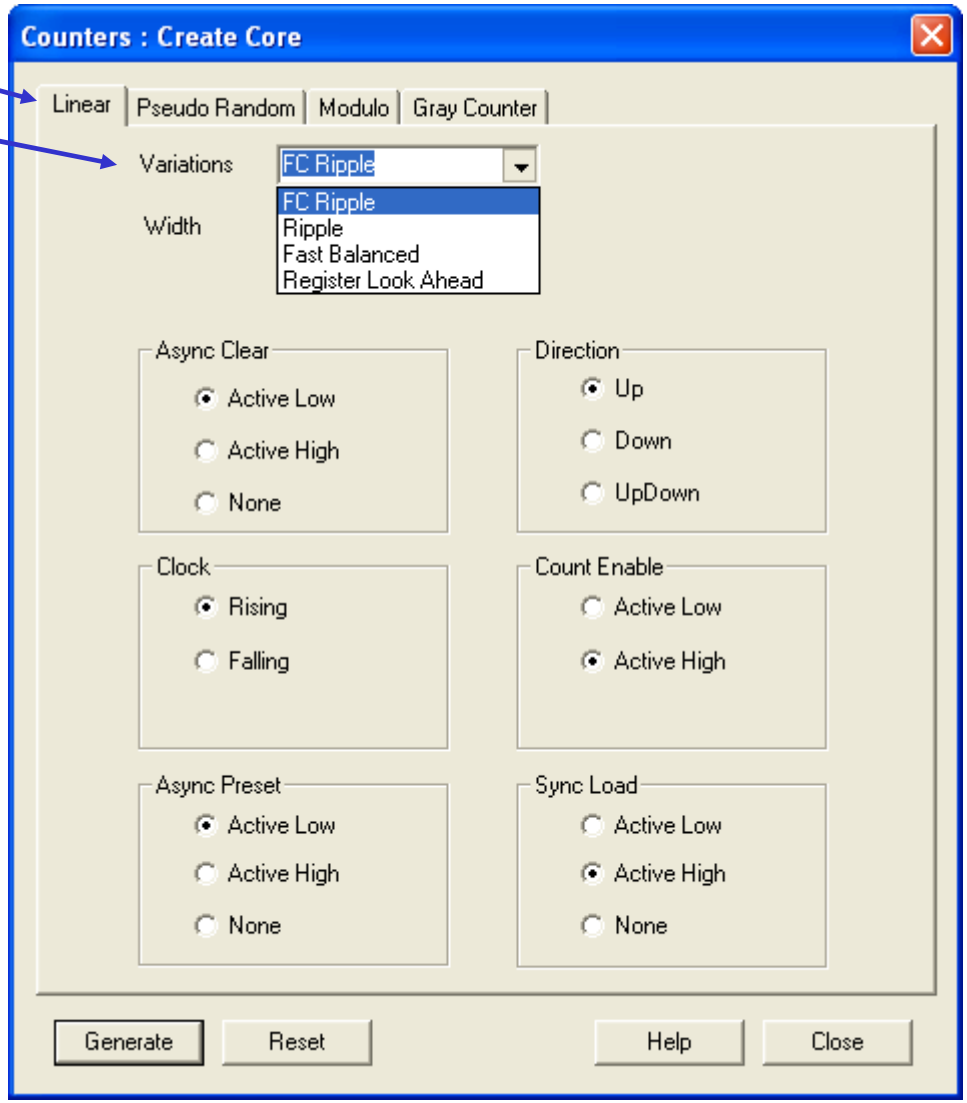
SmartGen
Counter Example

- Choose Function to Open SmartGen Create Core Dialog box



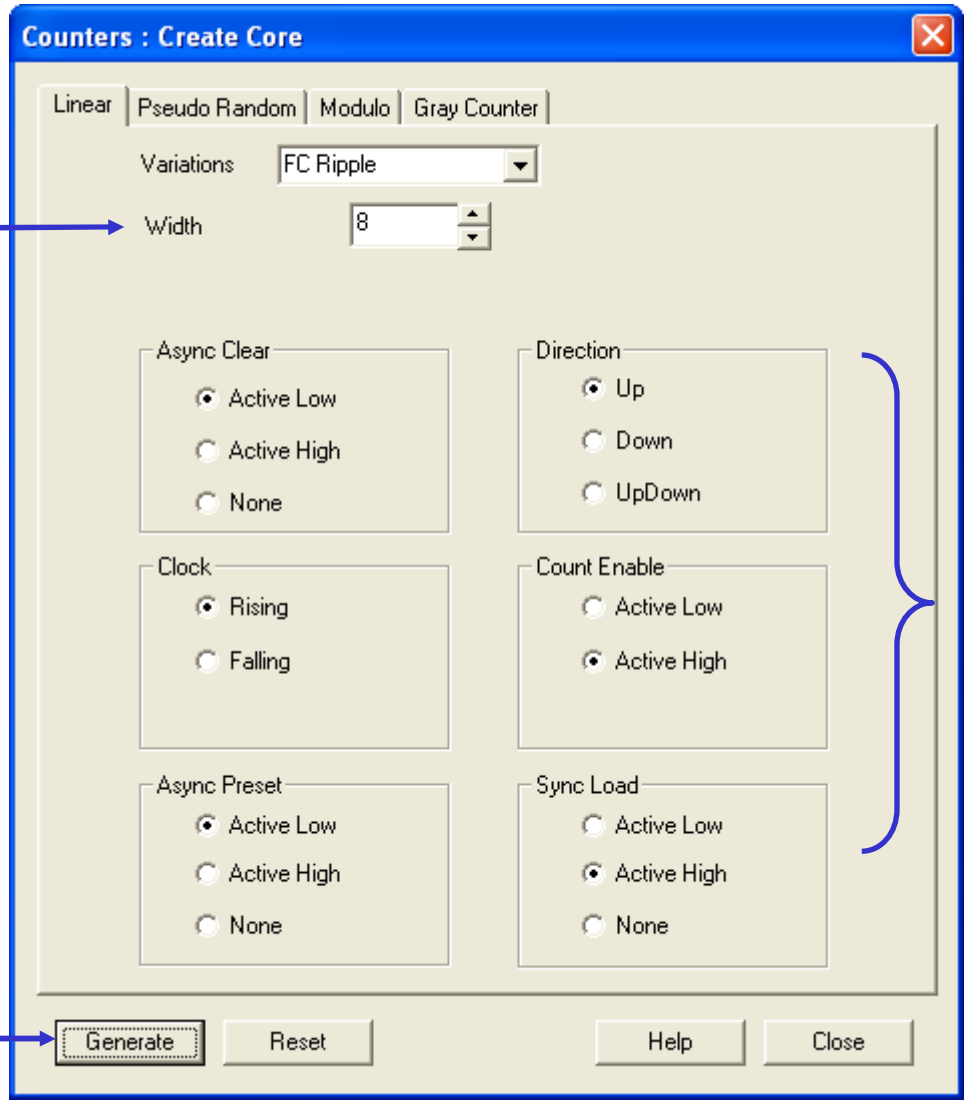
SmartGen
Counter Example

Select type and variation



SmartGen
Counter Example

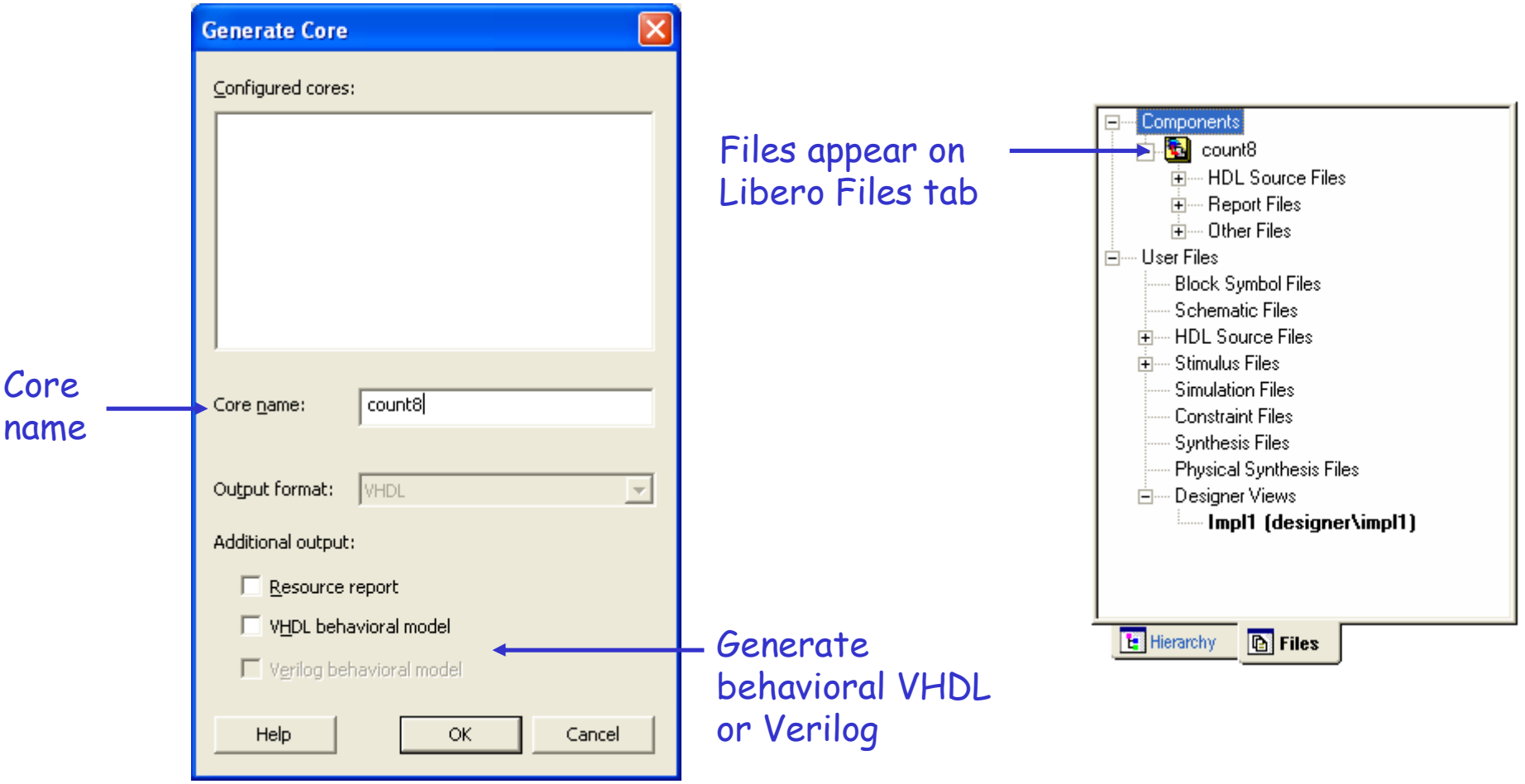
Enter width



Complete the rest of the description

Click Generate

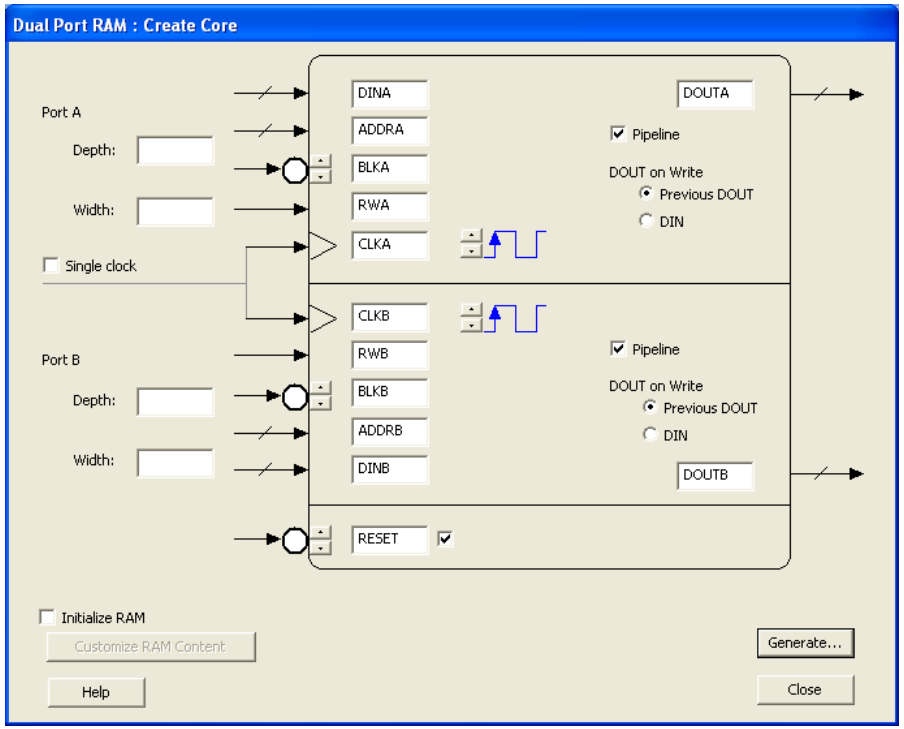
SmartGen Component Generation



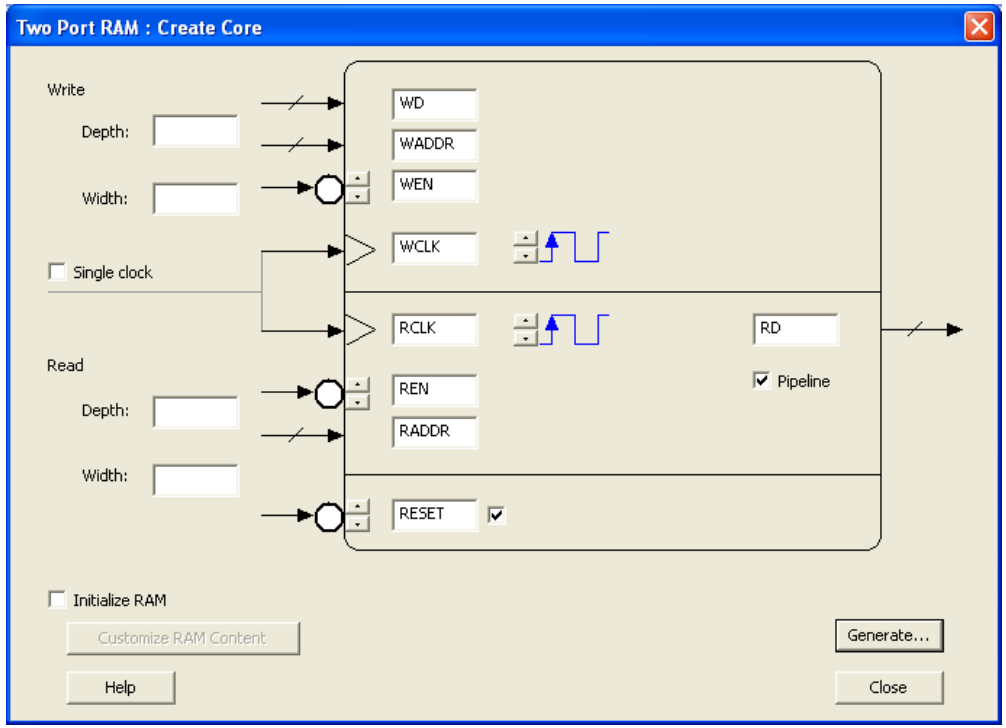
SmartGen Family Specific Features:
Fusion / ProASIC3 / Axcelerator

- BusLVDS Macro
  - Available for Fusion and ProASIC3
- Visual Configurator for Two-Port and Dual Port RAMs
  - Available for Fusion / ProASIC3 and Axcelerator
- Soft FIFO Controller
  - Available for Fusion / ProASIC3 and Axcelerator
  - Create Controller with or without Memory
- Dynamic Clock Conditioning Circuitry (CCC) Support
  - Dynamically change the CCC configuration by entering the control data serially
  - ProASIC3 and Fusion

SmartGen RAM Visual Configurator

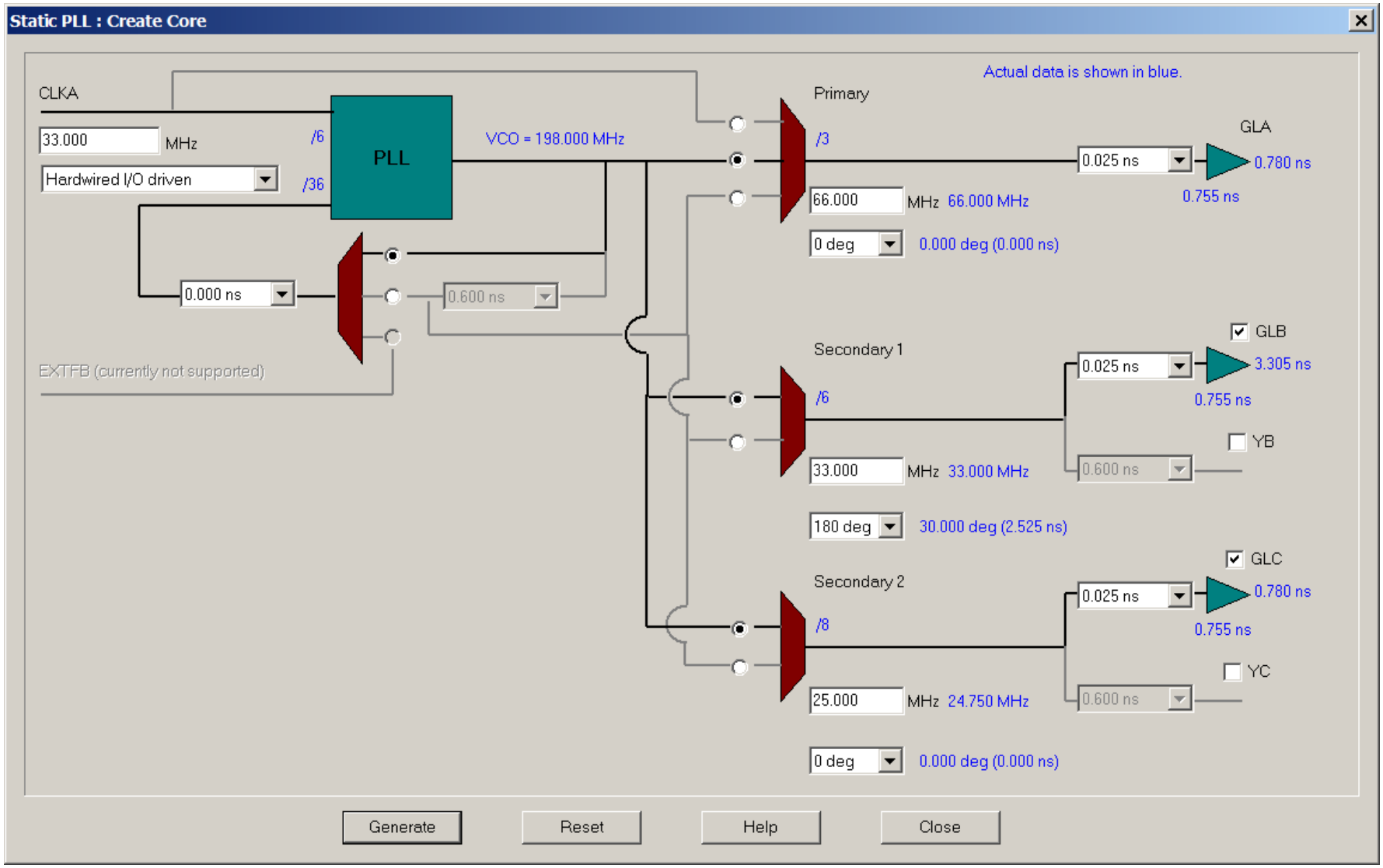


Dual-Port RAM Configurator



Two-Port RAM Configurator

SmartGen ProASIC3 PLL Configurator

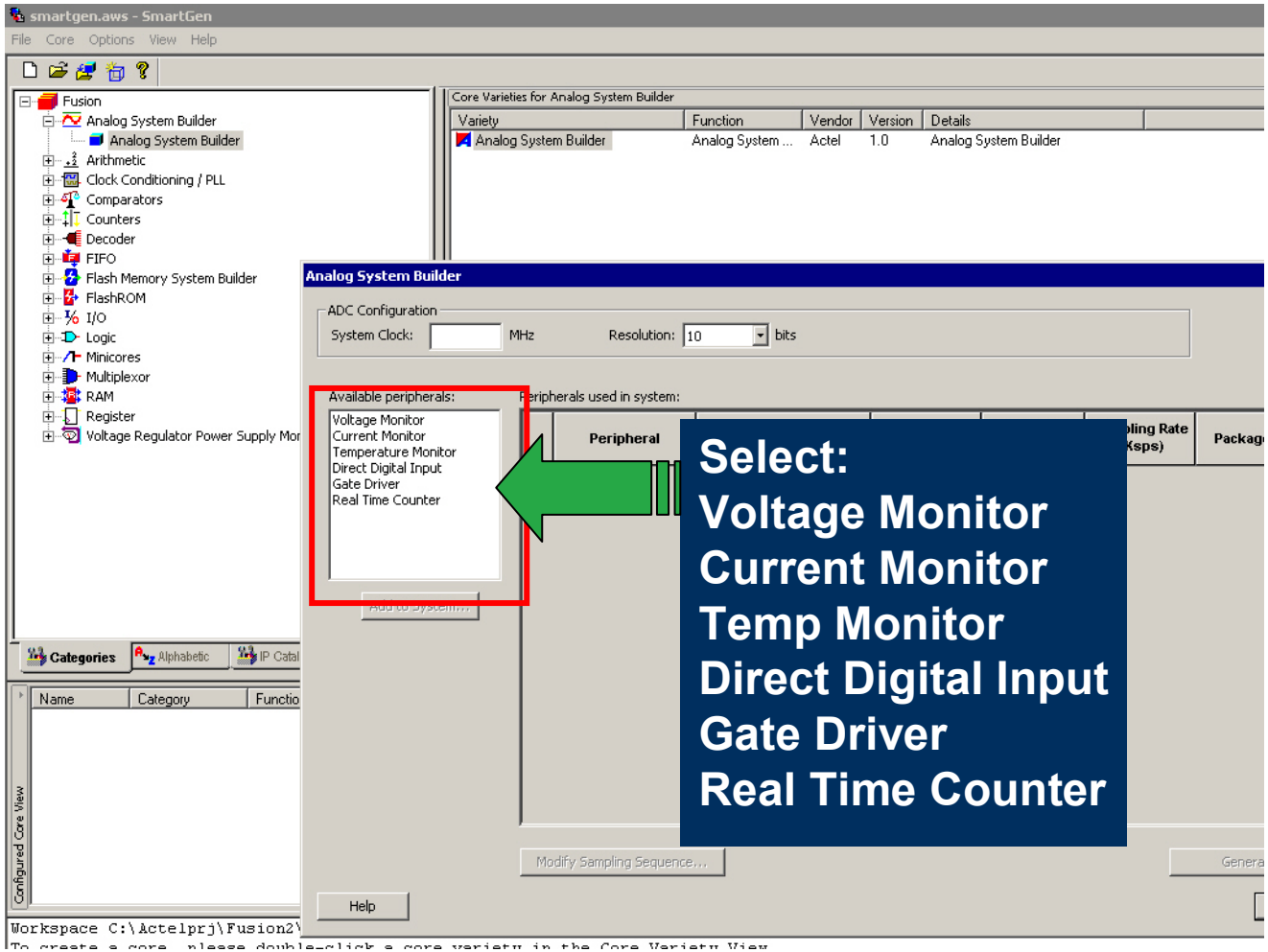


SmartGen Fusion Peripheral Support

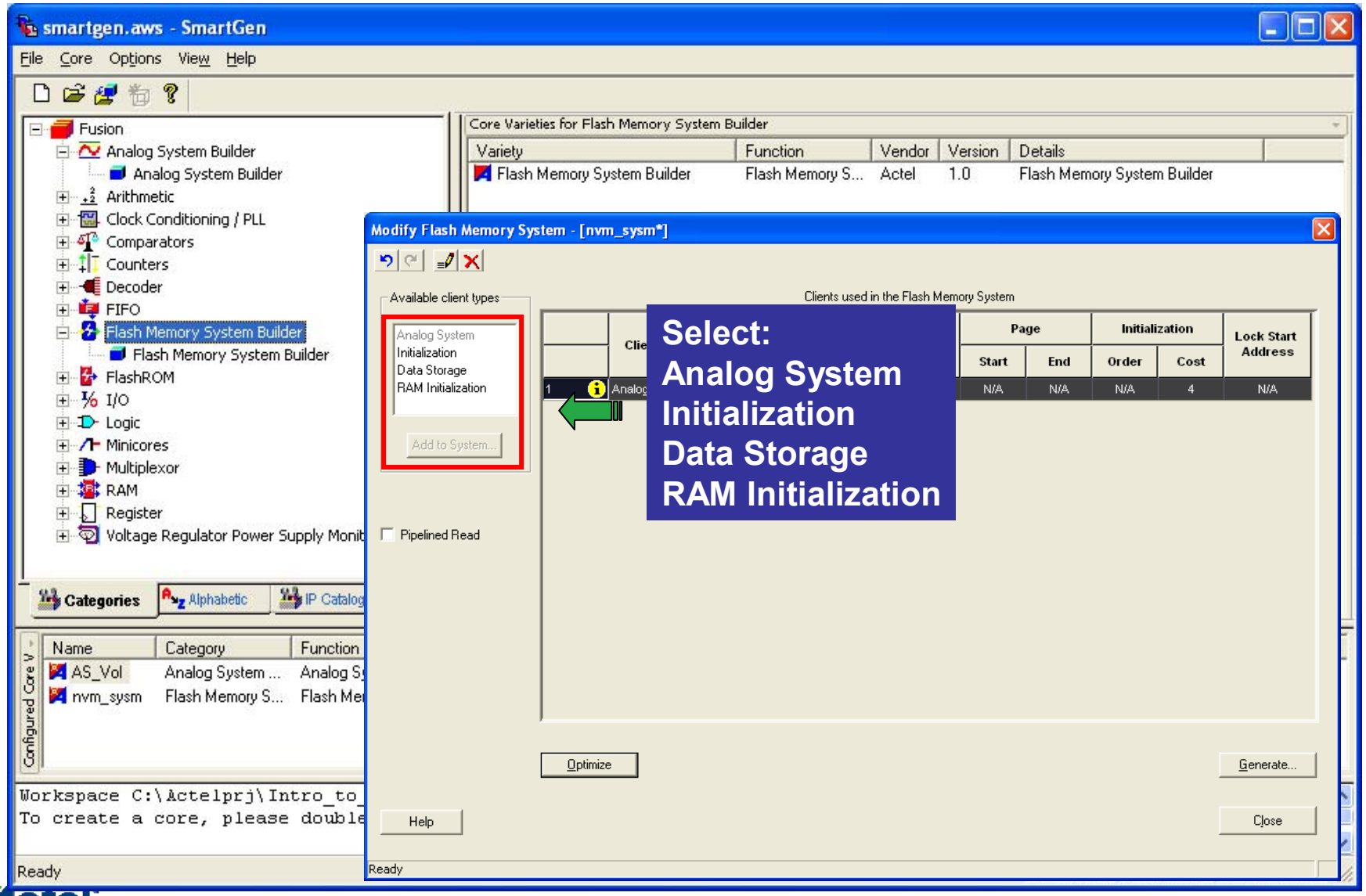
- SmartGen Includes “System Builders” For Actel Fusion Peripherals
  - Analog Peripherals
  - Flash Memory Blocks
  - FlashROM
  - Easy Customization via Visual Configurators

SmartGen Fusion Analog System Builder

- Fusion “Analog System Builder” category selection and configuration
- Intuitive menu for peripheral selection and configuration



SmartGen Fusion Flash Memory System Builder

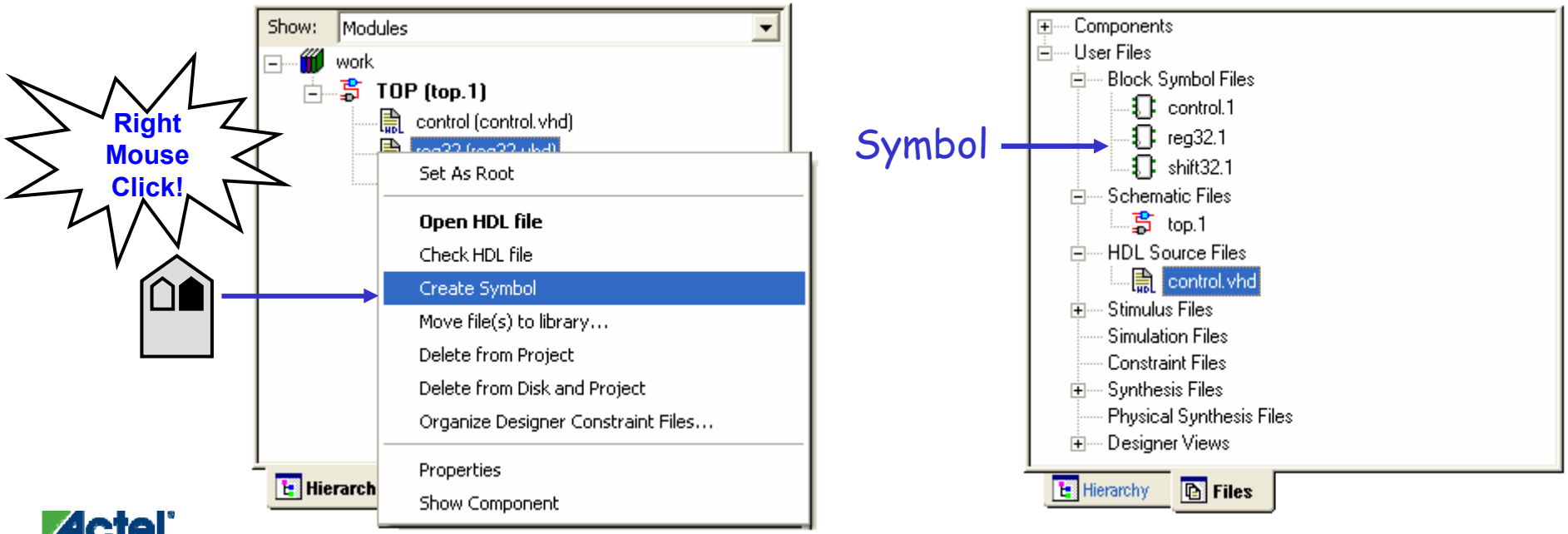


Structural and Mixed-Mode Flow
Using RTL and SmartGen Macros

- Create RTL from Libero HDL Editor or Import File

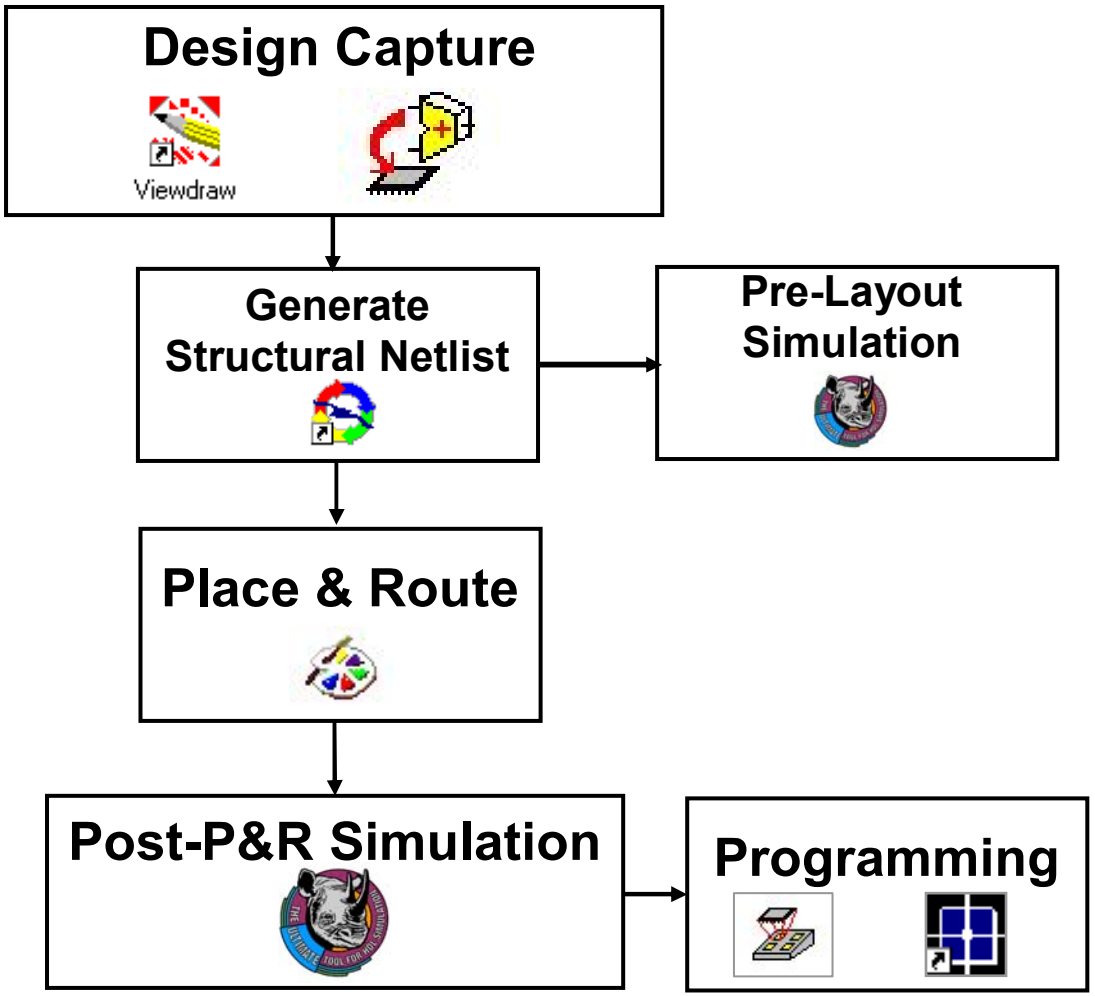
  OR

- Create HDL Structural Implementation using SmartGen
  - VHDL or Verilog
- Create ViewDraw Symbol from Libero and Instantiate Symbol in Schematic
  - Symbol Appears on Files Tab



Structural Schematic Designs

Structural Schematic Design Flow

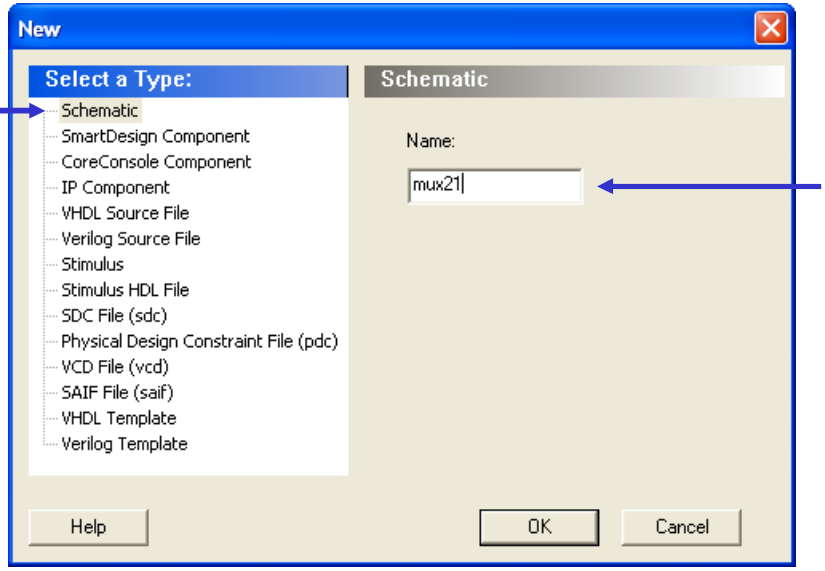


ViewDraw Overview

Opening ViewDraw

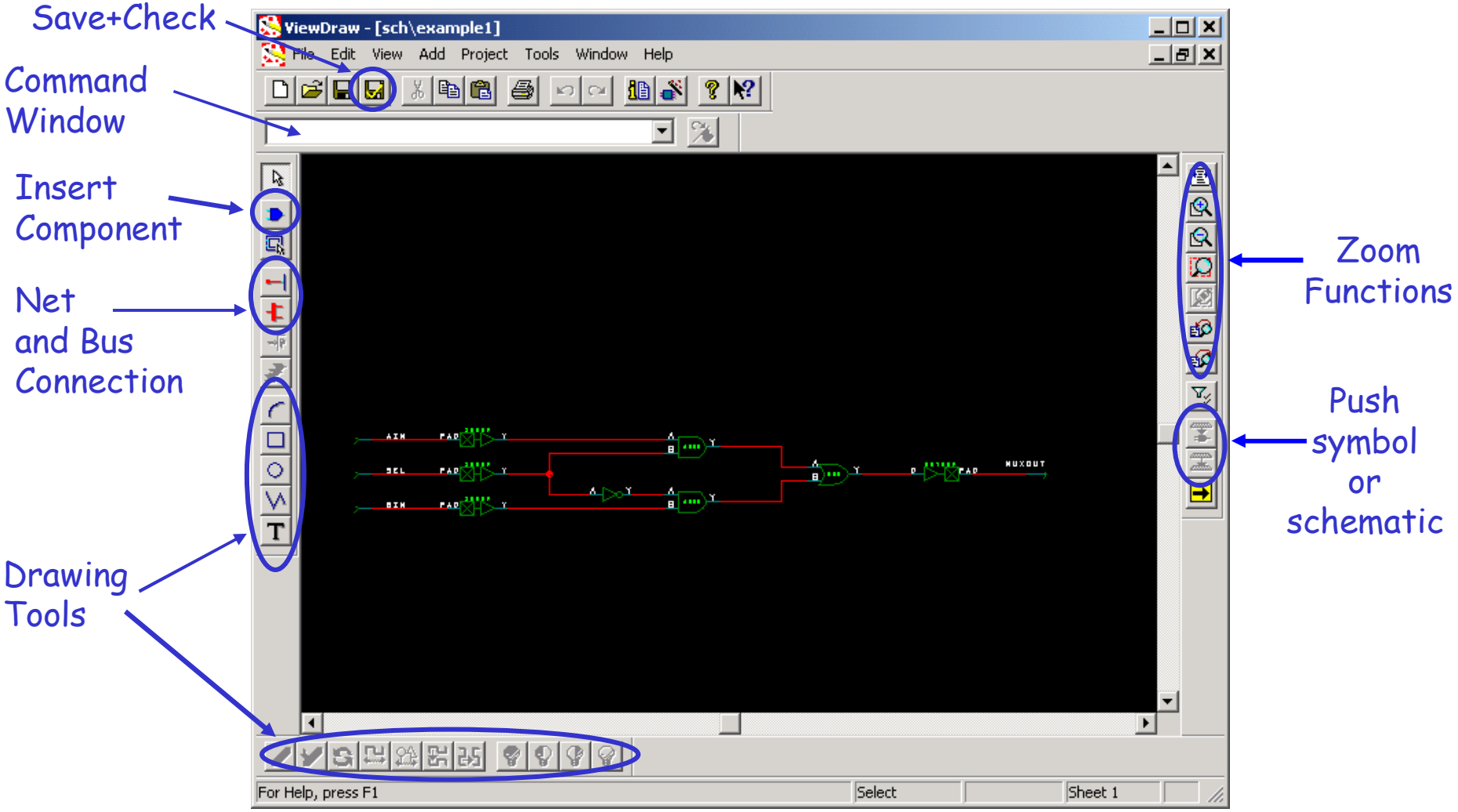
- Click ViewDraw Button in Project Flow Window
- Create Schematic
- Save and Check

Select schematic



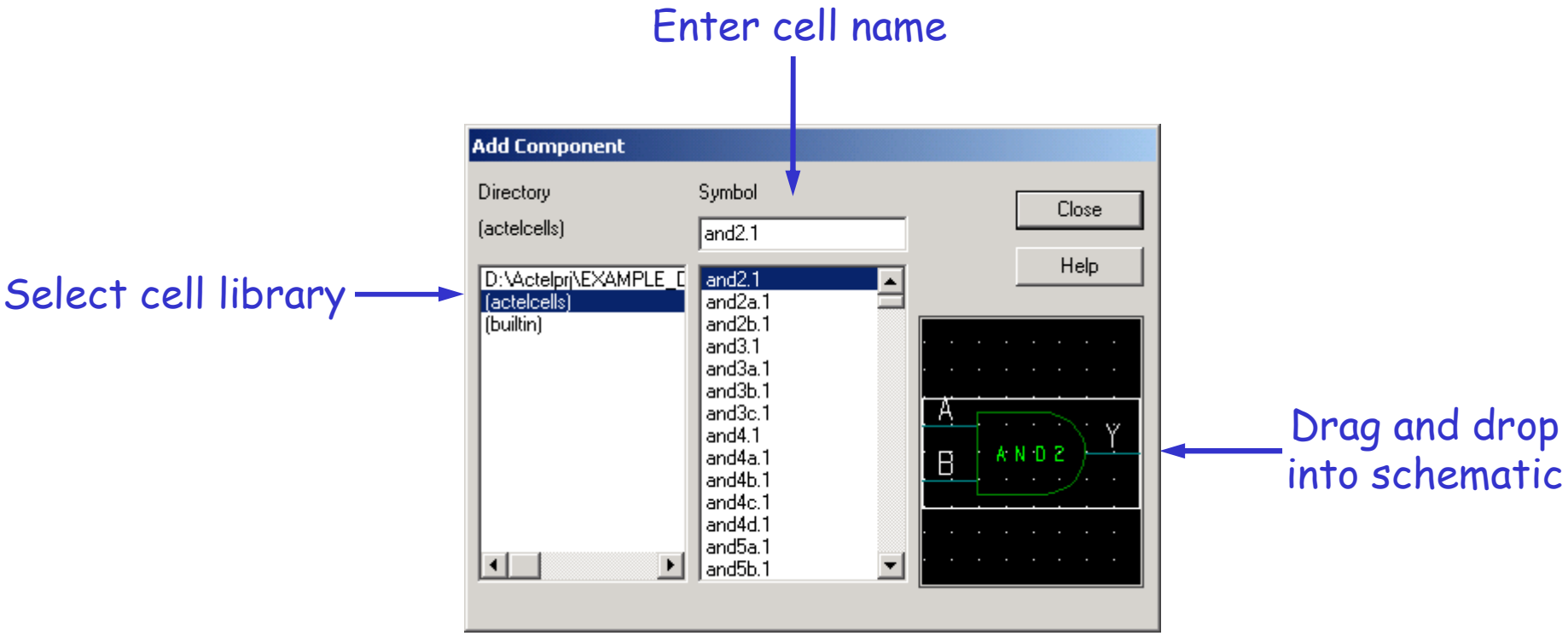
Enter schematic name

ViewDraw GUI



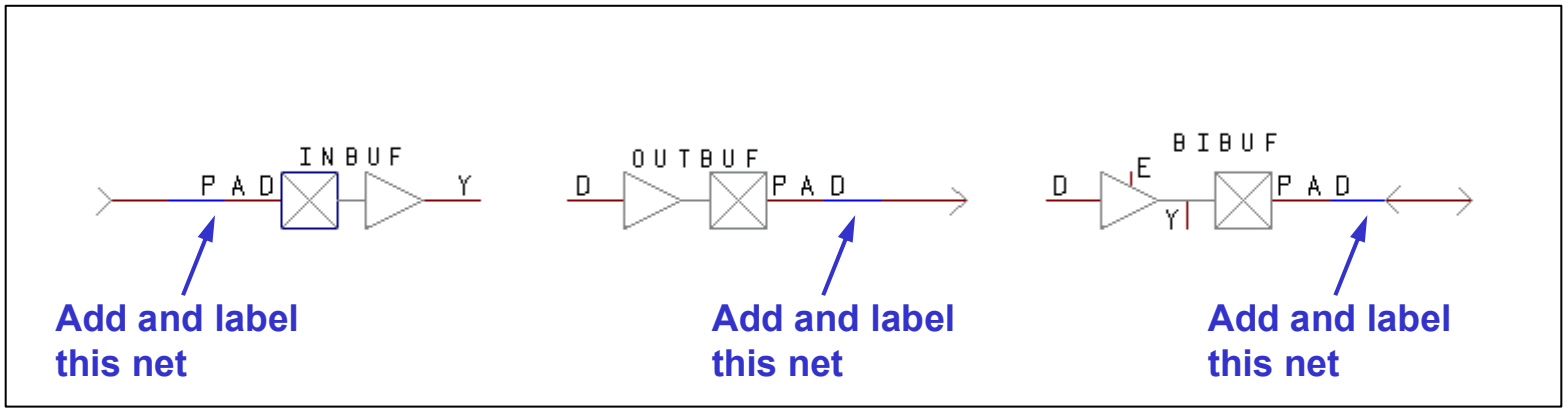
Adding Schematic Components

- Add > Component from ViewDraw Menu
  - Add SmartGen Macros, Custom Macros or Actel Basic Cells
    - Select VCC or GND from ‘actelcells’



Adding I/O Cells

- Add I/O Cells to Top-level Design Schematic
  - Schematic-only Designs or Structural Schematic Designs
  - Macros Contained in “actelcells” Component Library
- I/O Cells Must Have Dangling Hierarchical Connector Attached to Pad Side
  - Label Dangling Connector
- I/O Macros Can Be Buried in Hierarchy

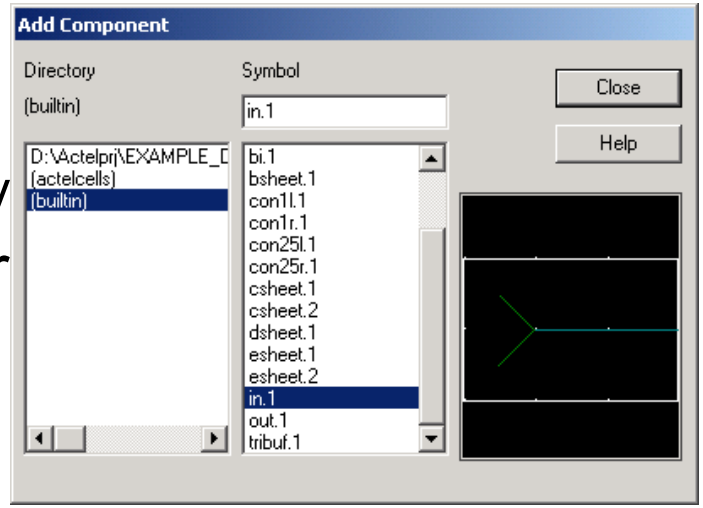


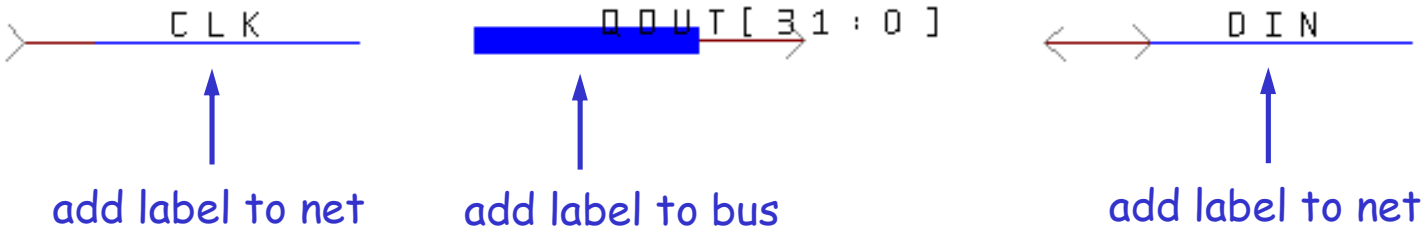
Drawing Wires and Busses
Adding a Net

- To Add Net:
  - Choose Add > Net (or Add > Bus)
    - Alternate: Click Wire () or Bus () Icon on Toolbar
  - Specify Net Origination Point and Depress Left Mouse Button
  - Drag Mouse to Form Net (or Bus), specifying Points along Net by Clicking Right Mouse Button
  - Click Right Mouse Button to Insert Vertex in Net
  - Release Left Mouse Button to Specify Ending Point for Net

Hierarchical Connectors

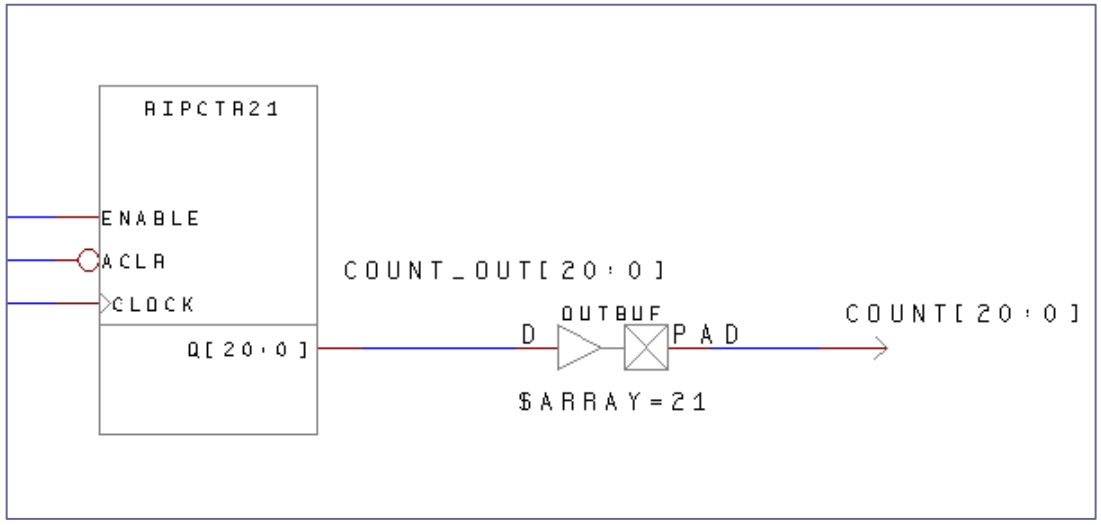
- Use Hierarchical Connectors from ViewDraw Built-in Library for All Designs
  - Add just like Any Other Component
  - Same Connector for Wire or Bus
  - Called ‘in’, ‘out’, or ‘bi’ in *Built-in* Library
- Label Net or Bus Next to Connector

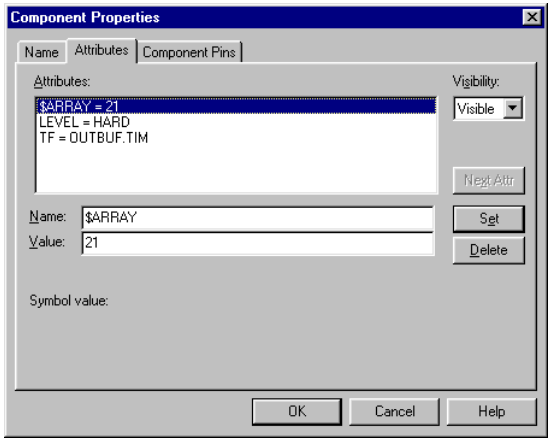




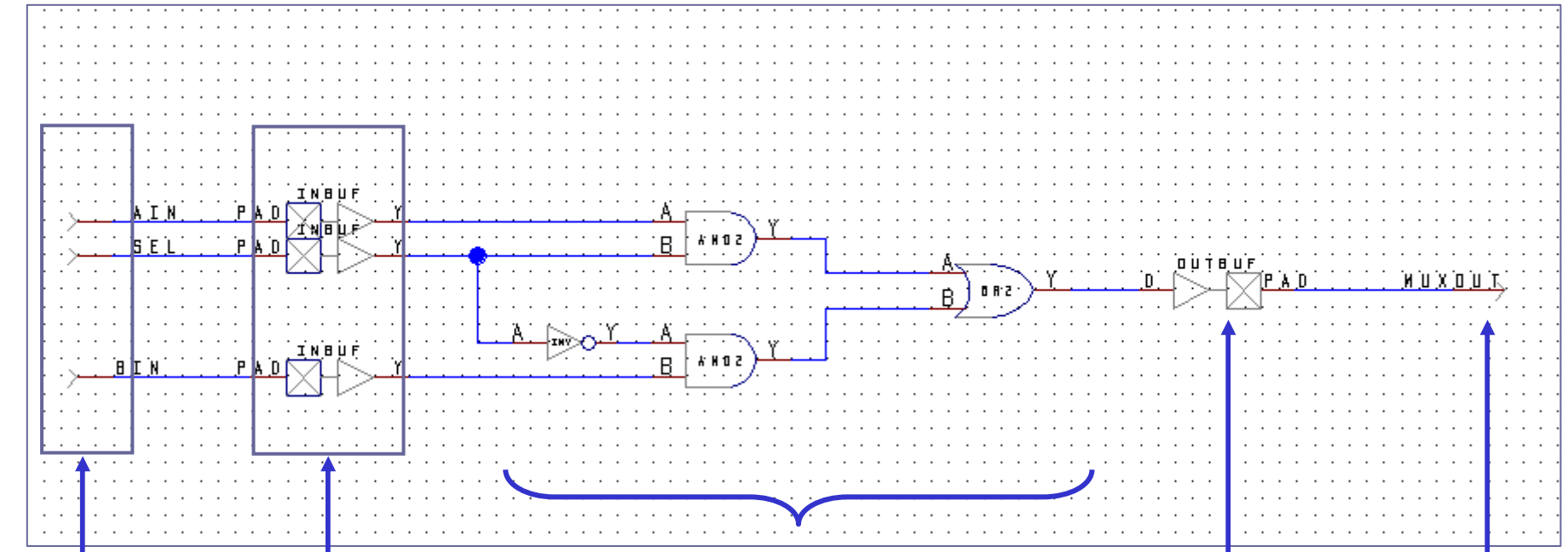
ViewDraw Attributes

- A *Limited* Number of Attributes Can Be Entered into Schematic and Passed to Designer
- $Array Attribute
  - Creates Arrays of Cells in Schematic
    - Useful for I/O Buffers
  - Double-click Cell, Enter on Attribute Tab





Completed Schematic



Hierarchical connectors

Input buffers

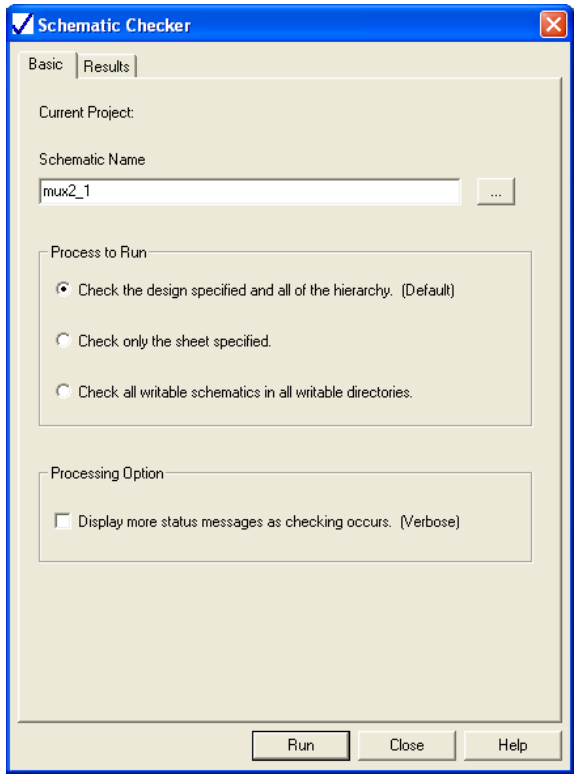
Actel library components

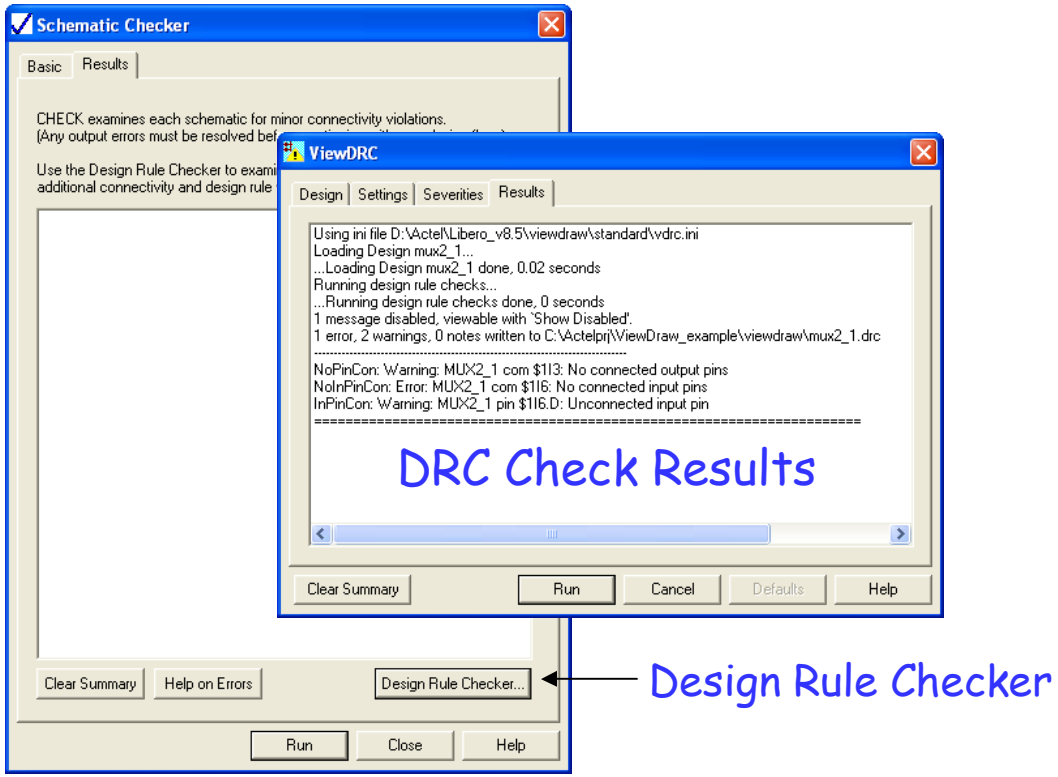
Output buffer

Hierarchical connector

Design Rule Checking

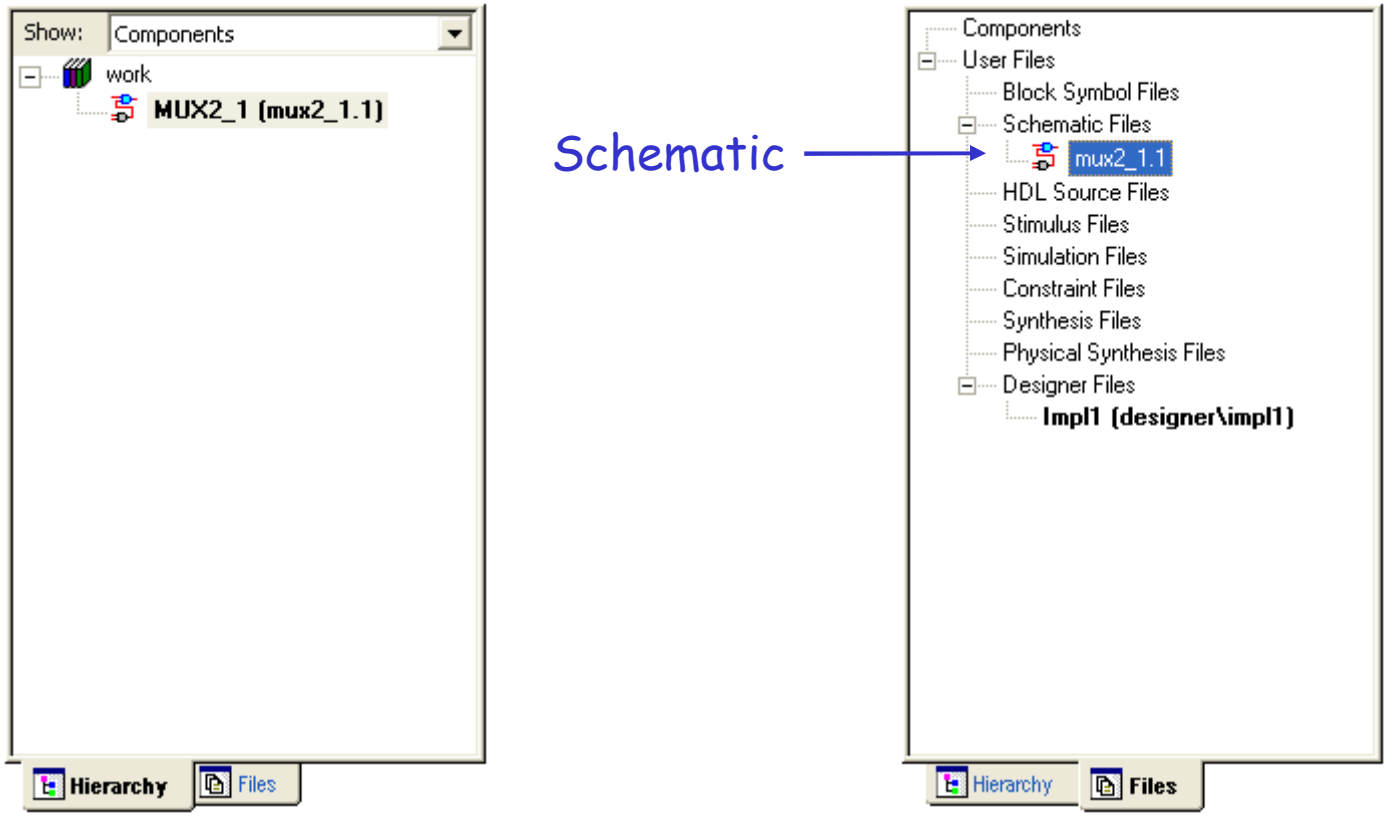
- At Design Entry Completion , Save and Check Design
  - Click Save Check Icon
  - Use Tools > Schematic Checker





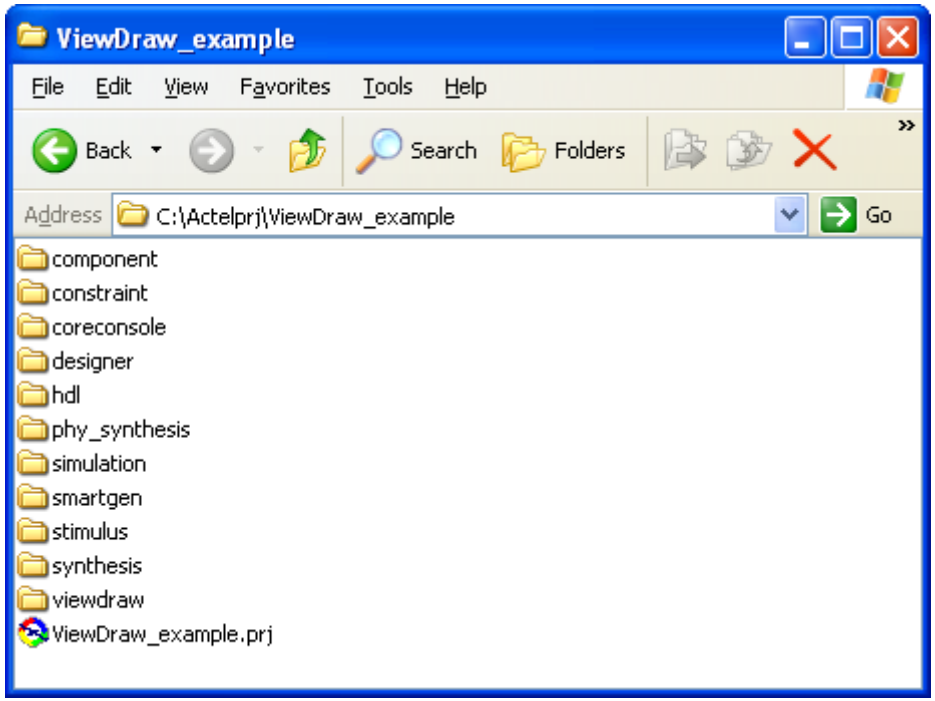
Design Entry Completion

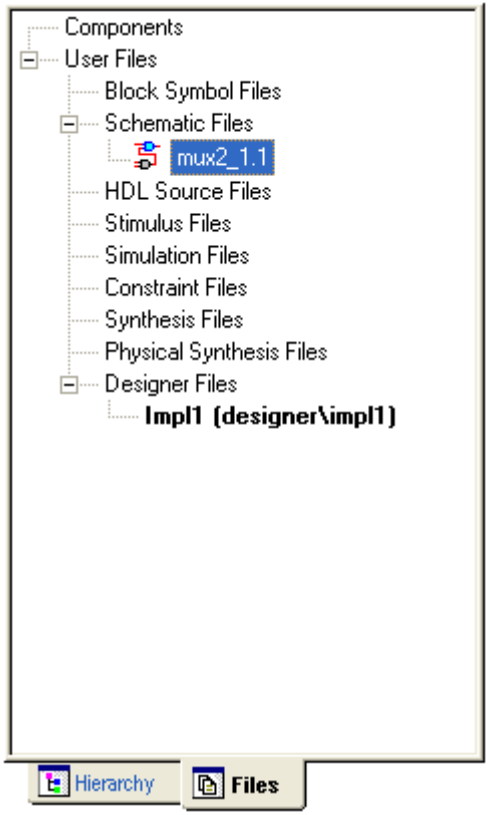
- Files in Implementation are Displayed on Libero Hierarchy and Files Tabs



ViewDraw File Structure on HDD

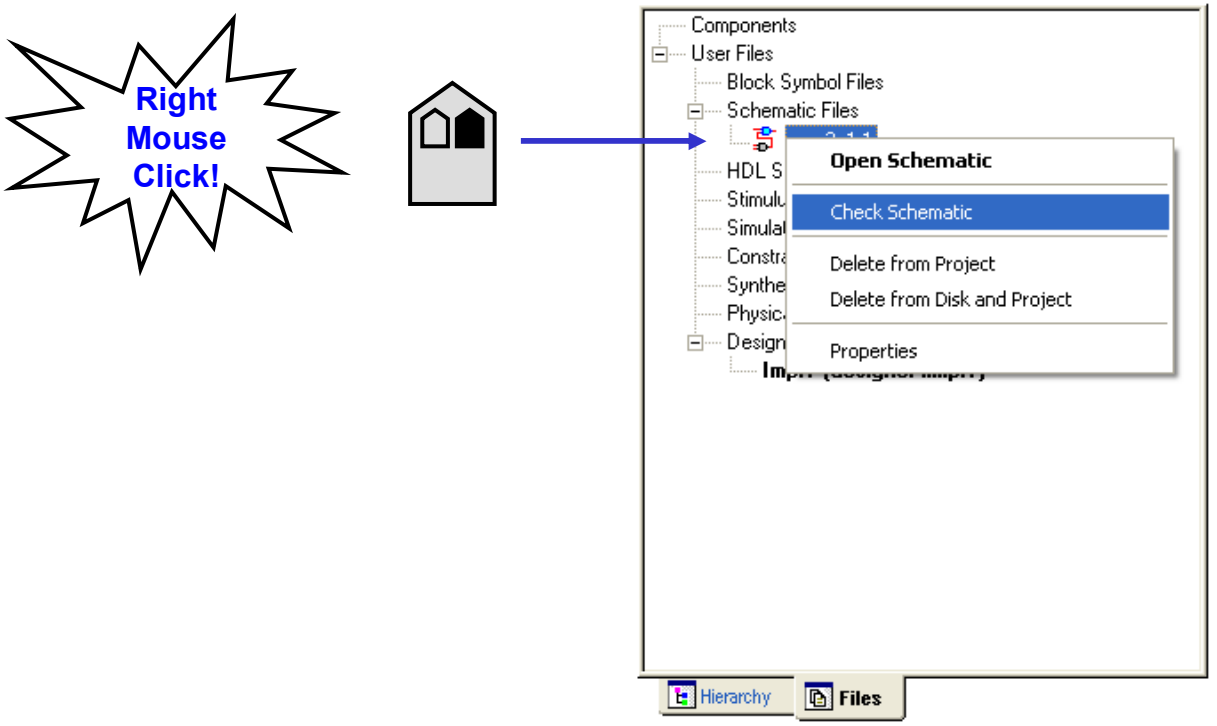
- Schematic Files Saved in “sch” Folder
- Symbol Files Saved in “sym” Folder
- Wire Files Saved in “wir” Folder
- Files Visible on Libero File Manager Tab



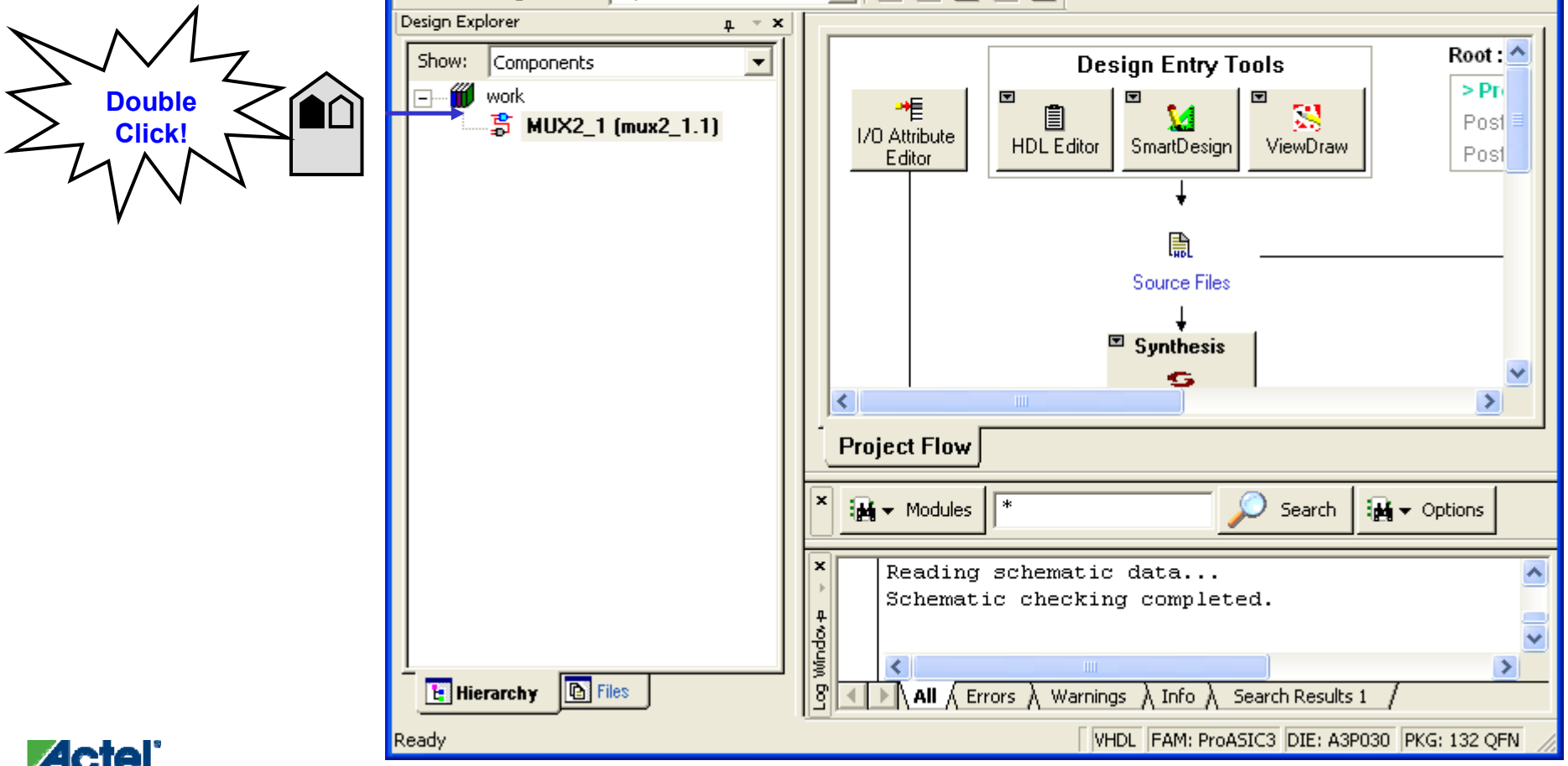


Libero Schematic Checker (Optional)

- Schematic Connectivity Checker in Libero
  - Checks for Errors Not Included in ViewDraw Save + Check
  - Optional Step Available from File Manager Tab

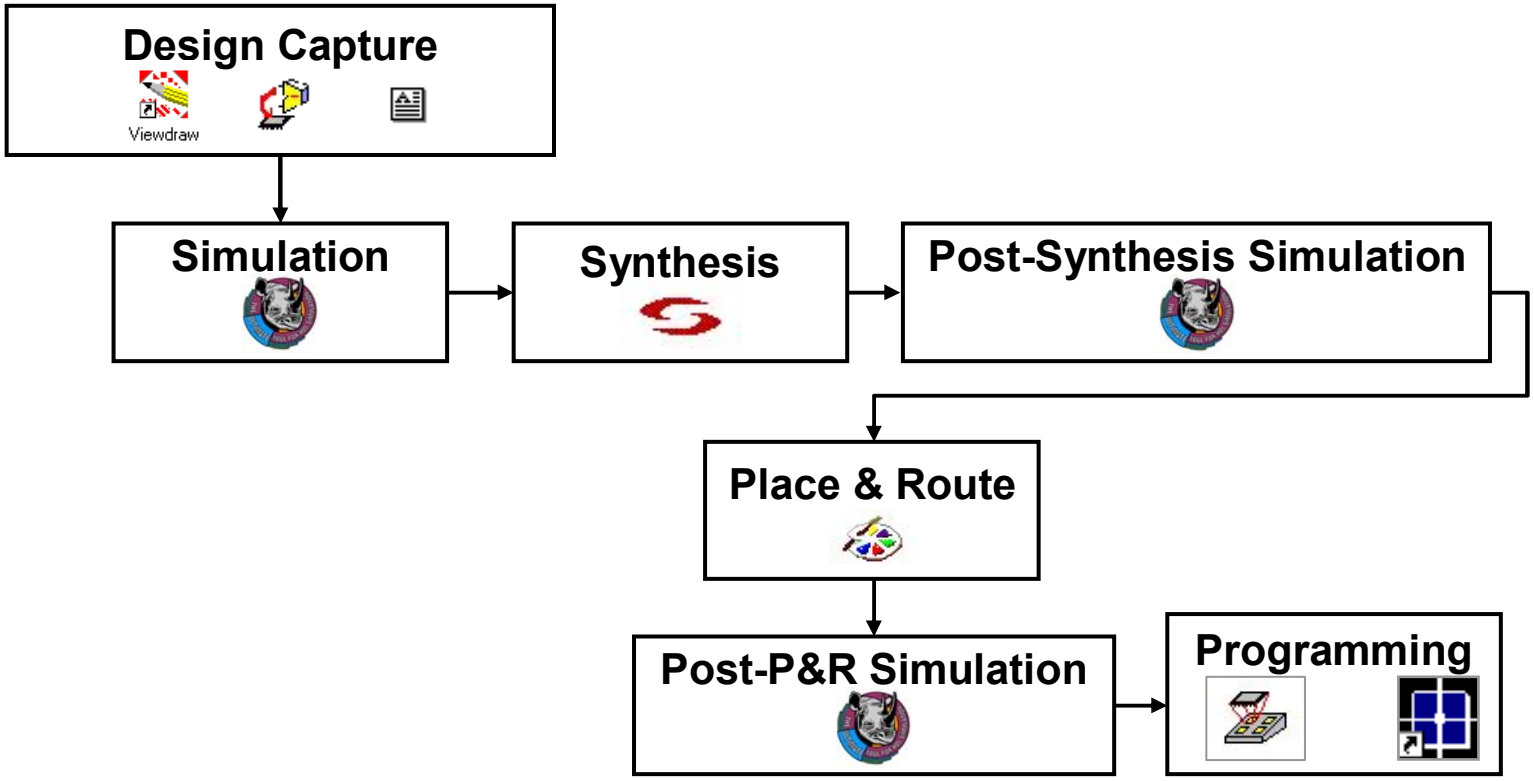


Opening Existing Schematics



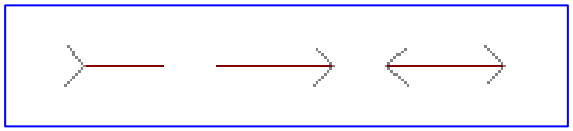
Mixed-Mode Designs

Mixed-Mode Design Flow



Mixed Mode Design Entry

- Mixed Mode => RTL Blocks within Schematic
  - HDL Blocks Can Be Structural or Behavioral RTL
    - Libero IDE Supports Mixed VHDL and Verilog Source Files
  - Top Level *Must* Be Schematic
- Procedure
  - Create HDL Blocks
    - RTL Blocks - Use HDL Editor or Import Existing Design Files
    - Structural Blocks - Use HDL Editor or SmartGen
  - Create ViewDraw Symbols for HDL Blocks
    - Done Automatically from Libero
  - Instantiate Blocks in Schematics and Make Interconnects
    - Use Hierarchical Connectors from ViewDraw “built-in” Library for HDL Ports in Schematic

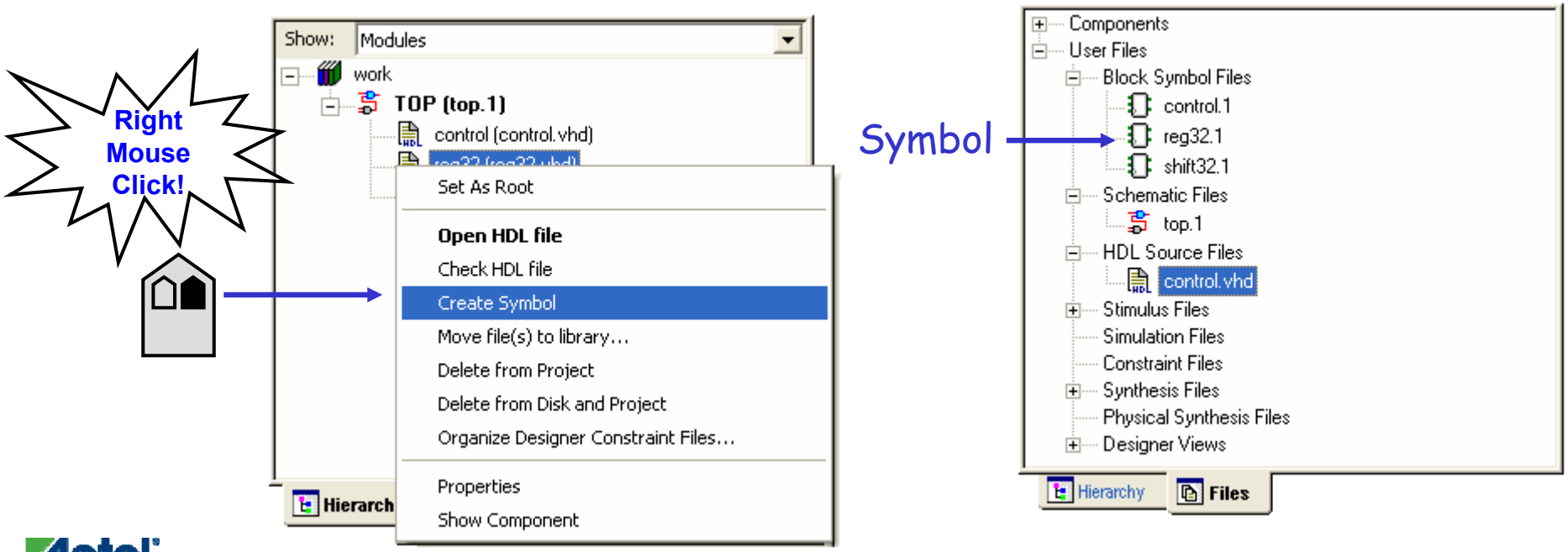


Structural and Mixed-Mode Flow
Using RTL and SmartGen Macros

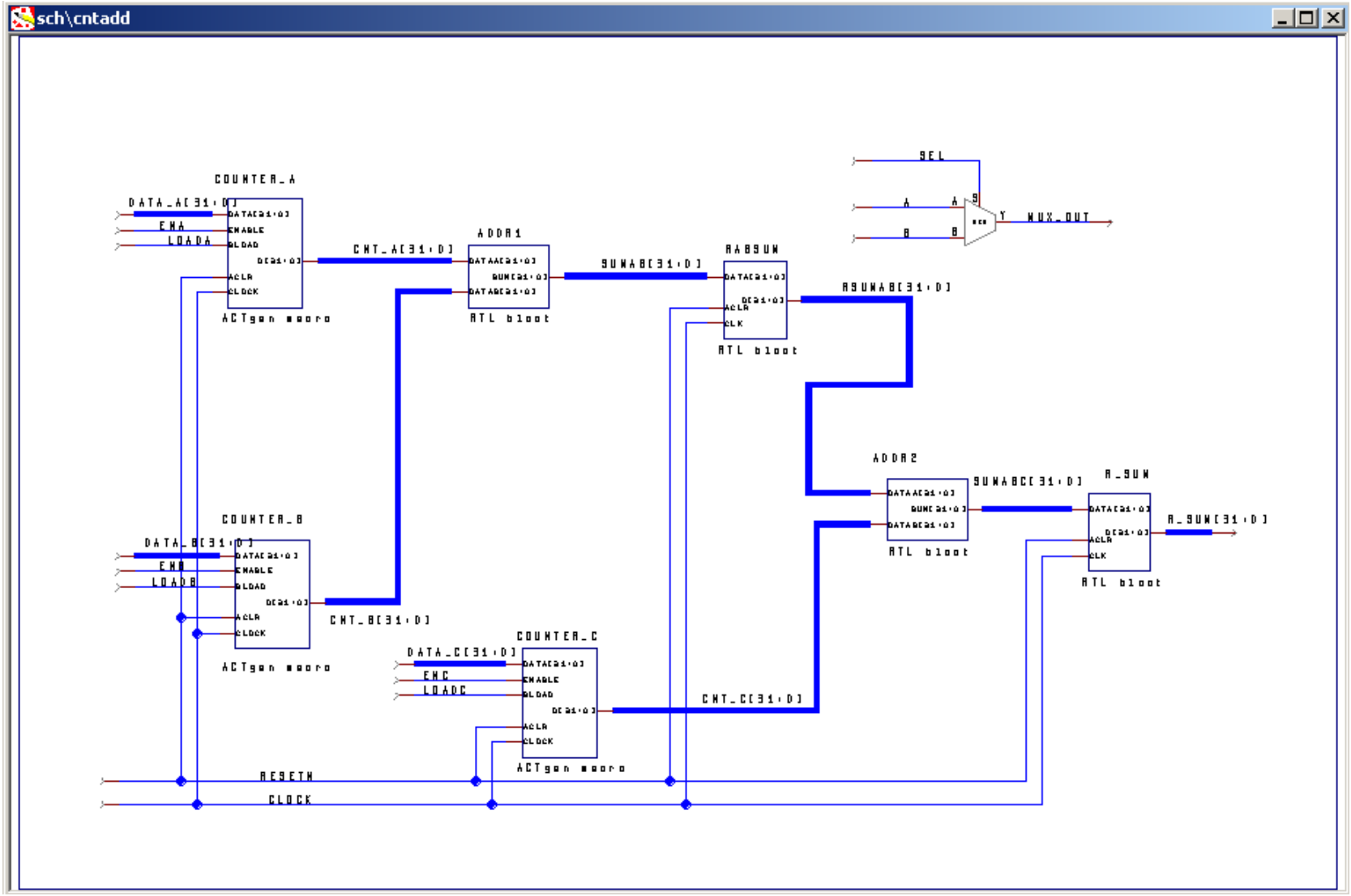
- Create RTL from Libero HDL Editor or Import File

  OR

- Create HDL Structural Implementation using SmartGen
  - VHDL or Verilog
- Create ViewDraw Symbol from Libero and Instantiate Symbol in Schematic
  - Symbol Appears on Files Tab



Mixed Mode Schematic

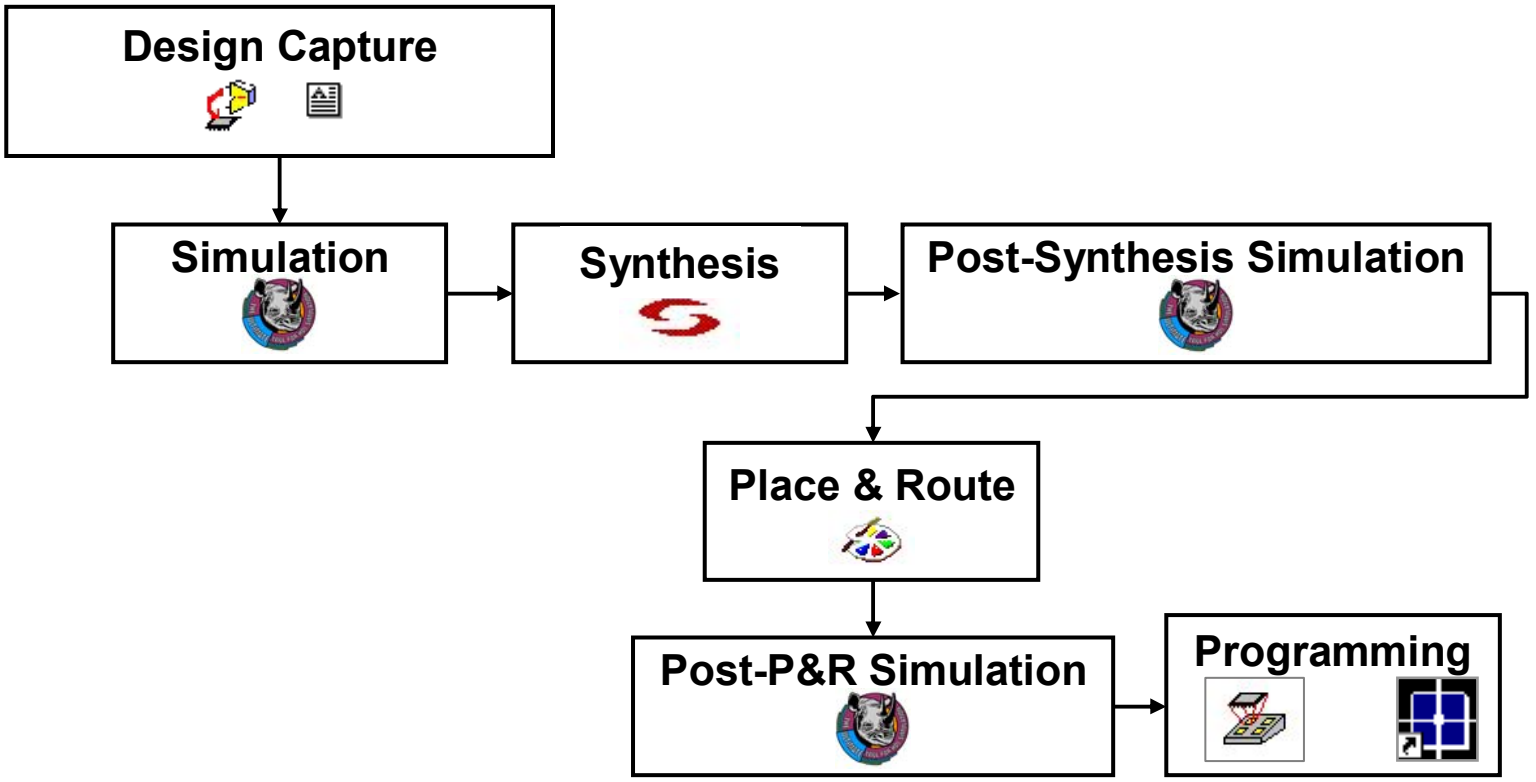


Synthesis

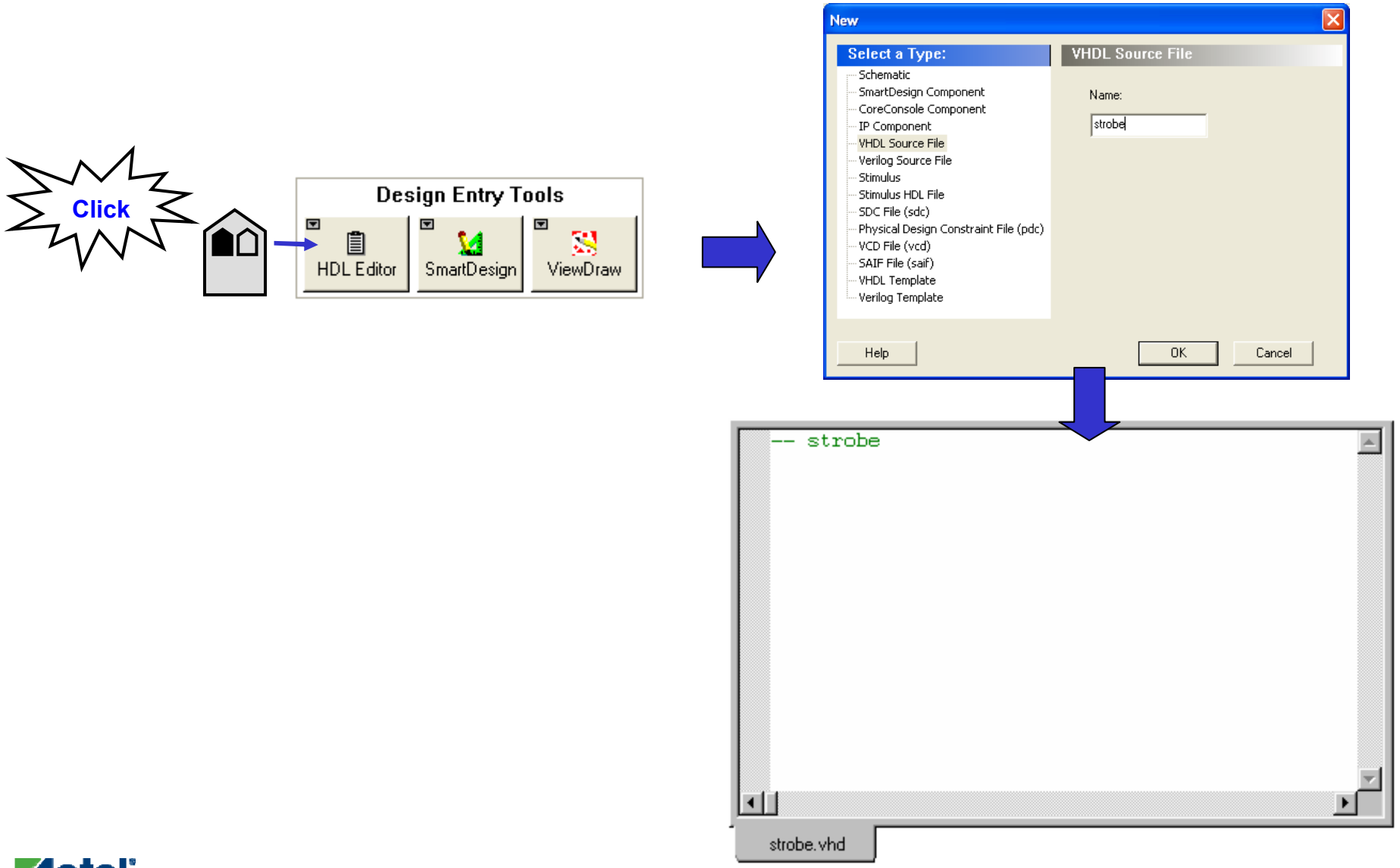
- Optional for Pure Schematic or Structural Schematic Flows
  - *All* HDL Blocks Are Structural VHDL or Verilog (e.g., SmartGen Blocks)
- Required for Mixed-mode Designs
  - Designs Containing RTL Blocks
- Libero Launches Synplicity to Insert Pads and Optimize Design
  - Hierarchical Connectors Must Be Used
  - Structural Schematics with All Pads Instantiated Can Go Directly to Designer Tool

HDL Designs

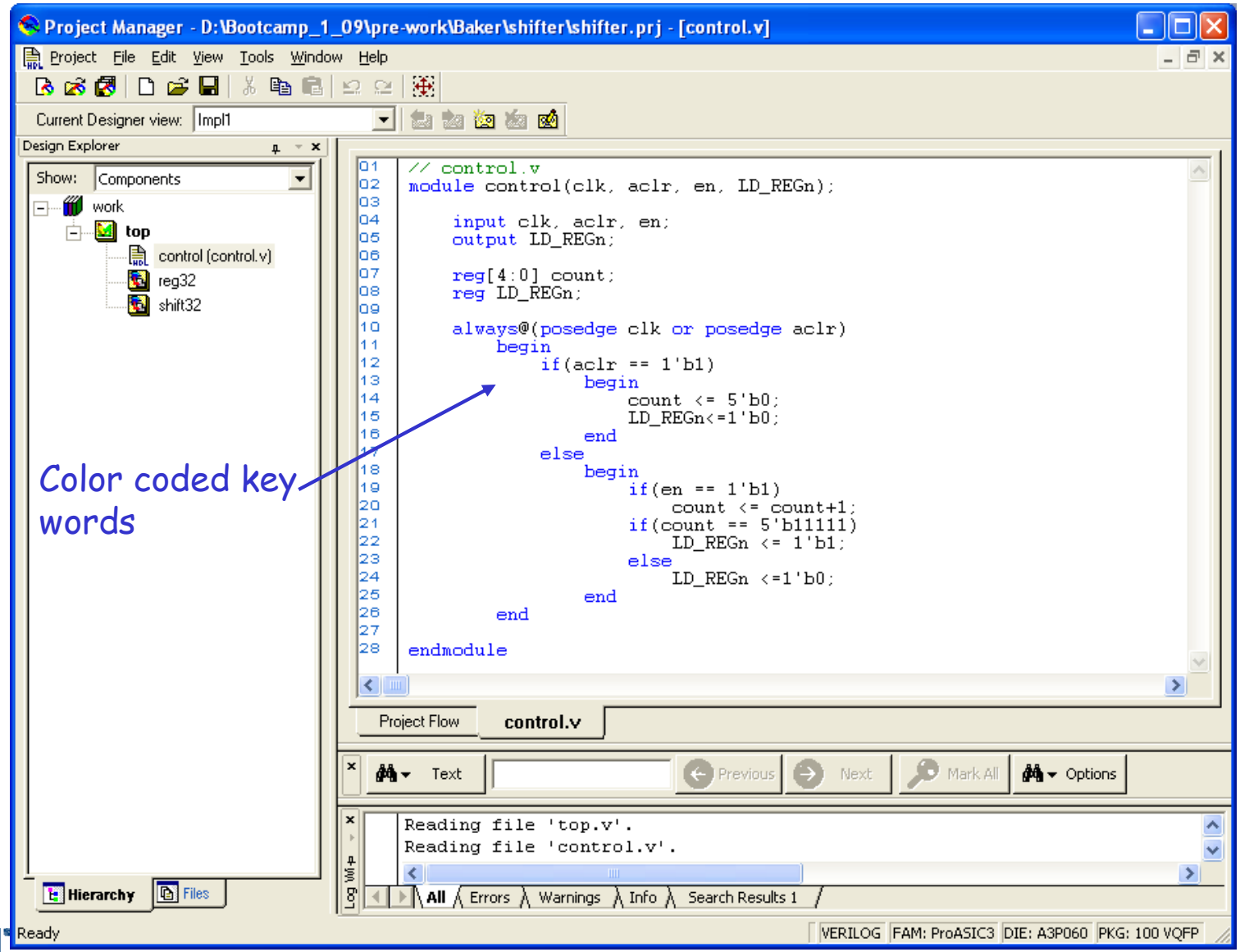
HDL Design Flow



Creating New HDL Macros

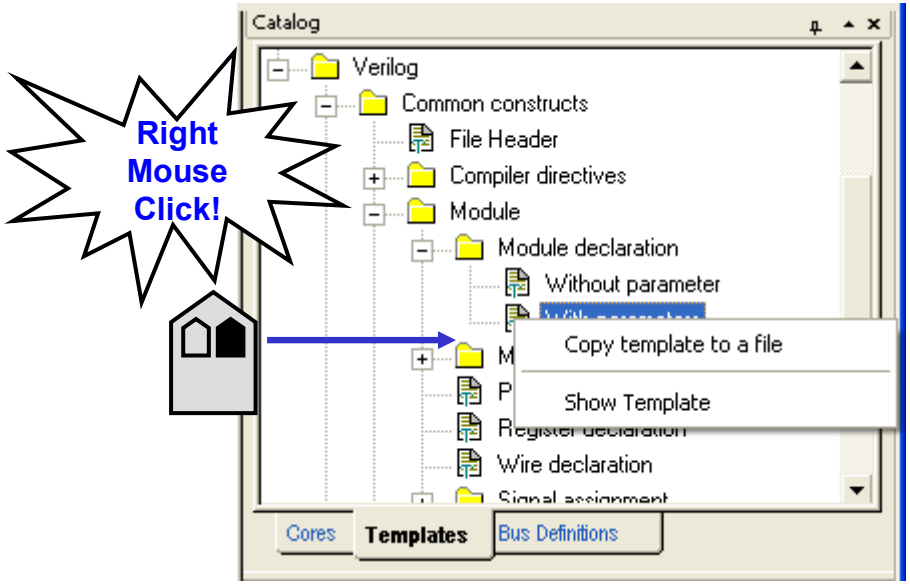


HDL Editor


Project Manager - D:\Bootcamp_1_09\pre-work\Baker\shifter\shifter.prj - [control.v]
Project File Edit View Tools Window Help
Current Designer view: Impl1
Design Explorer
Show: Components
work
top
control (control.v)
reg32
shift32
Hierarchy
Files
01 // control.v
02 module control(clk, aclr, en, LD_REGn);
03
04 input clk, aclr, en;
05 output LD_REGn;
06
07 reg[4:0] count;
08 reg LD_REGn;
09
10 always@(posedge clk or posedge aclr)
11 begin
12 if(aclr == 1'b1)
13 begin
14 count <= 5'b0;
15 LD_REGn<=1'b0;
16 end
17 else
18 begin
19 if(en == 1'b1)
20 count <= count+1;
21 if(count == 5'b11111)
22 LD_REGn <= 1'b1;
23 else
24 LD_REGn <=1'b0;
25 end
26 end
27
28 endmodule
Project Flow
control.v
Text
Previous
Next
Mark All
Options
Reading file 'top.v'.
Reading file 'control.v'.
All
Errors
Warnings
Info
Search Results 1
Log Window
Ready
VERILOG
FAM: ProASIC3
DIE: A3P060
PKG: 100 VQFP
Color coded key words

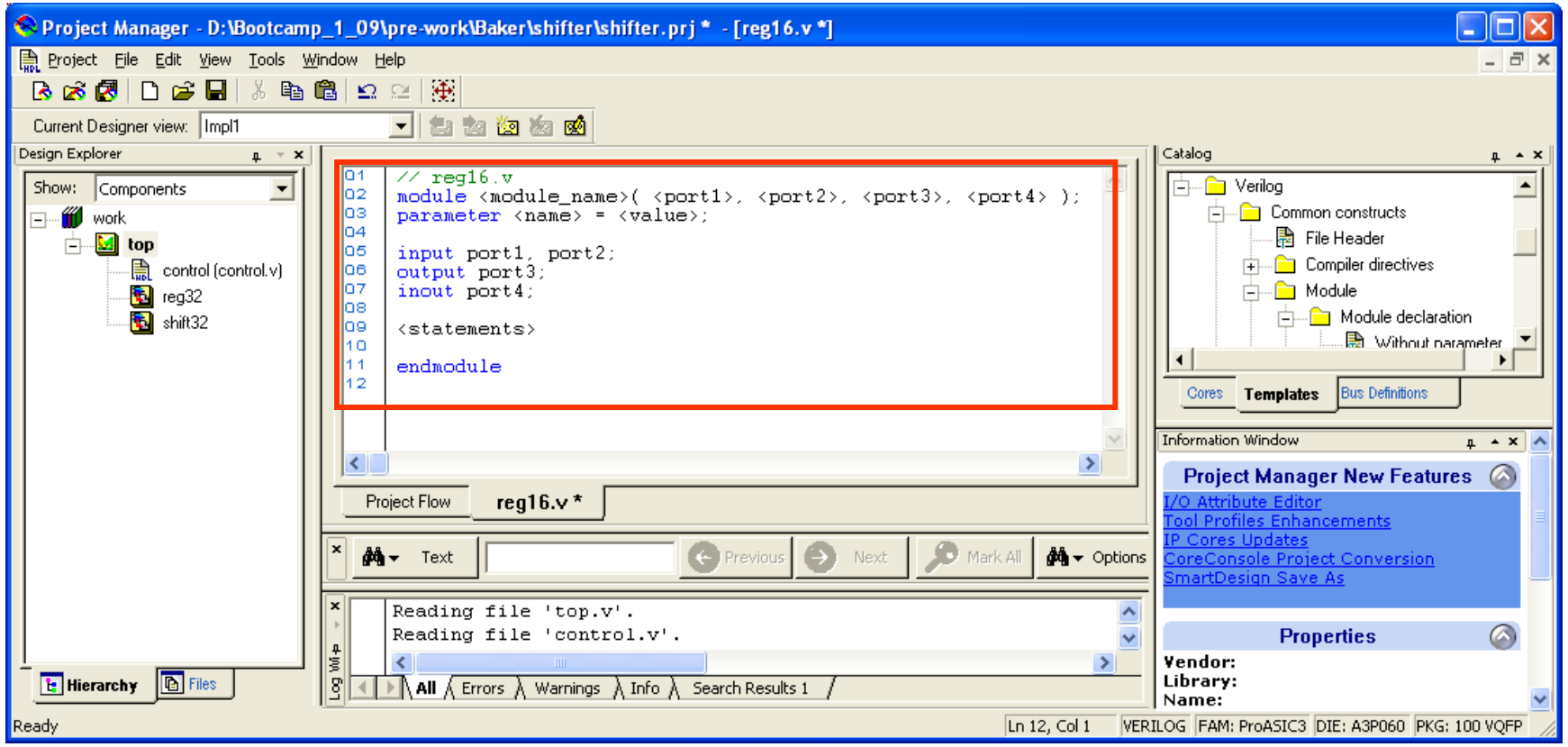
Inserting HDL Templates

- To Insert Template:
  - Place Insert Cursor Where you want it in the HDL Text Editor
  - Double Click Selected Template or ..
  - Right Click and Select “Copy template to a file”



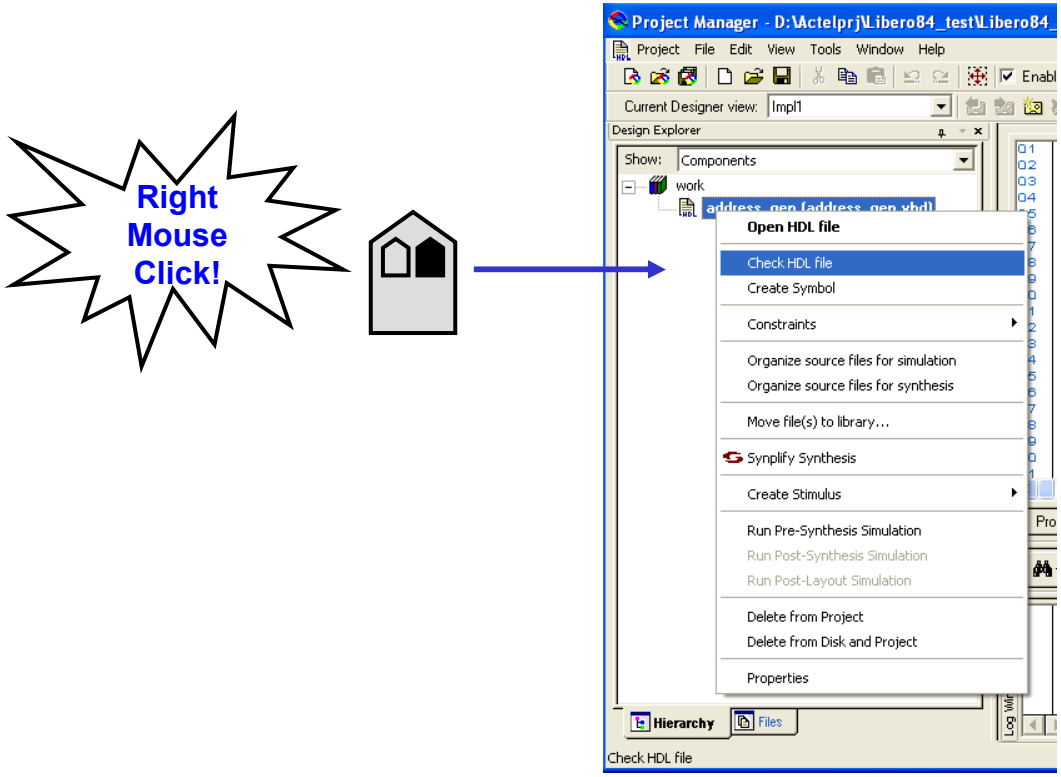
HDL Templates in Text Editor

- Template is Inserted into Text Editor Window



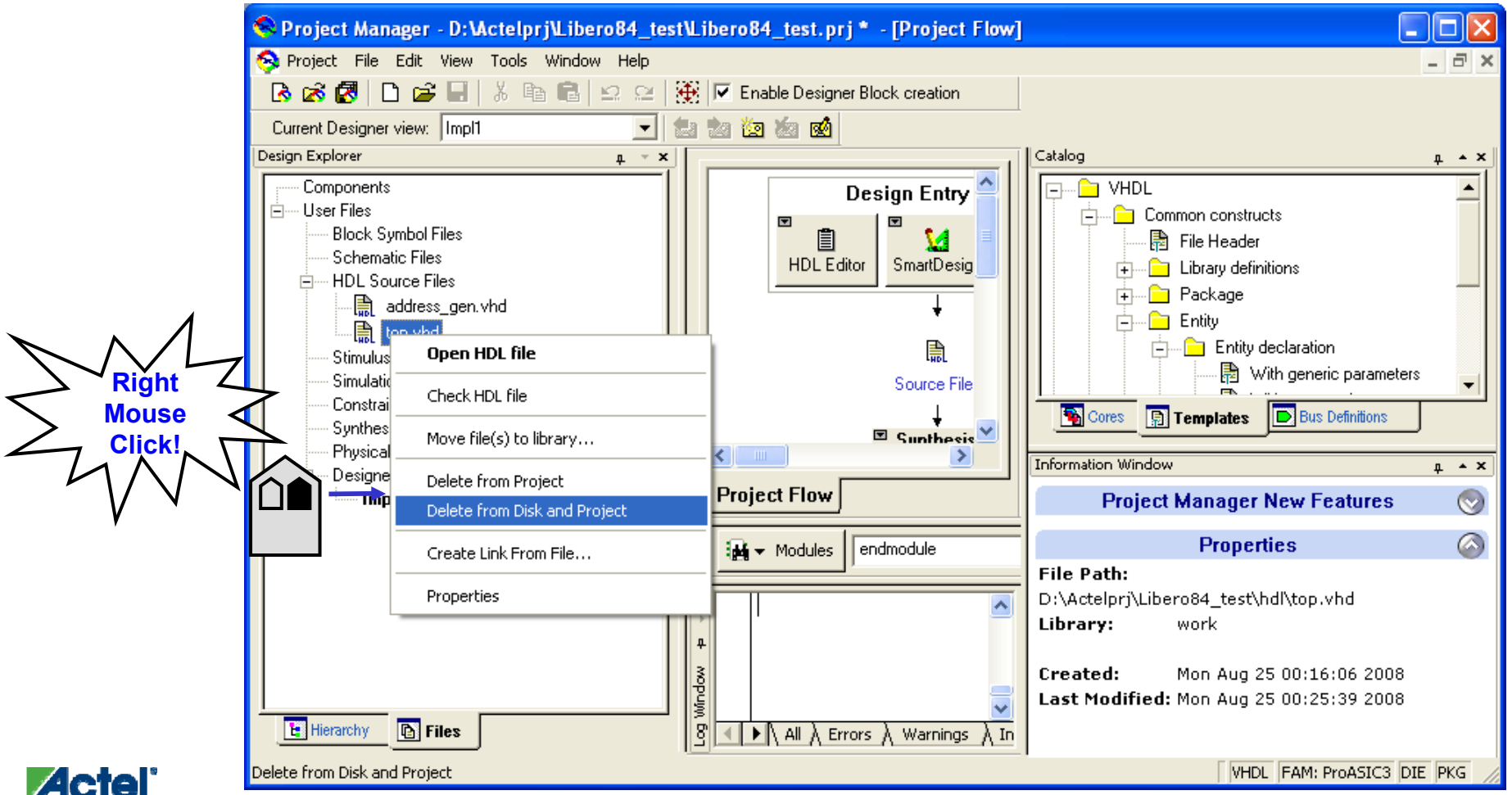
Libero HDL Syntax Checker

- HDL Syntax Checker Available from Hierarchy or Files Tab
  - Checks for Syntax Errors in HDL Blocks
  - Errors Indicated in Libero Log Window
  - Optional Step



Deleting Files from Libero Project

- Files Can Be Deleted from Project and from HDD
  - Files Deleted from HDD *Cannot* Be Recovered!



SmartDesign

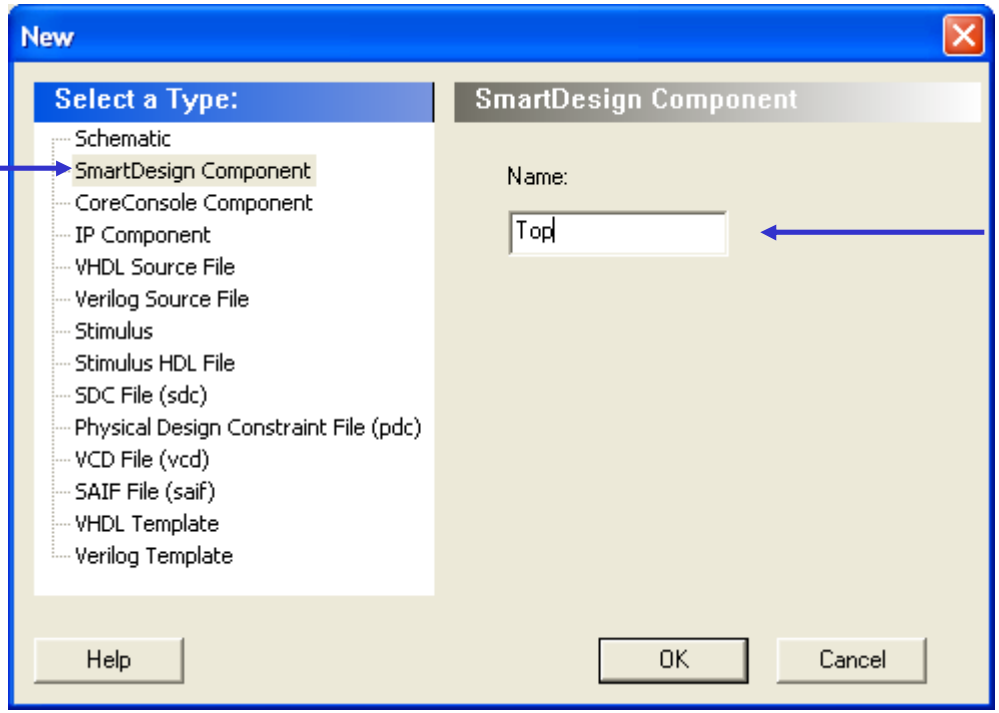
What is SmartDesign?

- Powerful Block-based Visual Design Creation Tool
  - Instantiate blocks from a variety of sources
    - DirectCore IP, SmartGen, User HDL, Companion Cores, Actel library cells, and the list goes on.
  - Supported for all platforms
- Simple and Intuitive Design Creation
  - Auto Connect
  - Fast manual connectivity between blocks
  - Hierarchical design support
- DRC
  - Checks rules to guarantee correct by construction design
    - Connectivity errors
    - Configuration errors
    - Special silicon rules
- SOC Features
  - Auto Connect
    - clocks and resets for processors and peripherals
    - Other known DirectCore connections
  - Memory Map Configuration Dialog
  - Testbench and Bus Functional Model (BFM) script generation

Opening SmartDesign

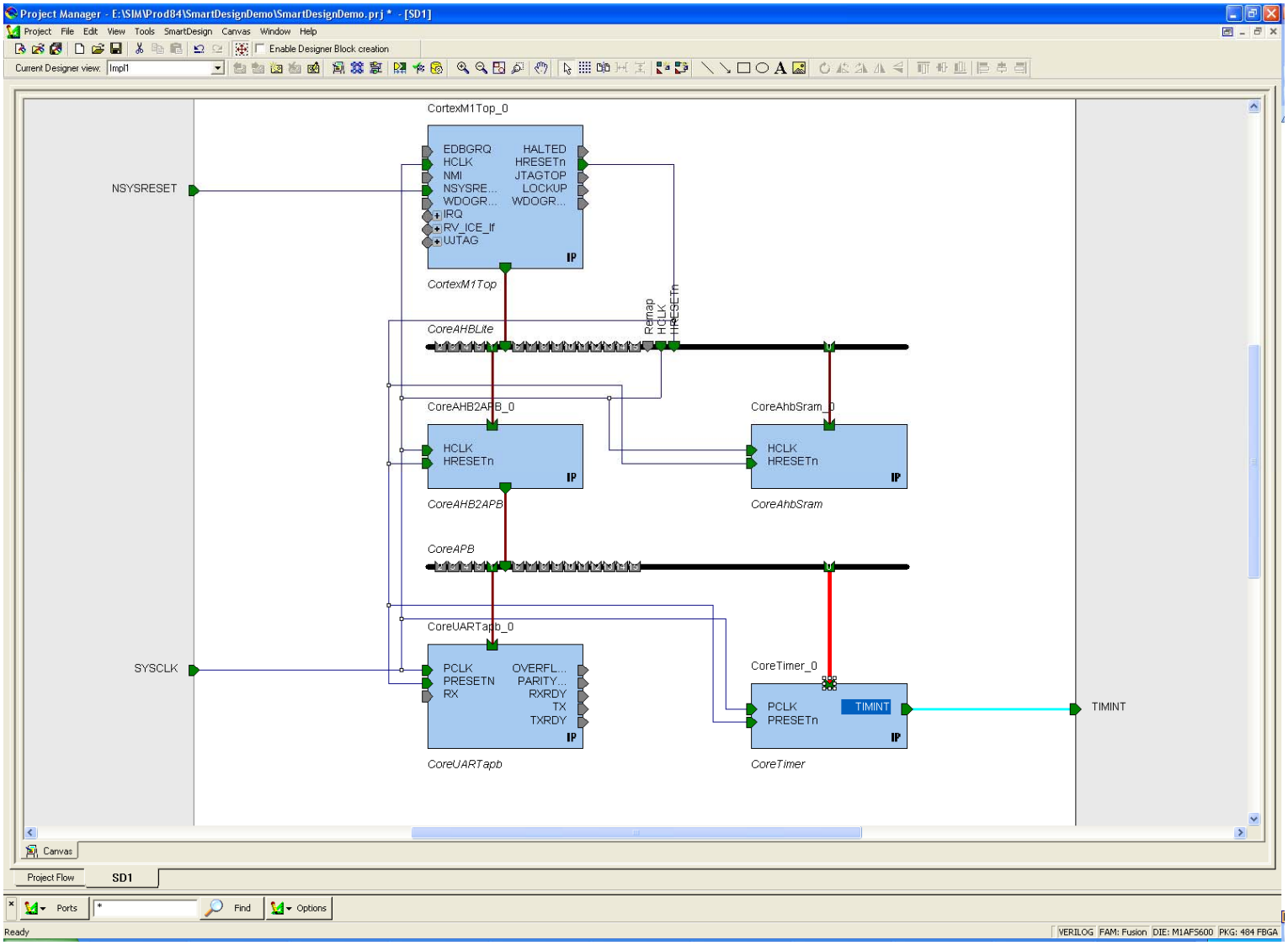
- Click SmartDesign Button in Project Flow Window
- Add and Stitch Blocks
- Generate HDL

Select SmartDesign Component



Enter name

SmartDesign Canvas

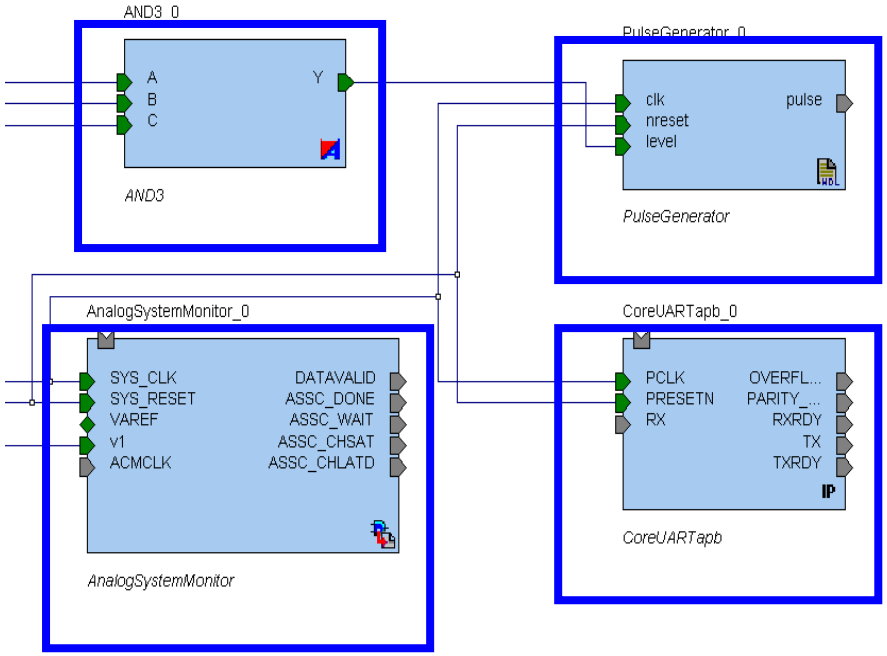


SmartDesign Canvas Enhancements

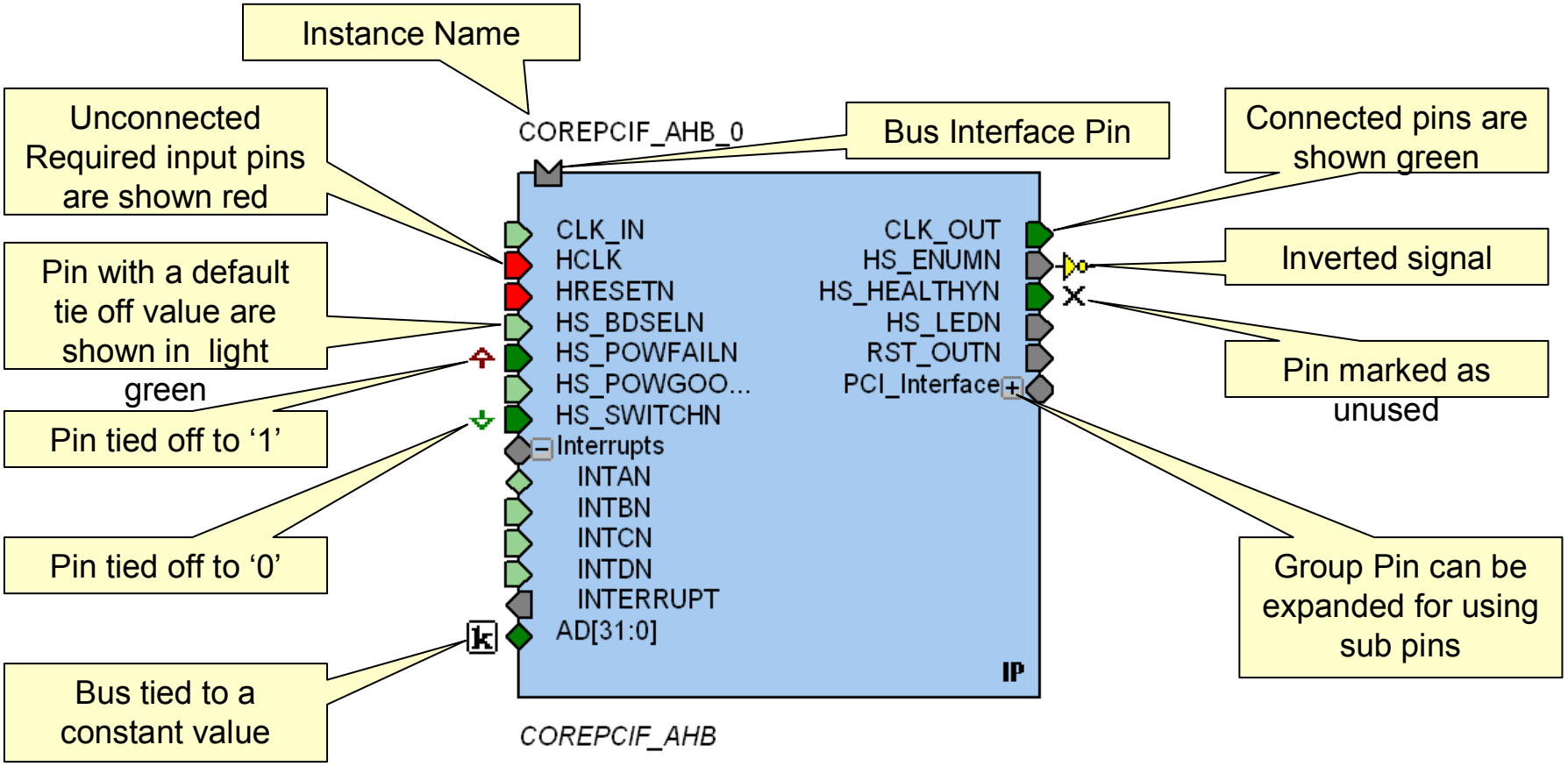
- SmartDesign Canvas
  - Instance pins are displayed on canvas.
  - Connections are shown using nets
    - Displaying of Nets is optional
    - Selective enabling / disabling of showing nets
  - Drag and Drop directly from the Catalog into the Canvas
- All Design Operations Available in the Canvas
  - Connect / Disconnect
  - Promote To Top
  - Tie Low / Tie High / Tie Constant / Inversion
  - Float
  - Split (if bus)
  - Group

Design with any block types

- One Tool to Connect Your Design
  - DirectCore IP
  - SmartGen Cores
  - User HDL
  - Actel Macros (And, Or, I/Os, etc)

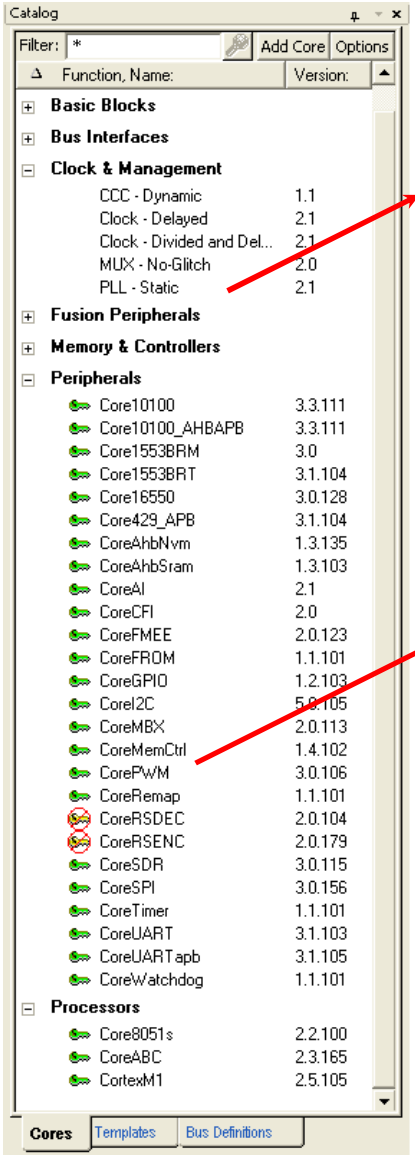


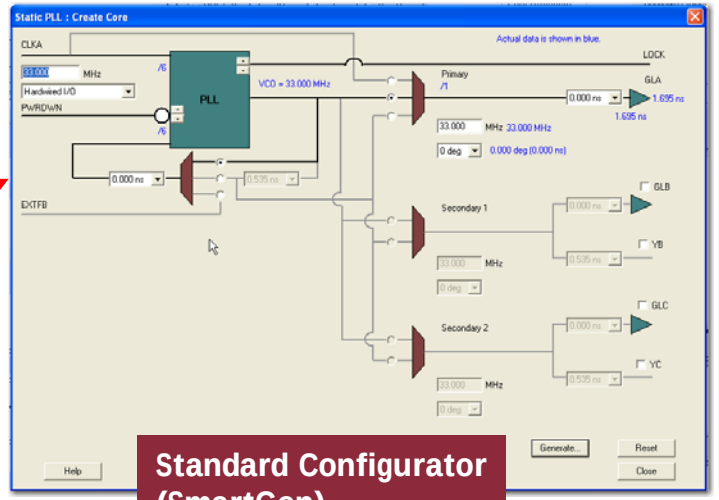
Canvas Instance

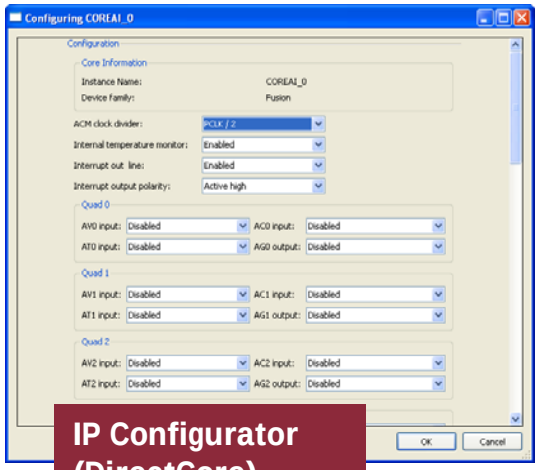


Cores Catalog

- Library of Proven Configurable Core Functions
- Actel Macros
- Quick-find
- Intuitive Configuration
- Drag and drop to Canvas

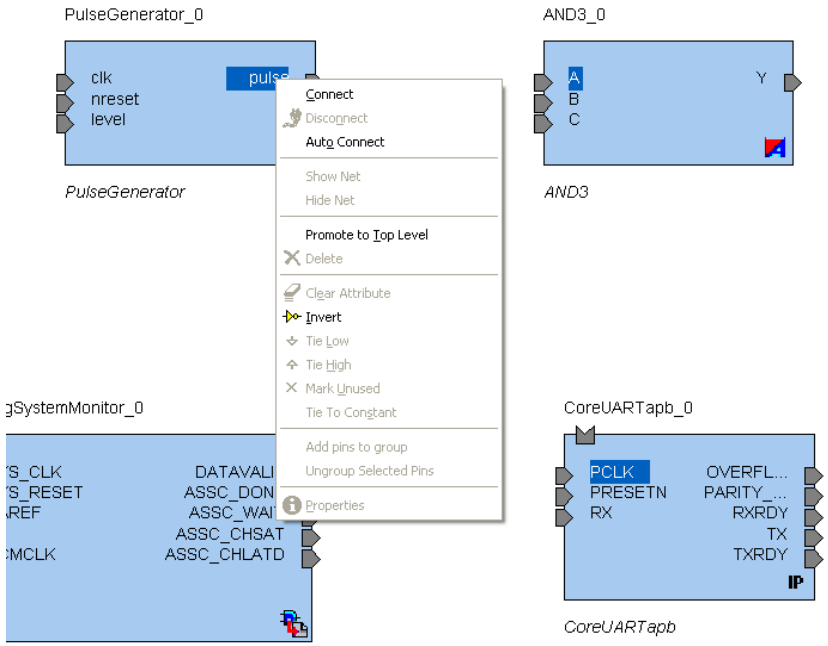






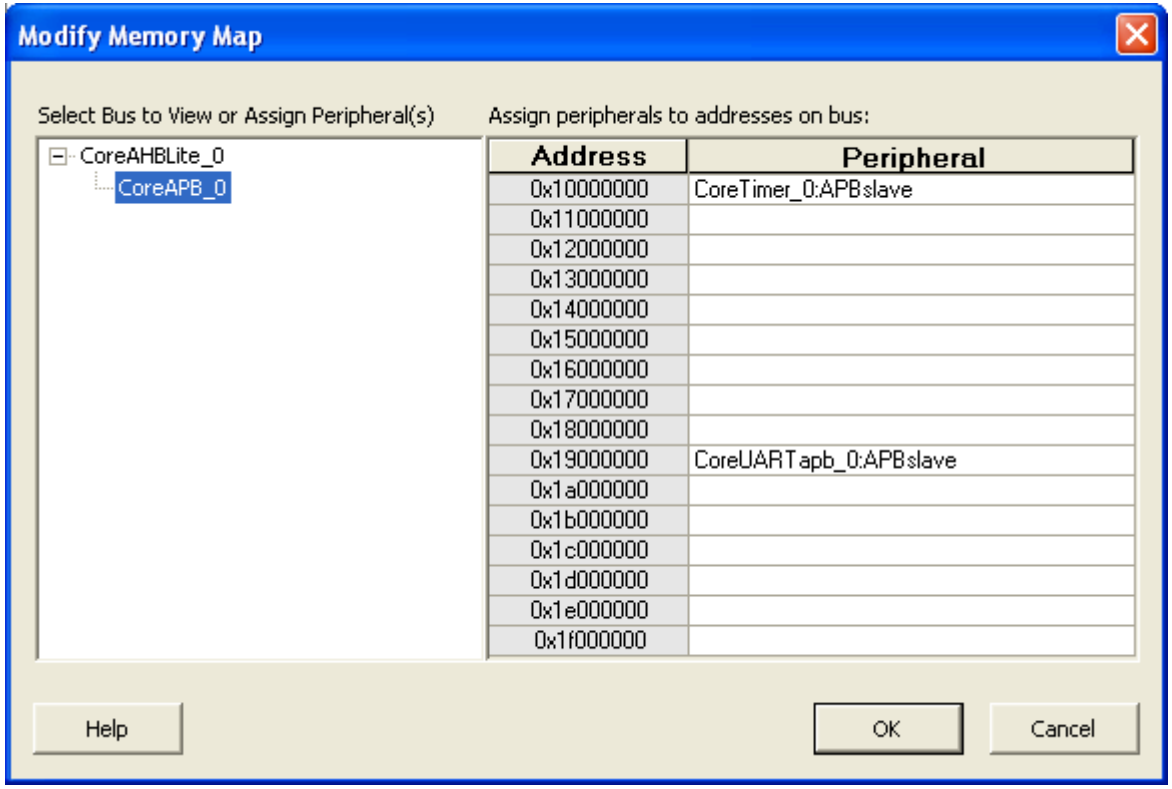
Making Connections

- Right-click on a pin for Available Operations
- Select 2 or More Pins With the CTRL key, Right-click and Connect



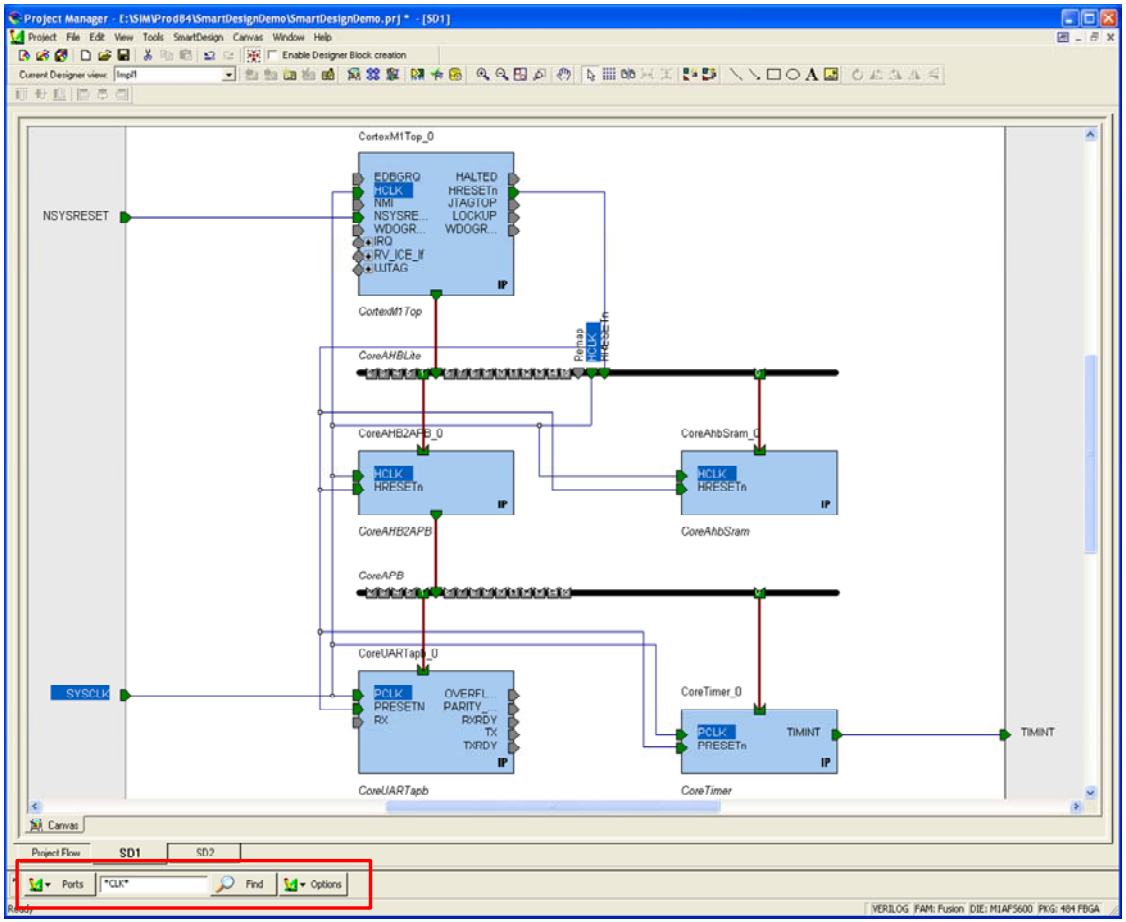
Modifying Memory Map

- Easy and Intuitive Method to Connect Peripherals at Particular Addresses on the Bus
- Interactive and Immediate Updating of Base Addresses



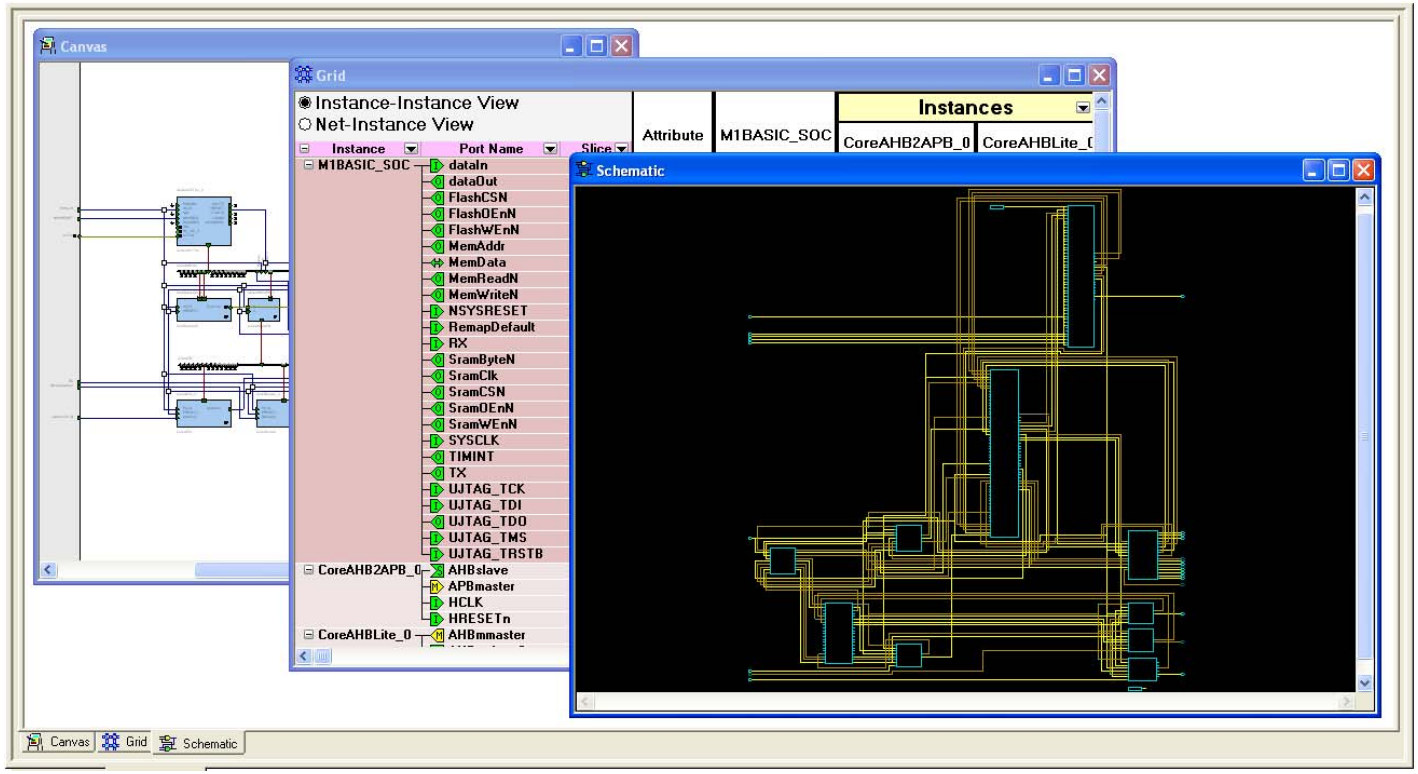
Fast Design Search

- Design Level Search
  - Find Pins, Instances, and Nets quickly and easily
  - Wild card query
  - Matching objects will be highlighted in the Canvas



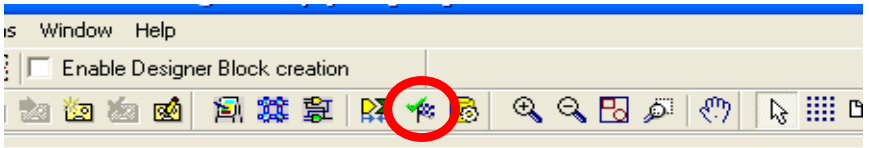
Multiple Representations of the Design

- Connectivity Grid
  - Spreadsheet like view of your design
  - Enables quick filtering / sorting
- Schematic
  - Shows all pins and nets
  - Traditional schematic view

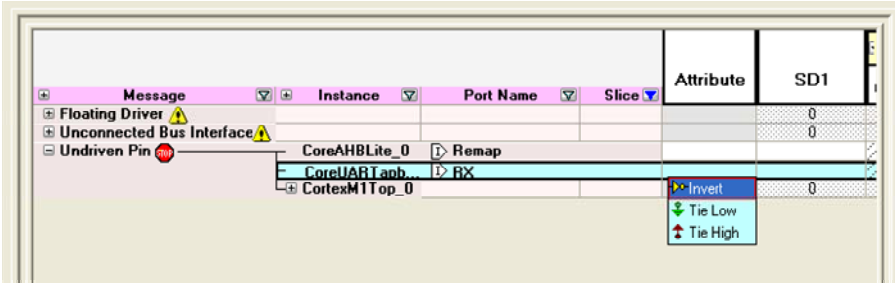


Design Rule Checker

- Checks your Design for Errors
  - Invoke with Checker icon (see picture)
  - Connectivity
    - Unconnected/Required input pins
    - Floating output pins
    - Silicon required connection
      - Example: RTC must be driven by Crystal Oscillator
  - Configuration
    - Check consistency between configurations
      - Example: CoreMP7 and CoreMP7Bridge debug configuration



- Errors Reported in a Connectivity Grid
  - Directly fix your connectivity mistakes
  - Enables fast sorting / filtering

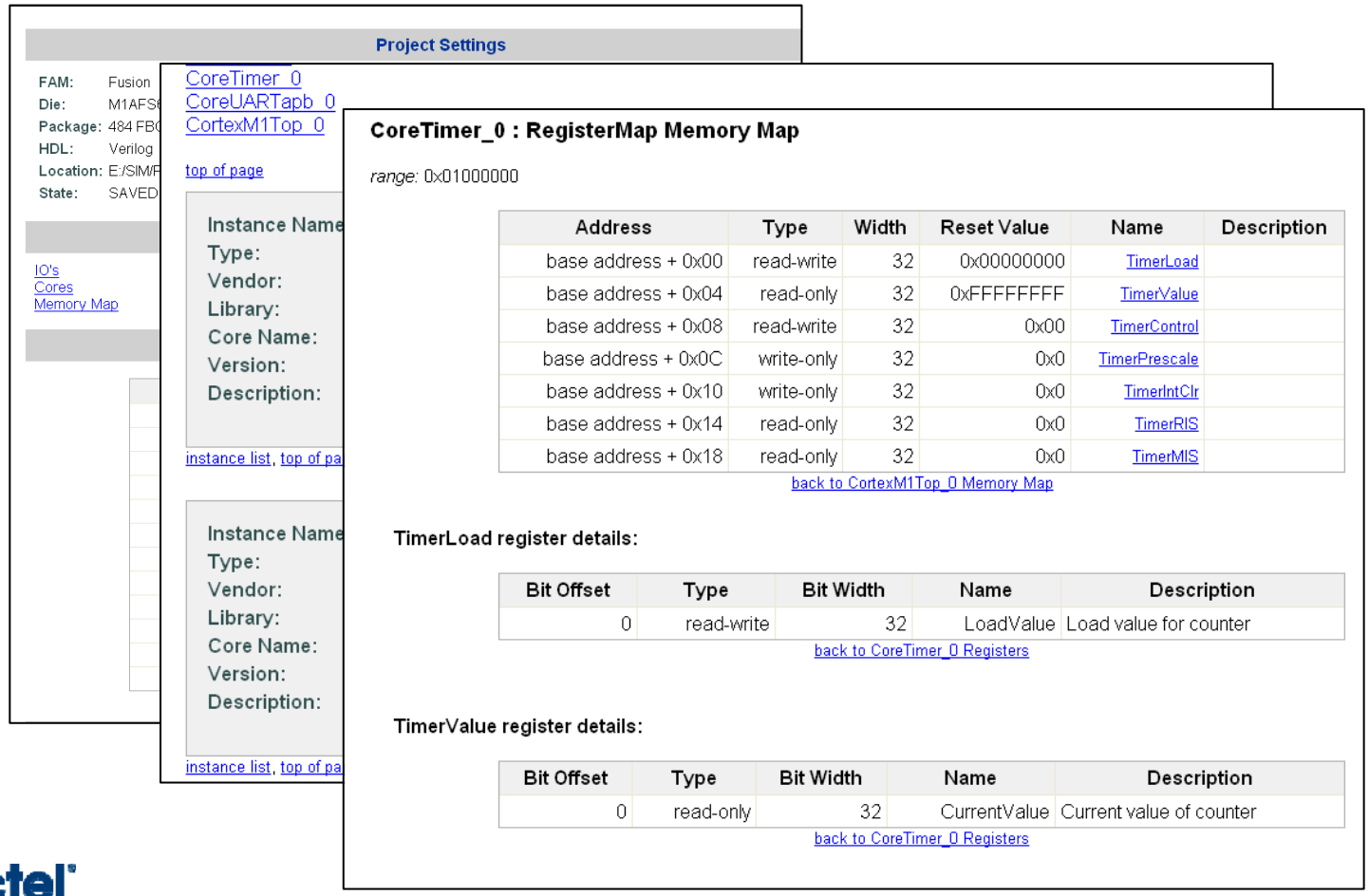


Testbench Generation

- Design Testbench
  - Generates a top level testbench
    - Clock and Reset drivers automatically generated and connected
- Bus Functional Model Script Generation
  - Generates BFM script file for processor based designs
  - Based on your peripheral connectivity in your designs
  - Look at Processor Core Handbooks (ex: CortexM1) for more details on BFMs and usage

Datasheet Generation

- Design Datasheet
  - Pin outs of the design
  - Cores used and their description
  - Memory Map



SmartGuide Design Guidance

- Identify Silicon Design Rules and Compatibilities
  - Eliminate errors typically detected only at place & route time
    - Example: Using RTC in Fusion requires the Crystal Oscillator
- Detect and Flag Logic Design Errors
  - Eliminate errors typically detected only at synthesis time
    - Un-driven inputs
    - Floating outputs
- Restrict Illegal Choices When Making Connections
  - Direction incompatibilities
  - Dimension mismatches
- Auto-connect Standard Interfaces

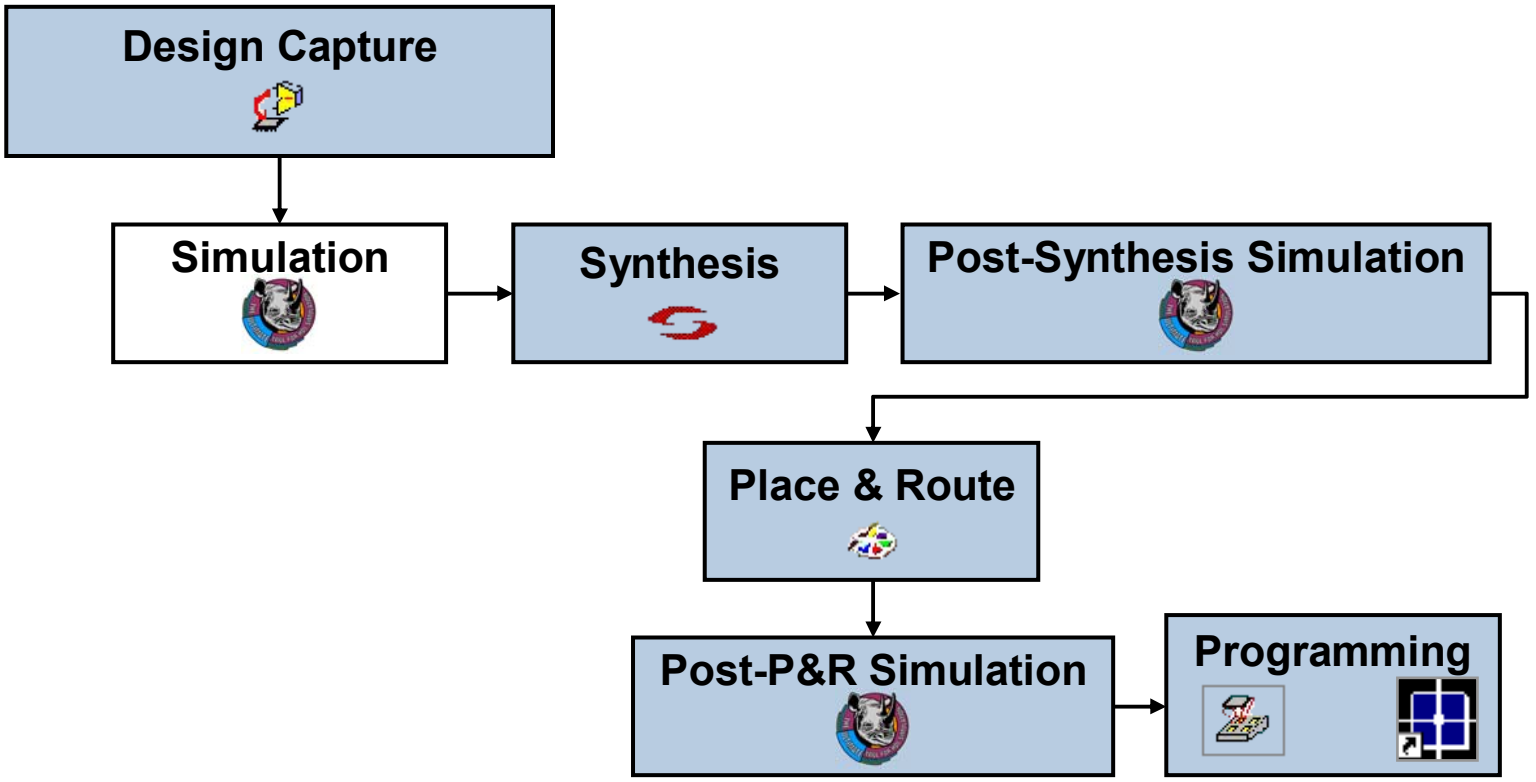
SmartGuide Design Management

- Streamline Iterative Design Changes
  - Unique to SmartDesign
  - Detect and flag out-of-date components
  - Assist in component interface updates
    - E.g. Removing connections to a deleted port
- Synchronize Data Between Building Blocks
  - E.g., Flash Memory must be updated because Analog System changed
- Flag the Next Steps For a Design Change
  - Example, Reprogram the Flash Memory region for Boot-Code if a new Boot-Code file is generated
  - Example, Re-synthesize the design if a new Analog peripheral is added to the design

Hands-on Labs 1 and 2

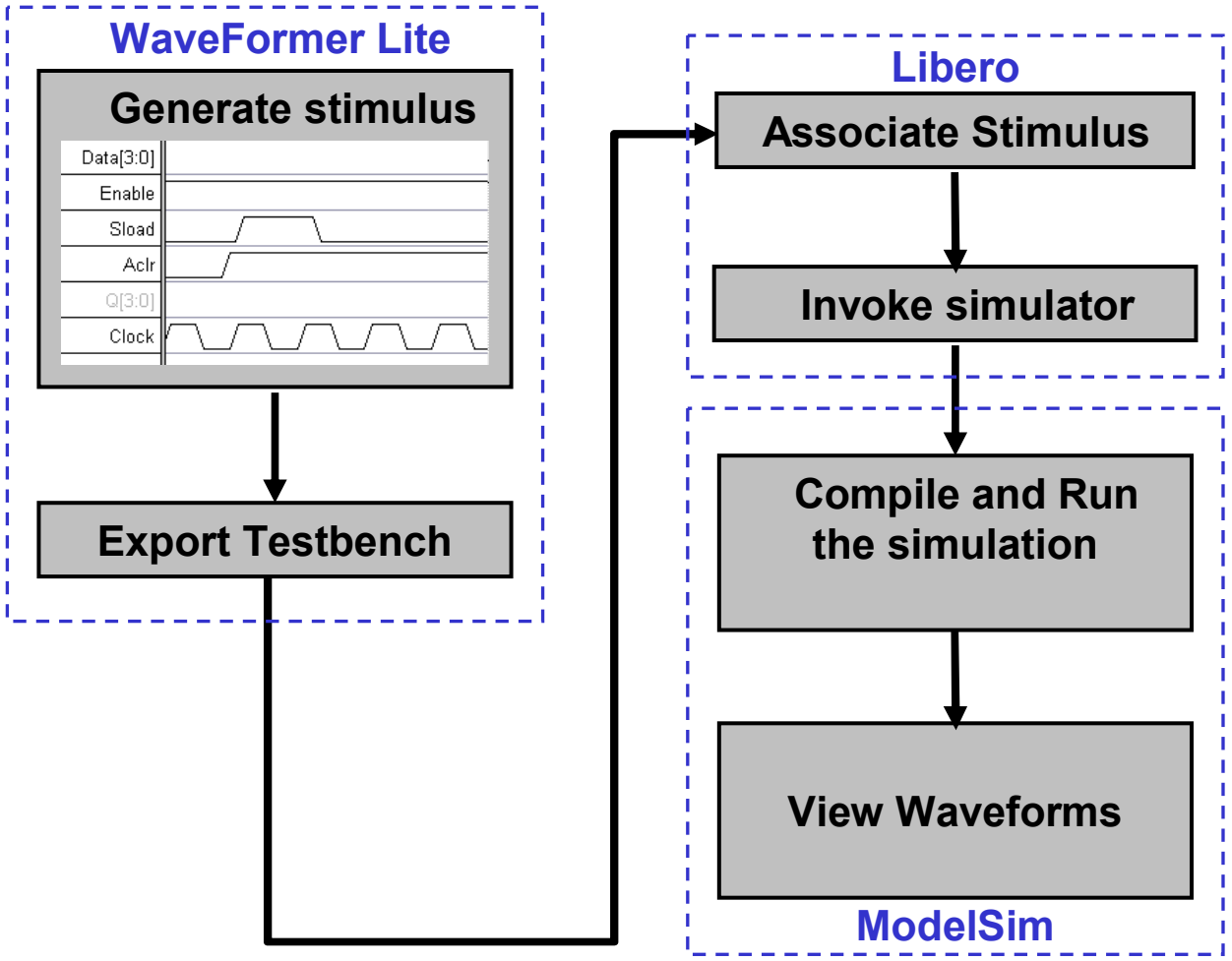
- Complete Labs 1 and 2 in the Lab Guide
- Lab 1
  - Extract Lab Files
  - Create New Libero Project and Import Source Files
- Lab 2
  - Configure FlashROM with SmartGen Save Your Work

Functional Simulation

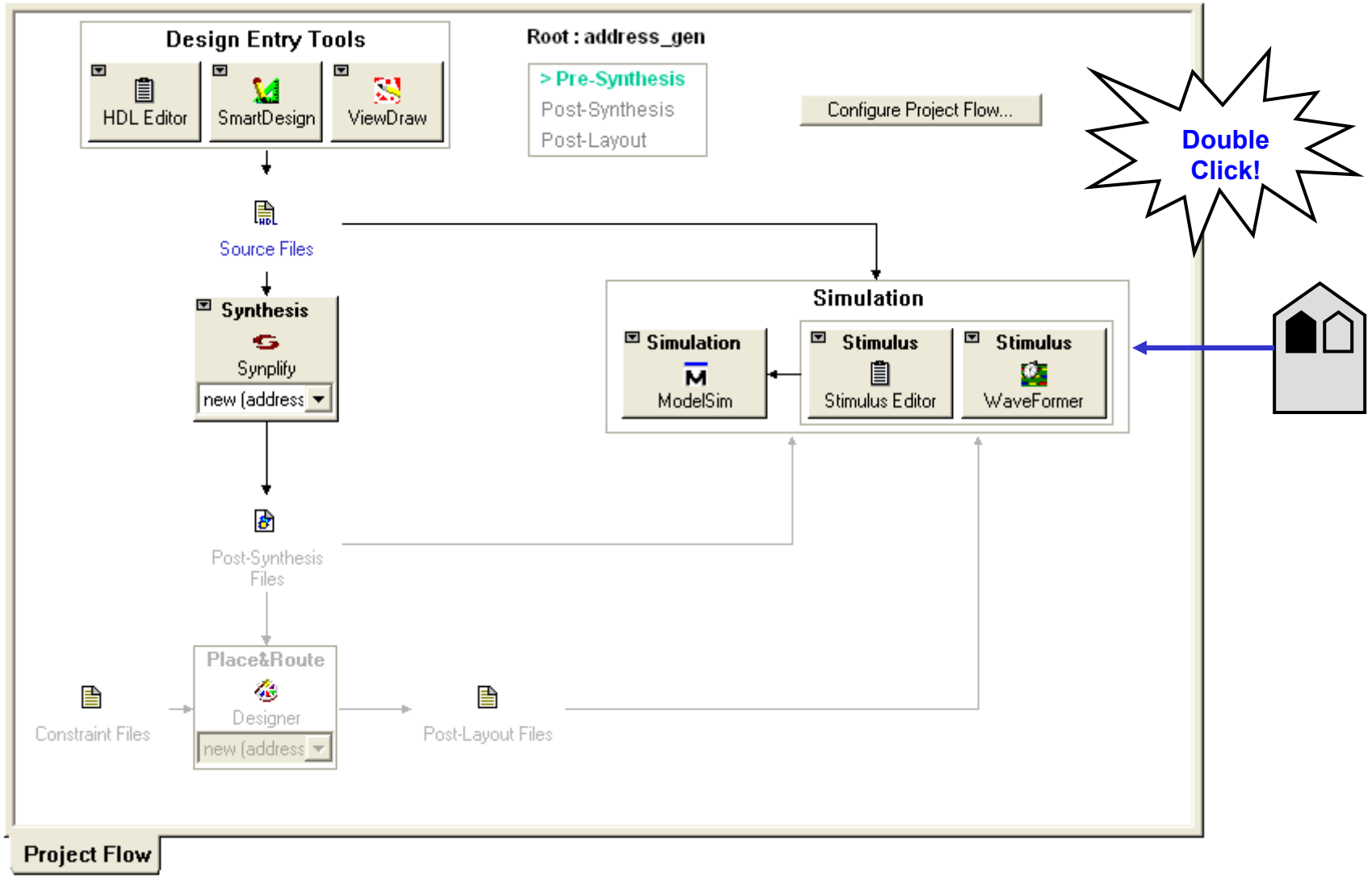


Simulation Flow

- For Each Block You Want to Simulate . . .



Invoking WaveFormer Lite

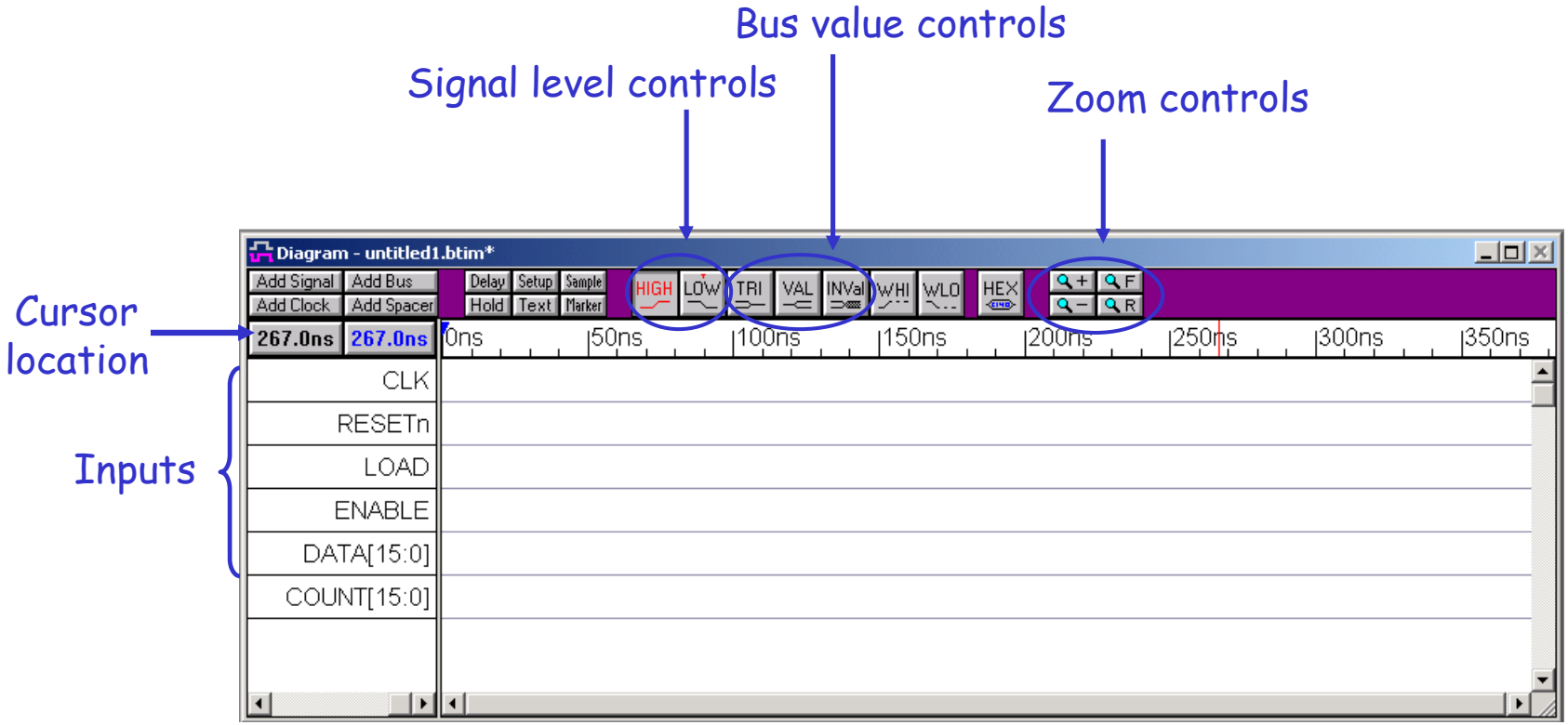


WaveFormer Lite Features

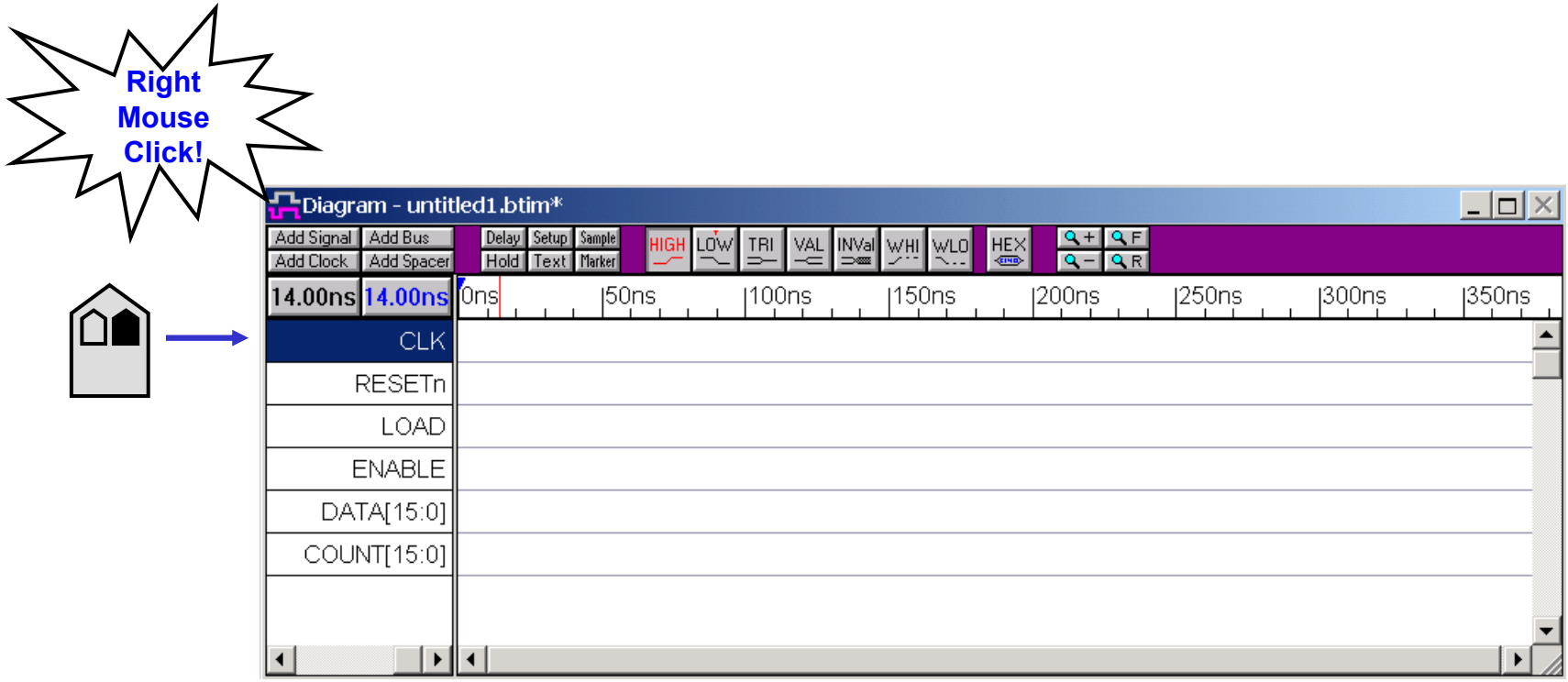
- Allows Convenient Test Stimulus Specification via GUI
  - User Specifies Stimulus by Drawing Waveforms
    - Supports Copy / Paste / Append Operations
  - Significantly Reduces Testbench Creation Time
  - Automatically Converts Graphical Stimulus Files into HDL TestBenches
  - Can Generate:
    - VHDL Testbench (*.vhd)
    - Verilog Testbench (*.v)
- Users Can Annotate Waveform For Design Documentation

Drawing Stimulus

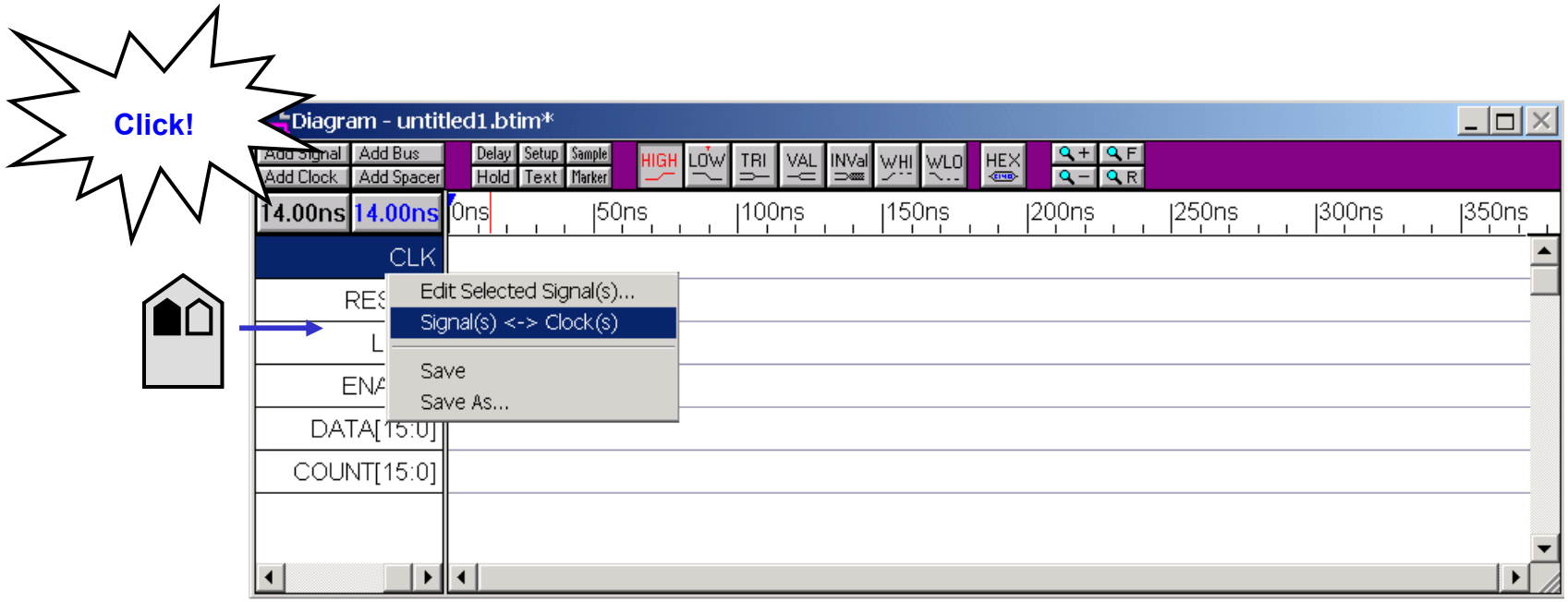
- WaveFormer Lite Diagram Window



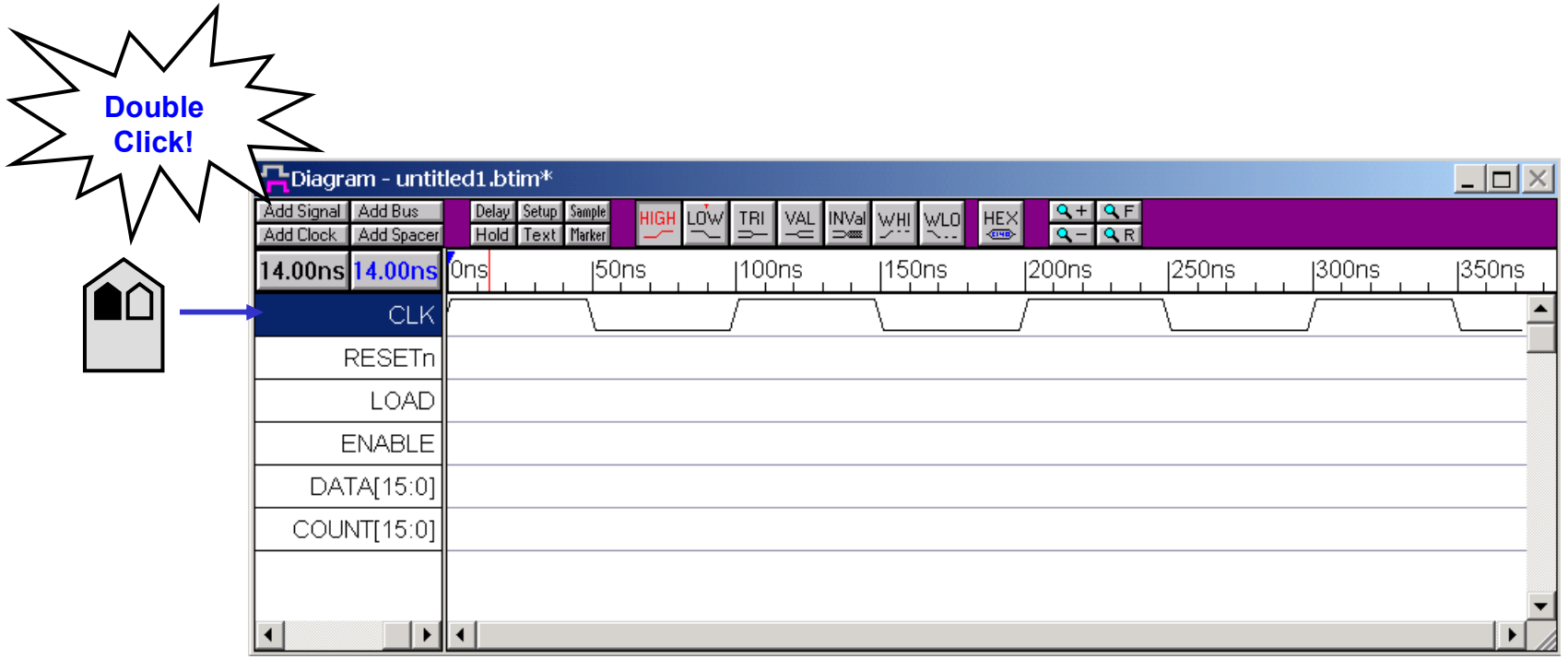
Creating Clocks



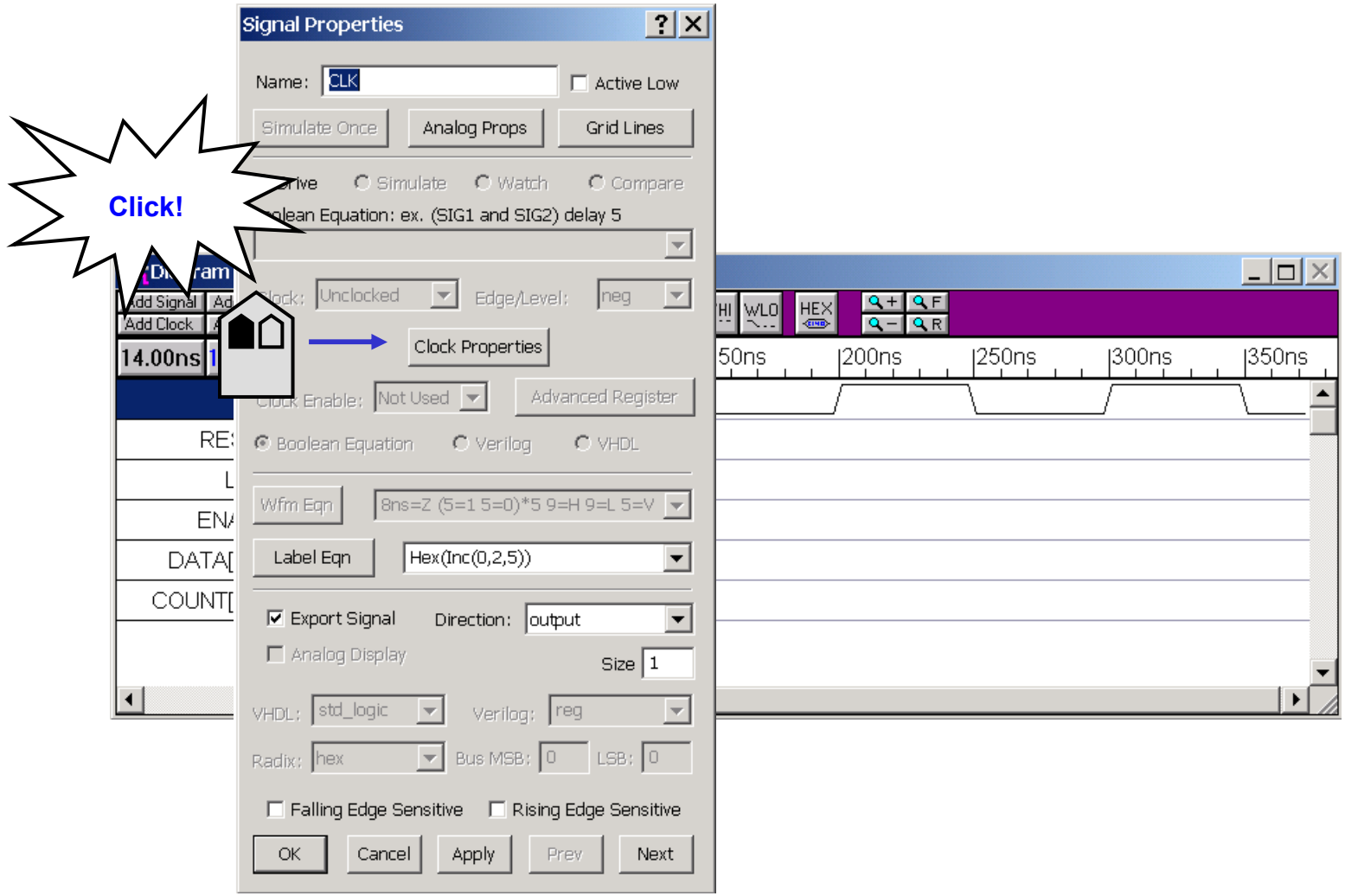
Creating Clocks



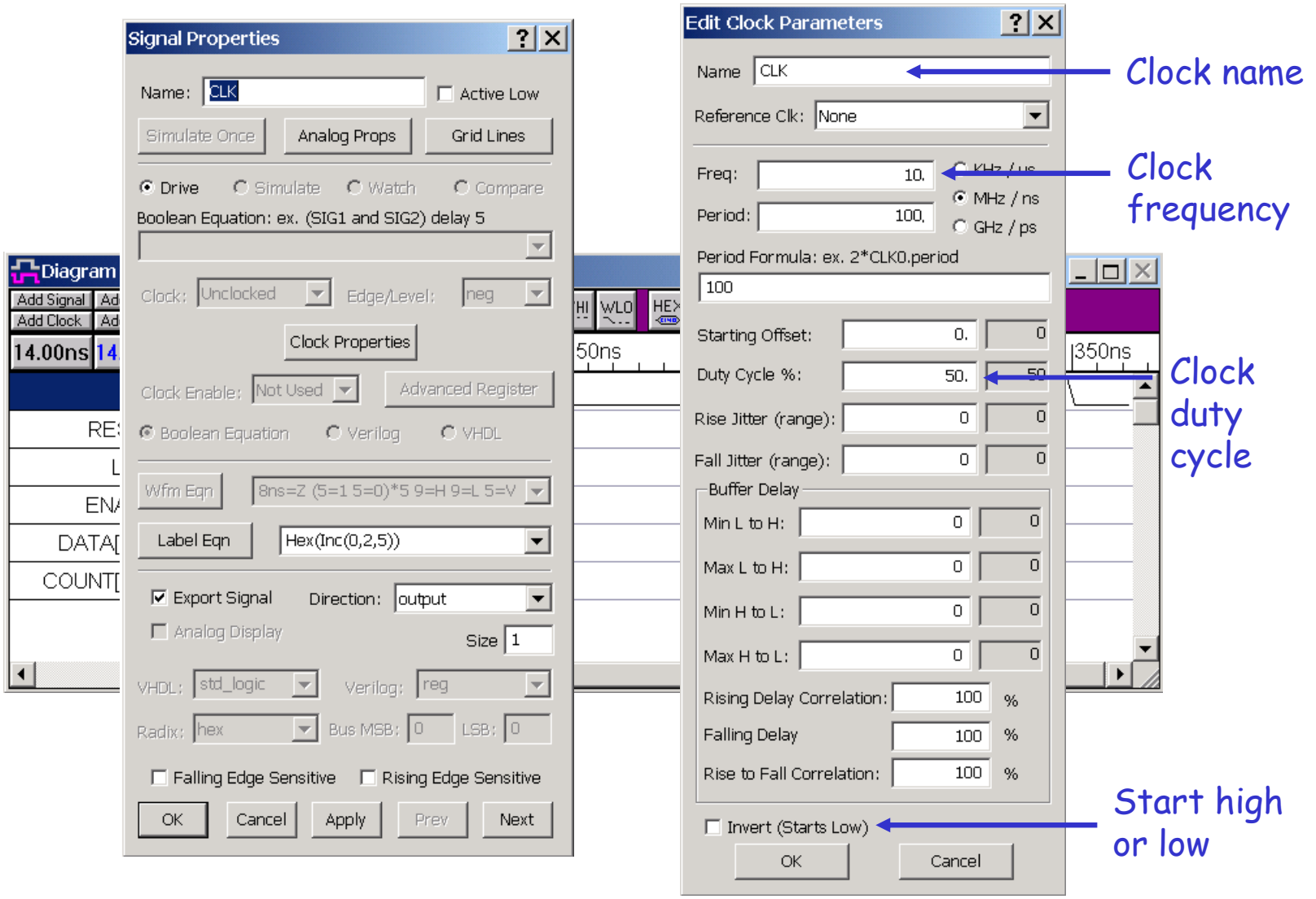
Creating Clocks



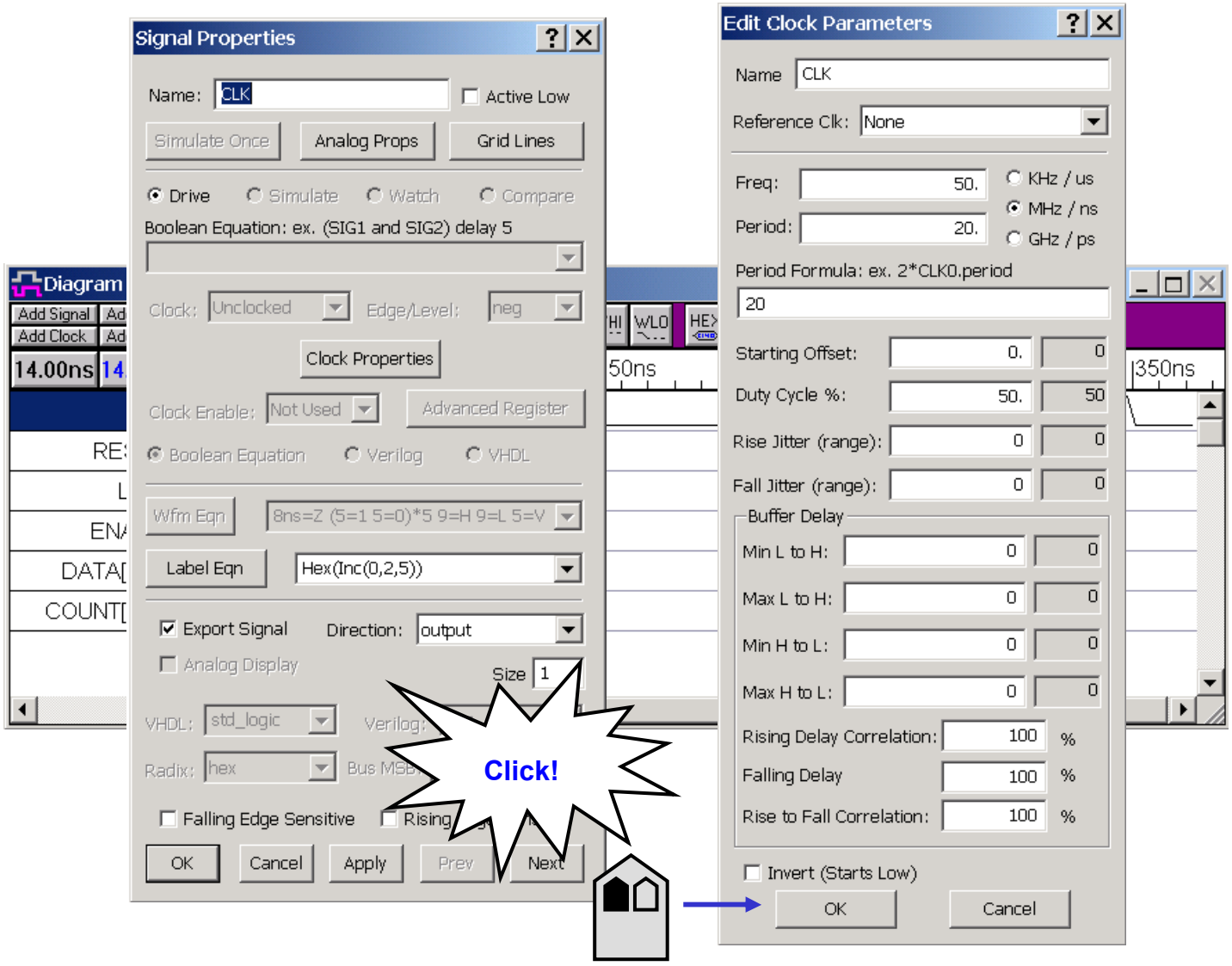
Creating Clocks



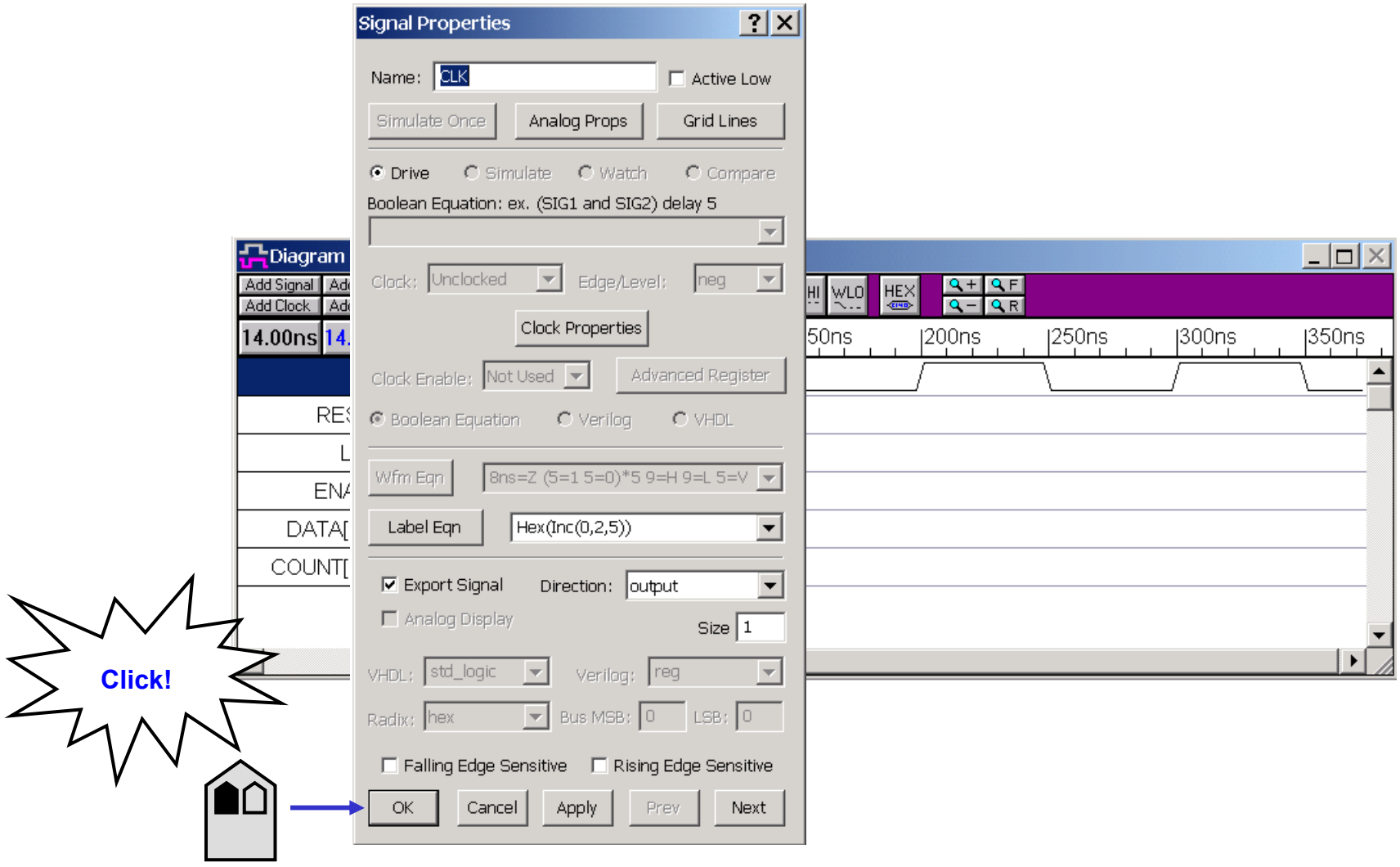
Creating Clocks



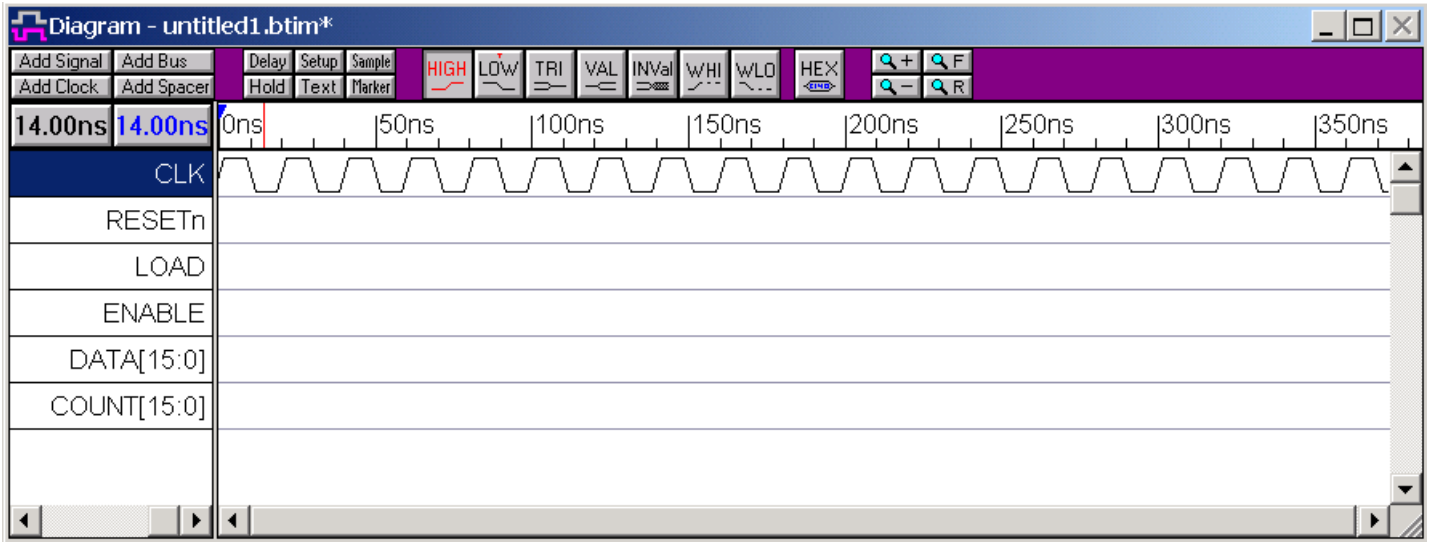
Creating Clocks



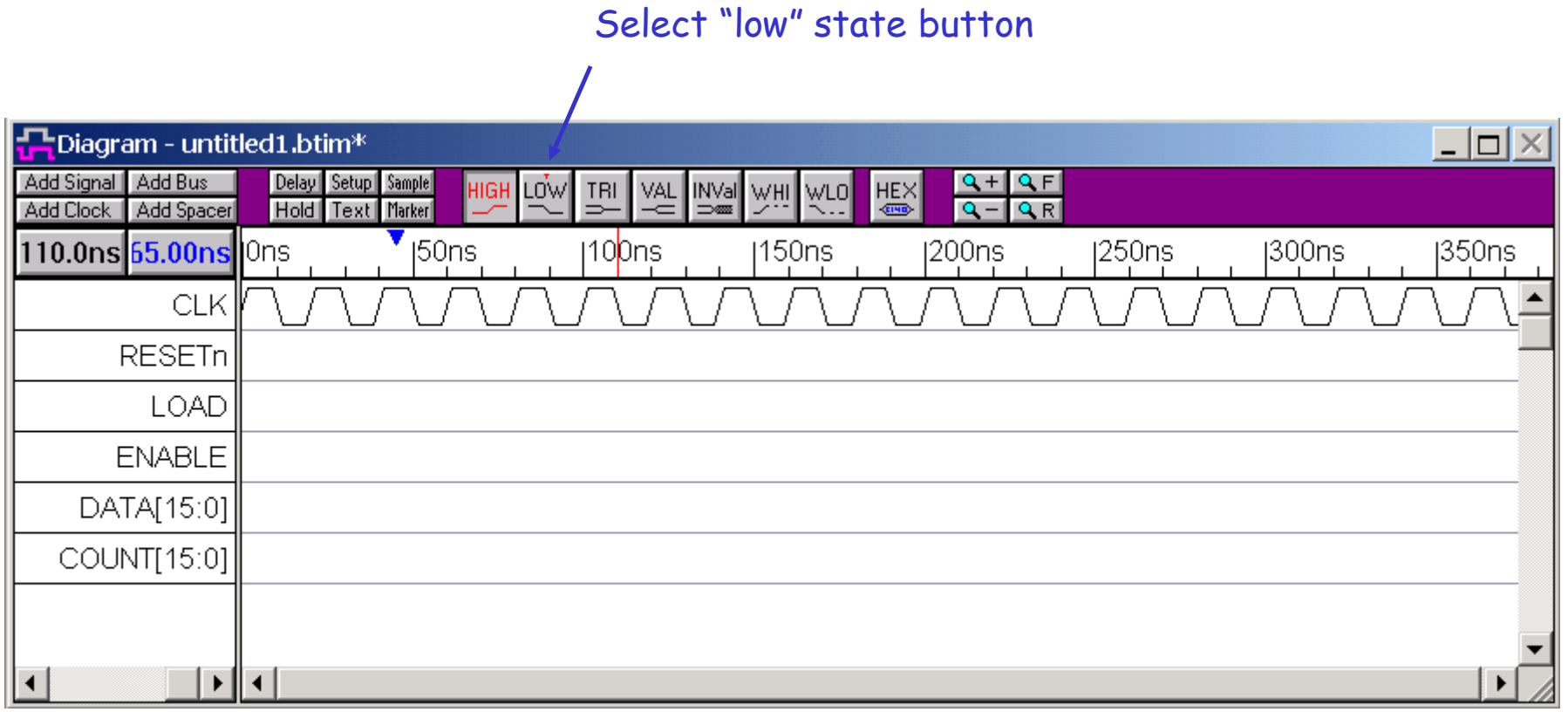
Creating Clocks



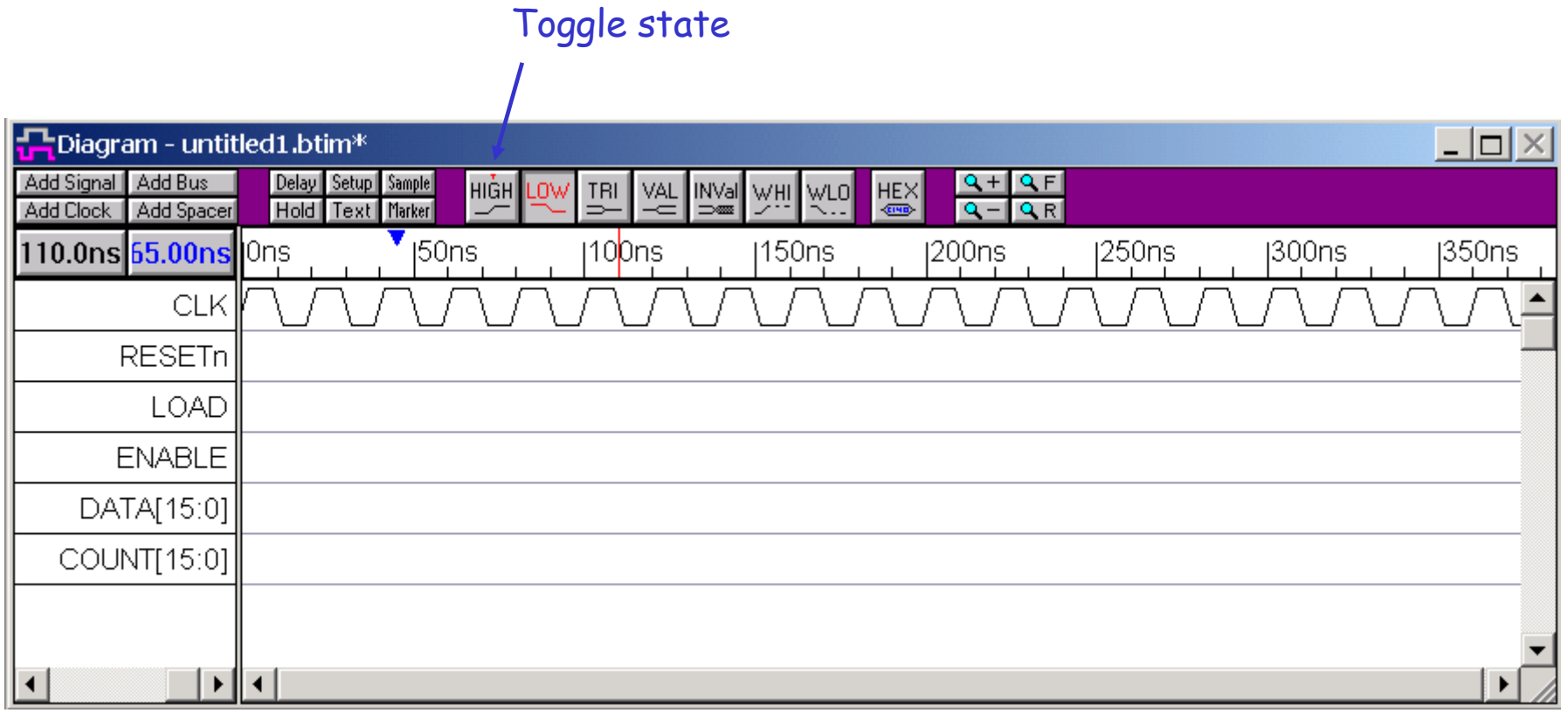
Creating Clocks



Drawing Signals

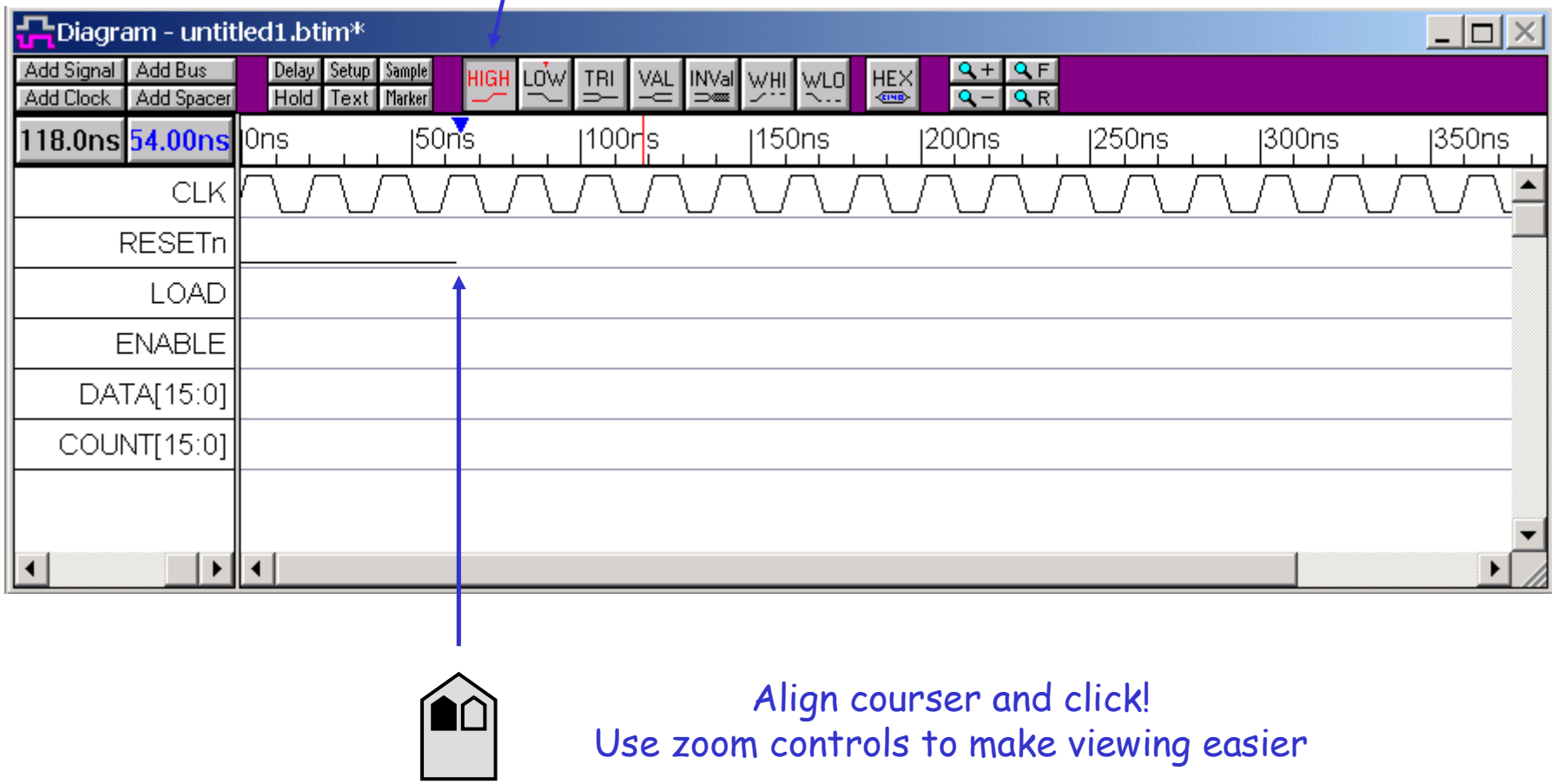


Drawing Signals



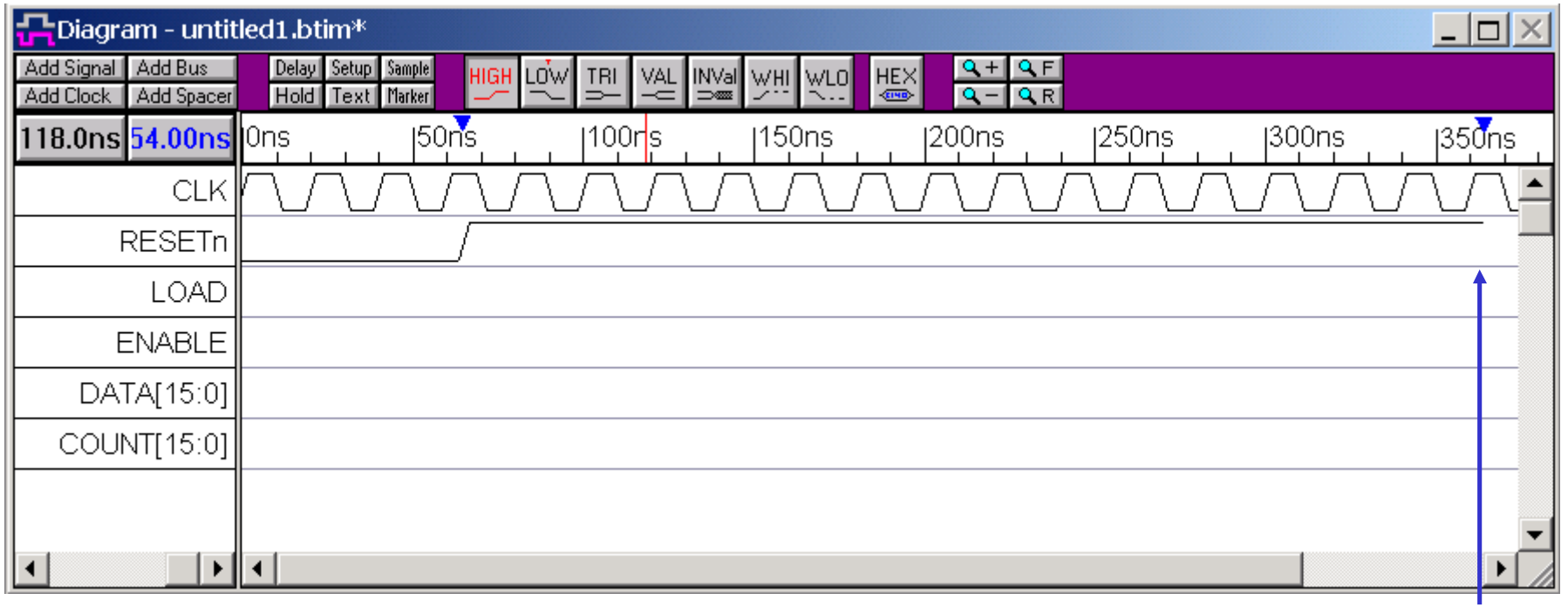
Drawing Signals

State button toggles automatically



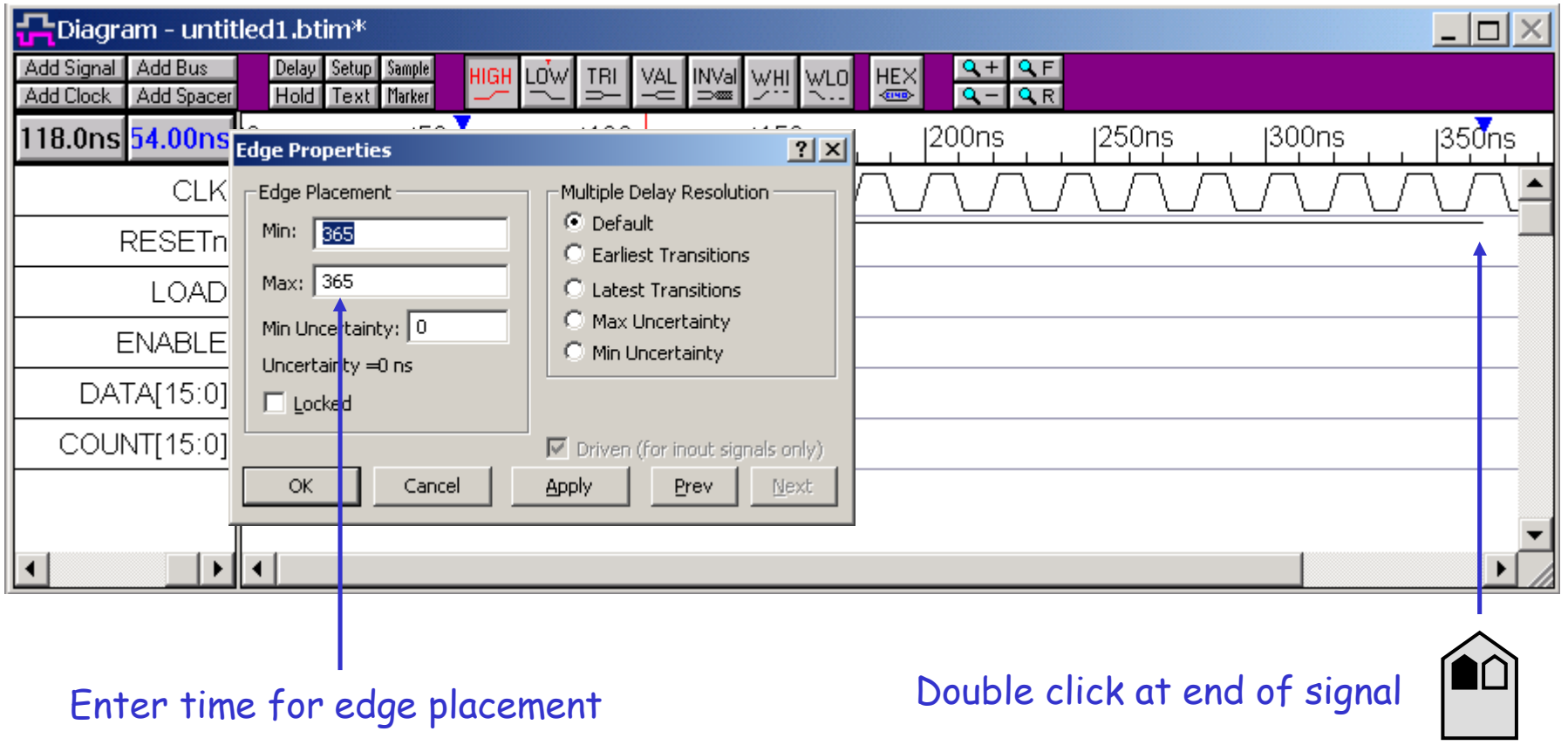
Align courser and click!
Use zoom controls to make viewing easier

Drawing Signals



Align courser and click!
Use zoom controls to make viewing easier

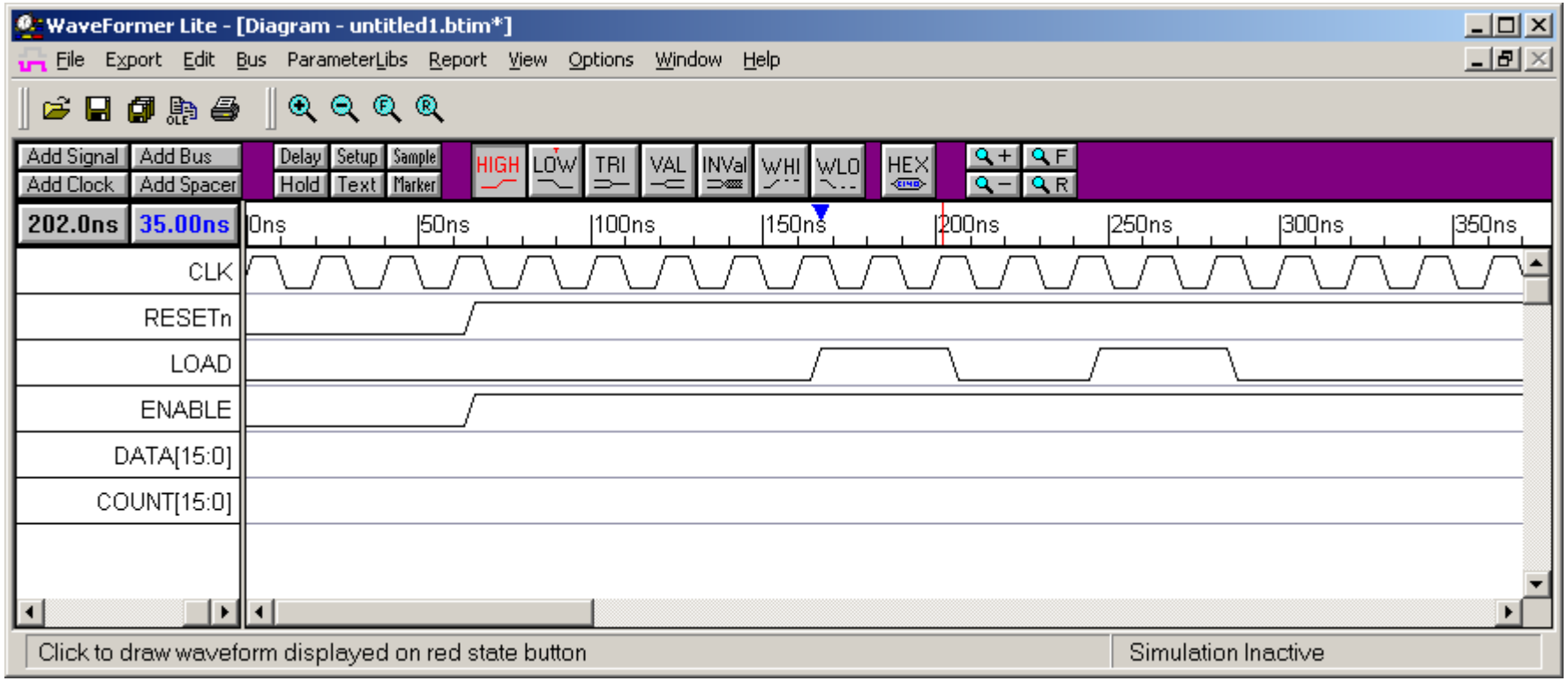
Drawing Signals
Edge Placement



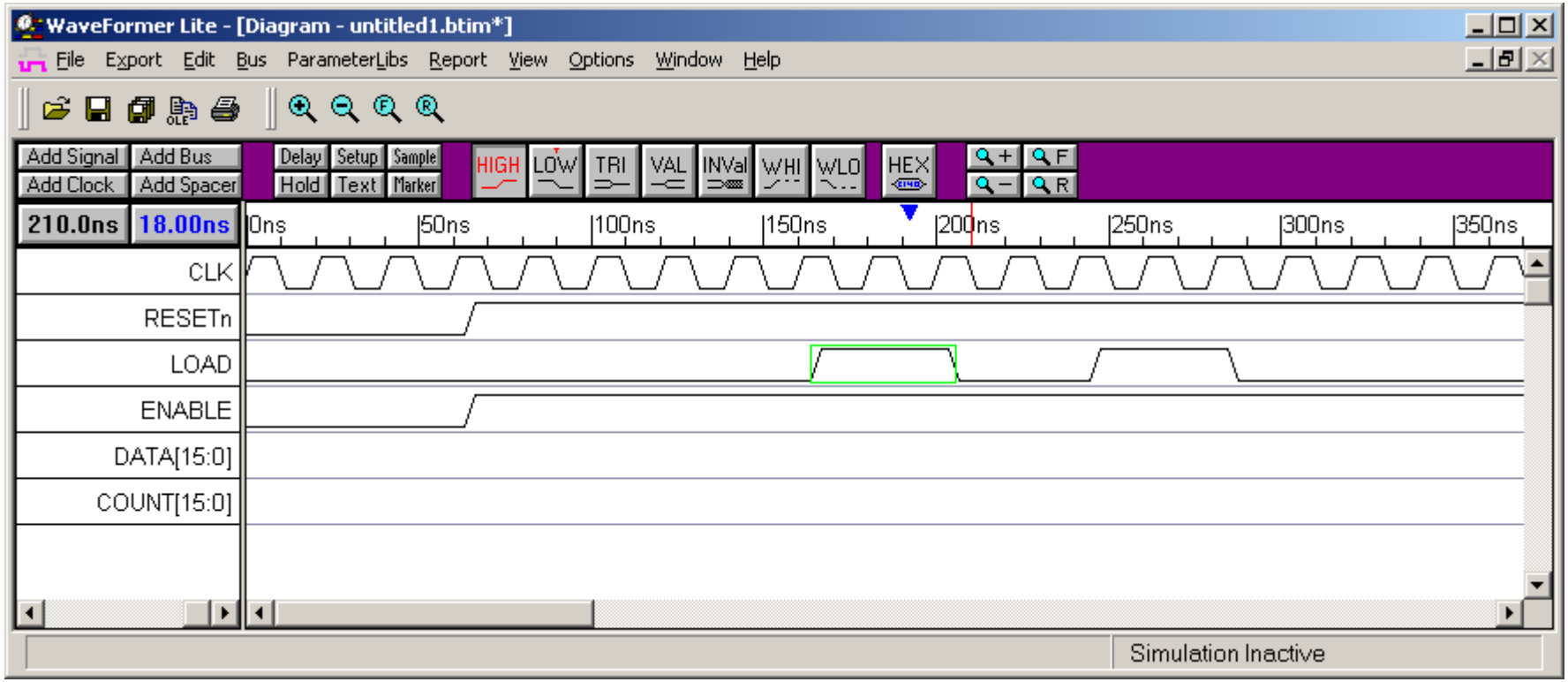
Enter time for edge placement

Double click at end of signal

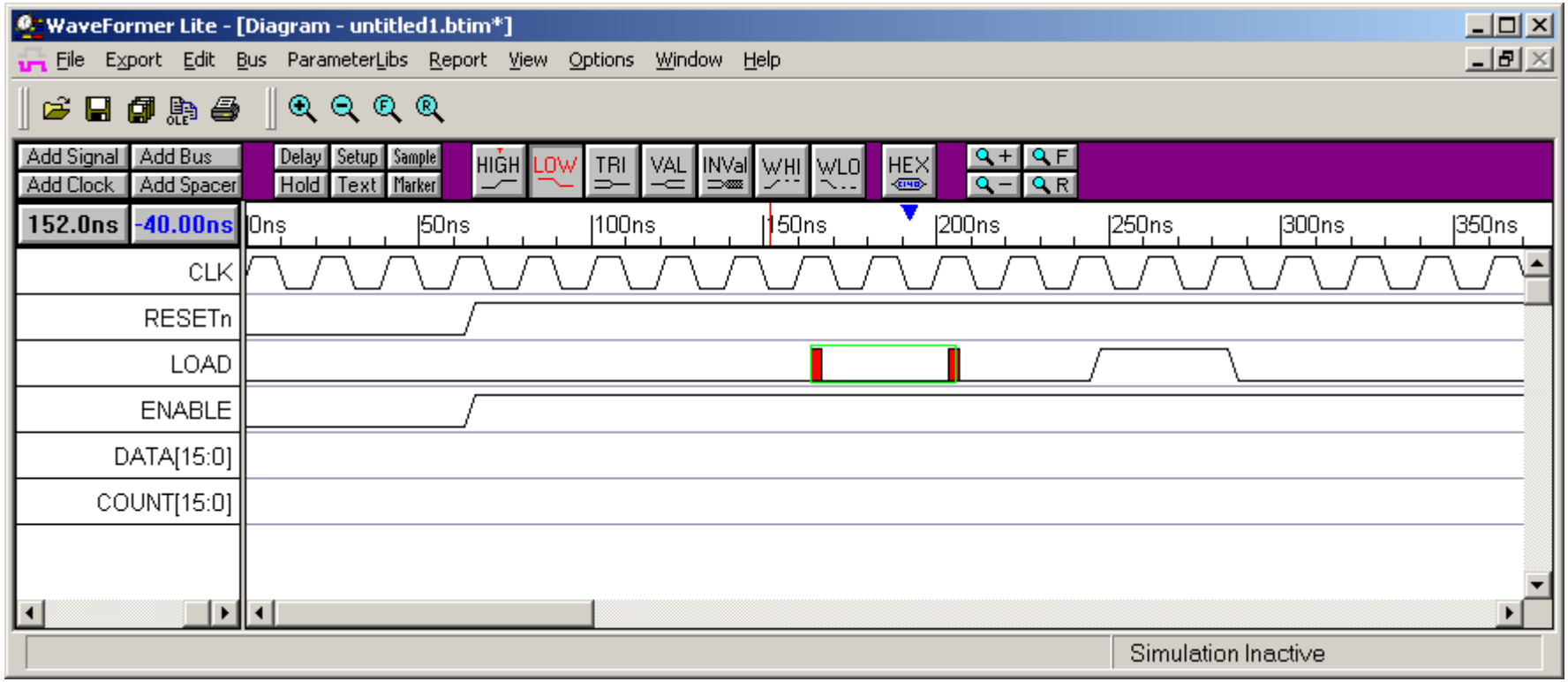
Editing Signals



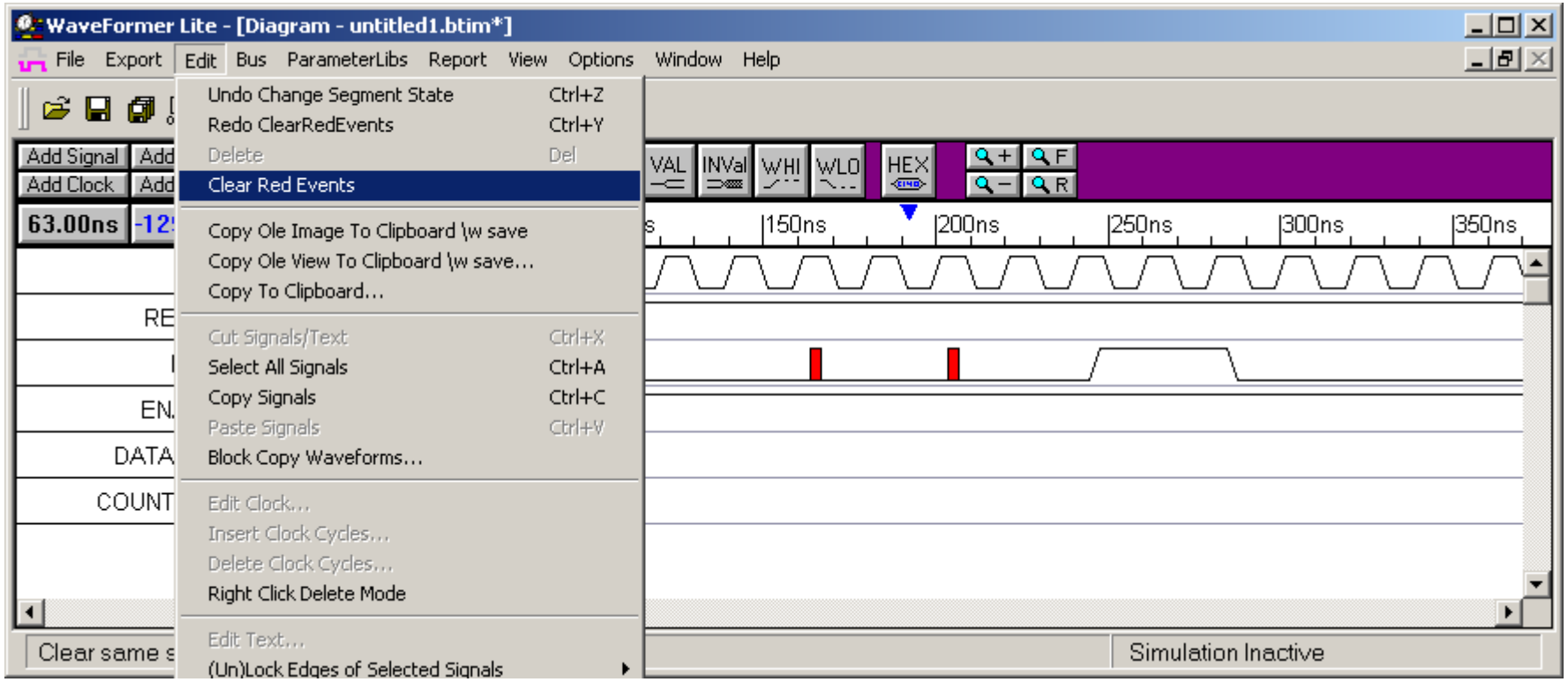
Editing Signals



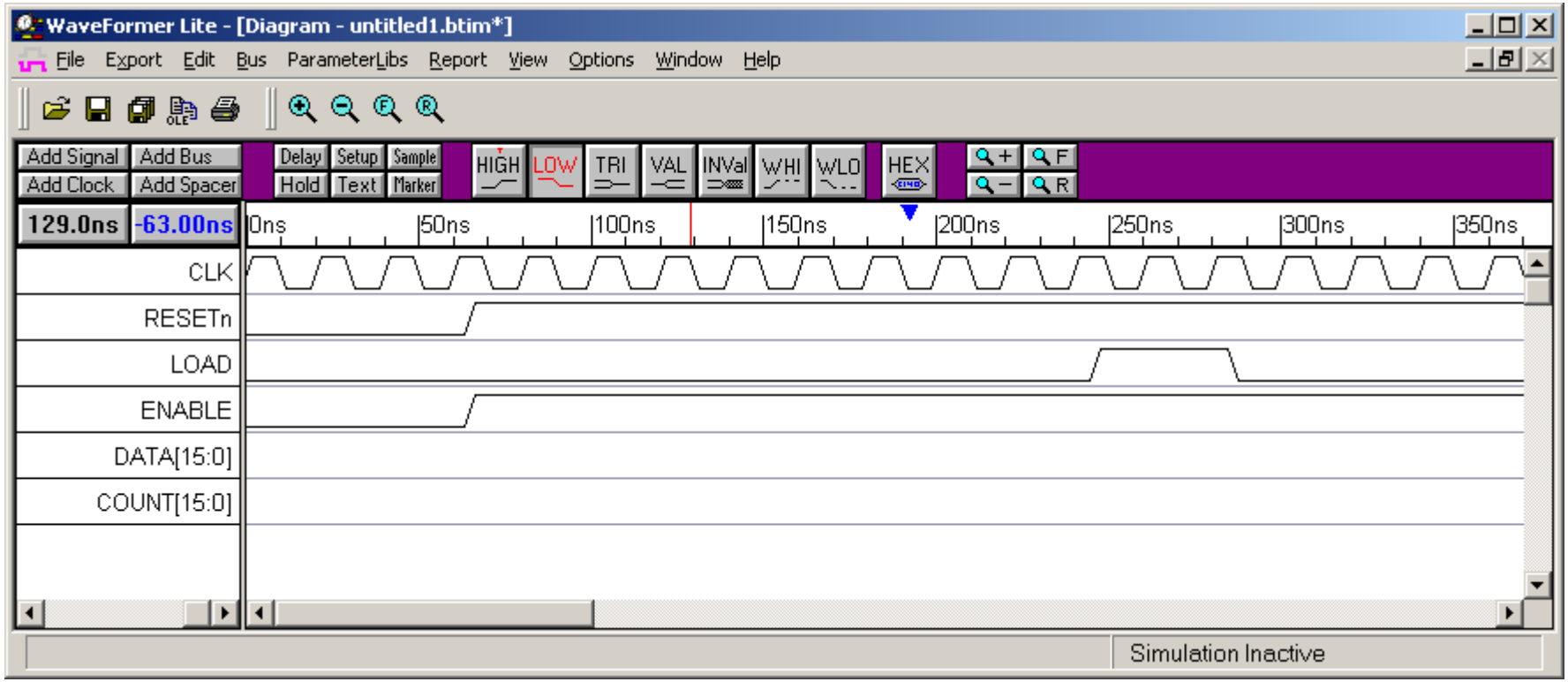
Editing Signals



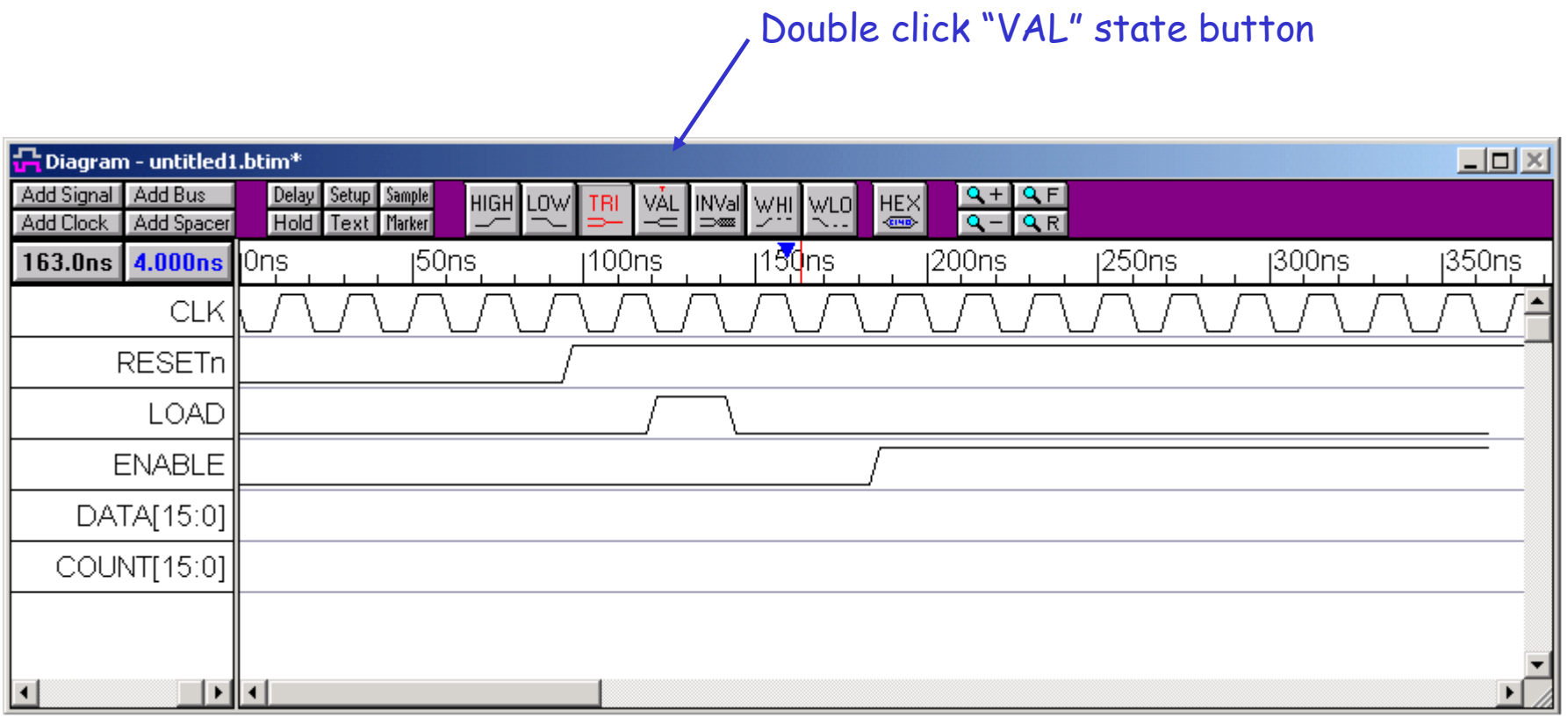
Editing Signals



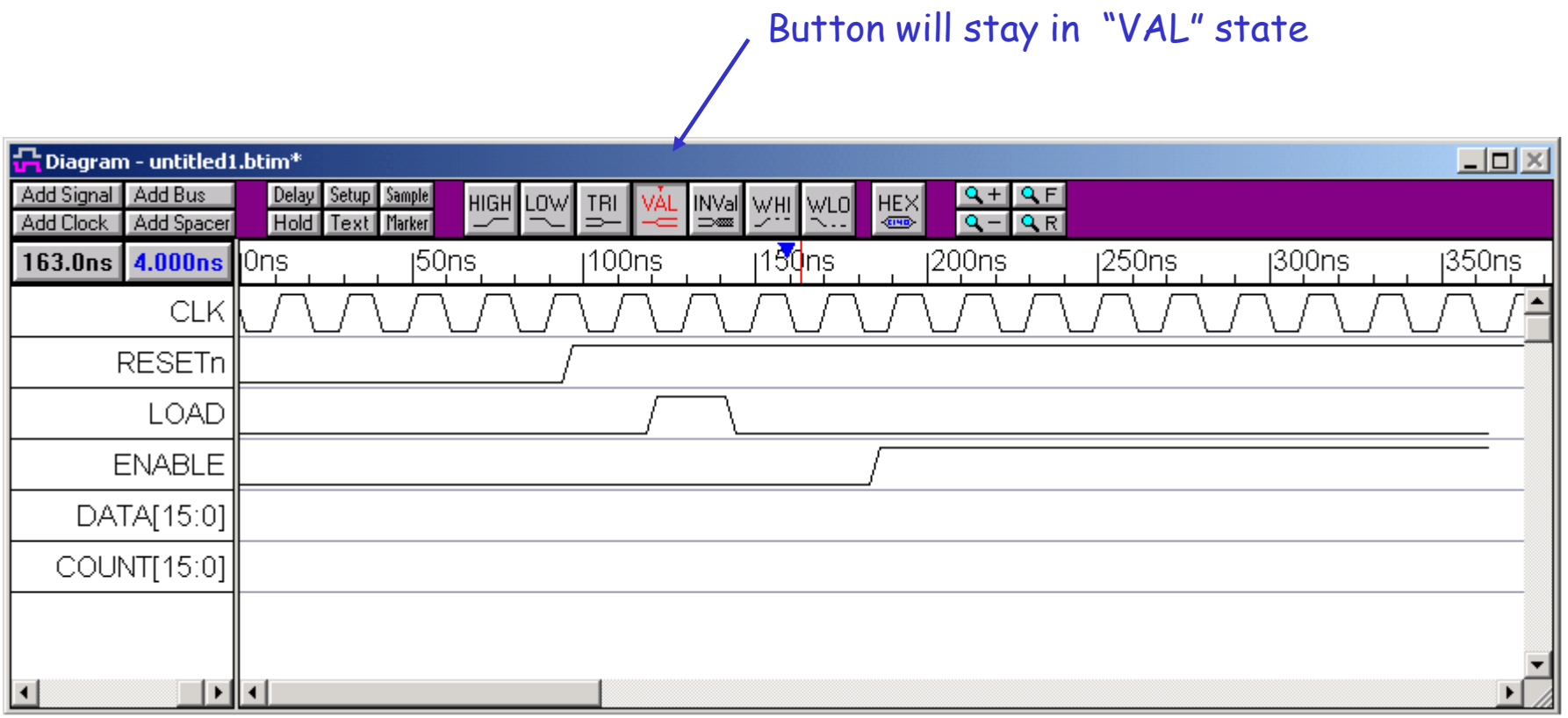
Editing Signals



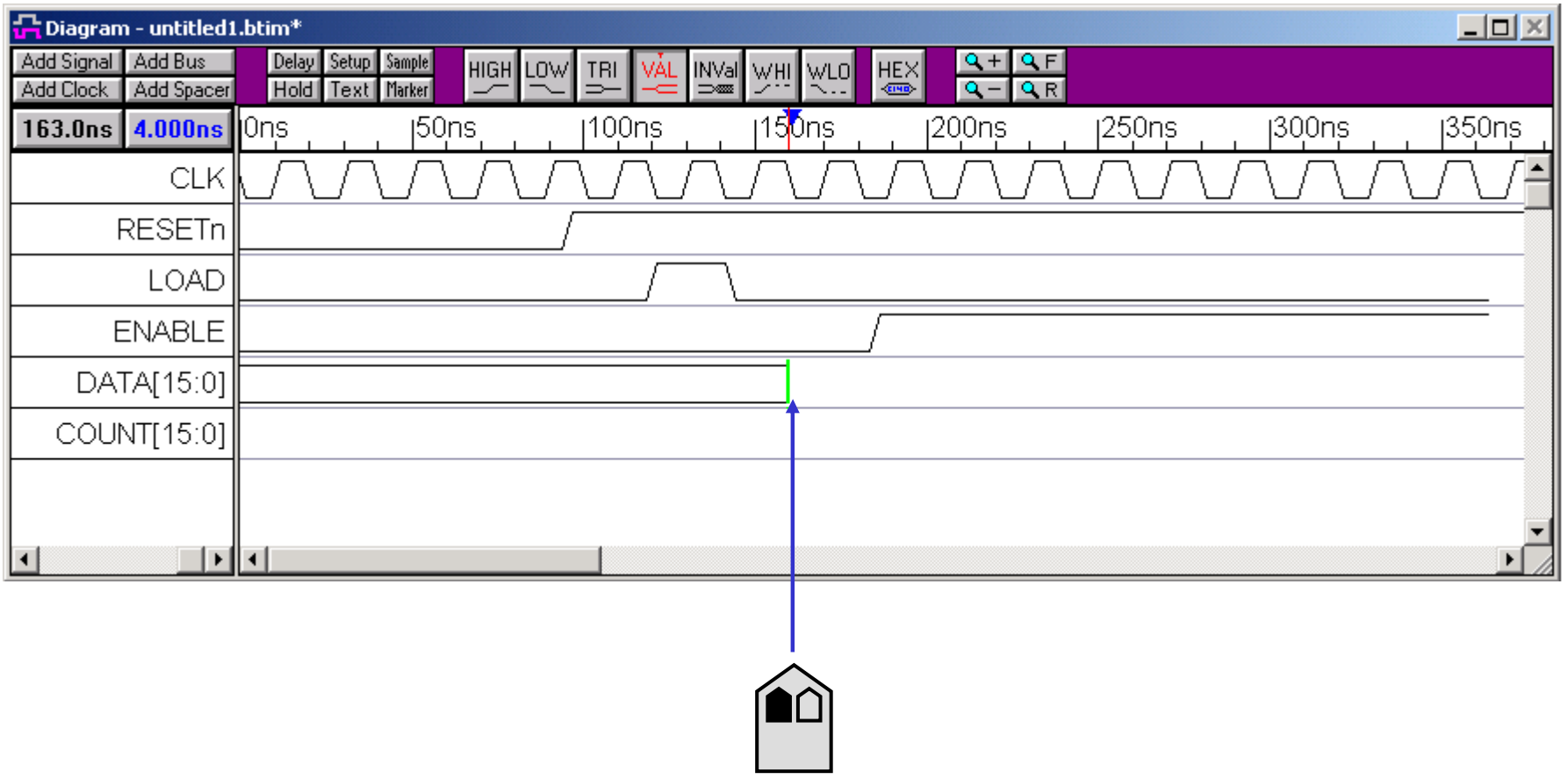
Drawing Busses



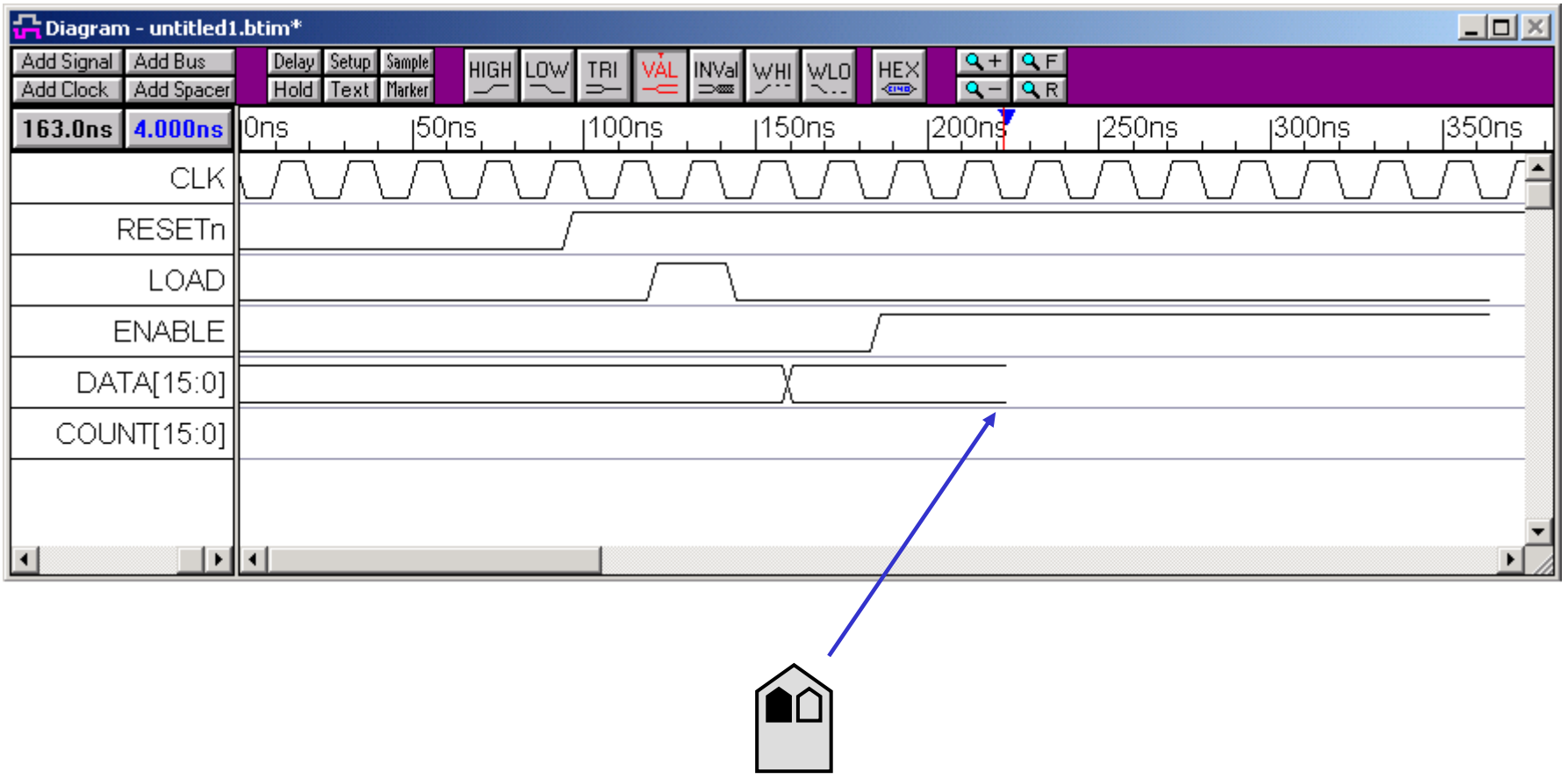
Drawing Busses



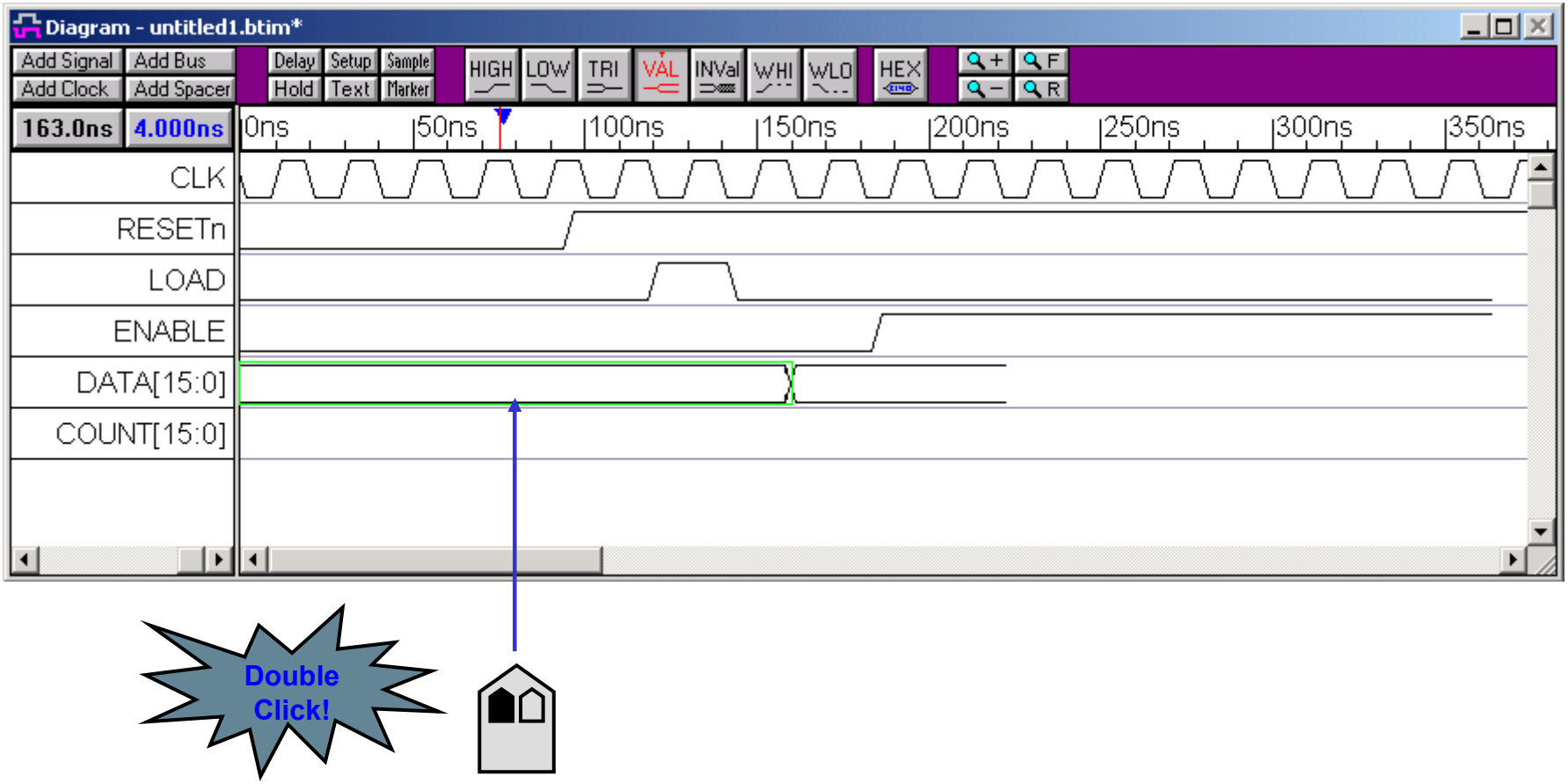
Drawing Busses



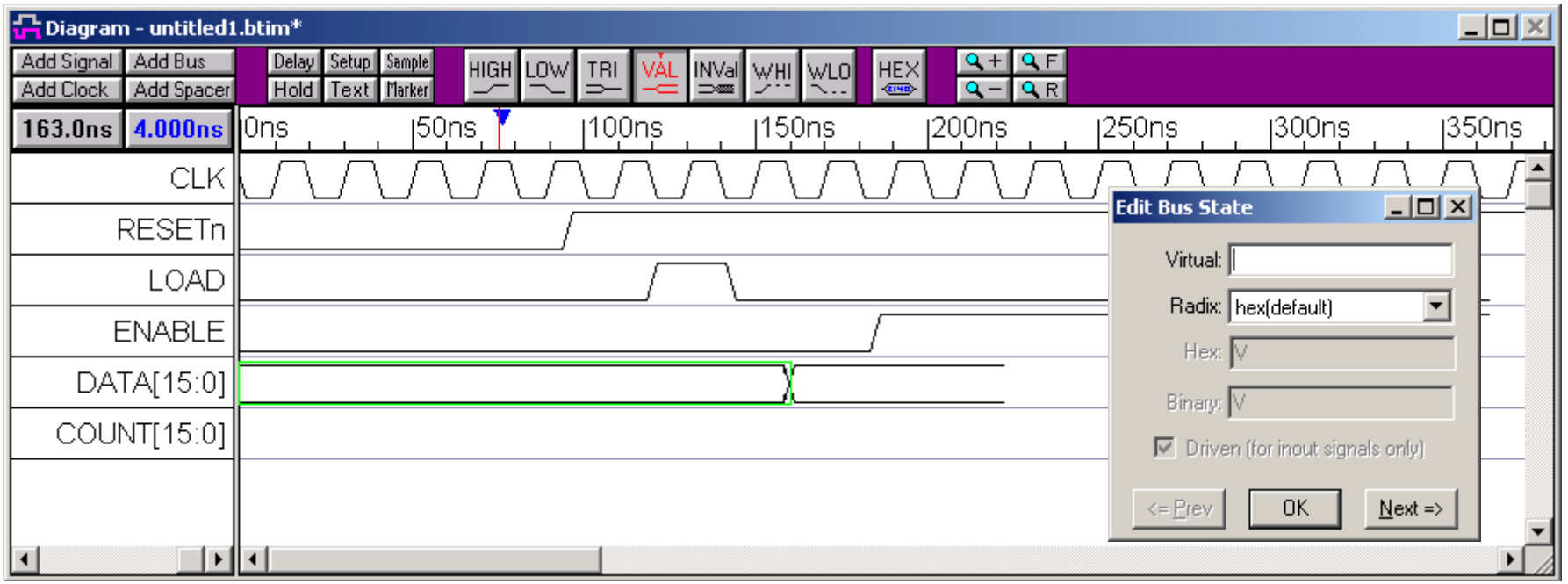
Drawing Busses



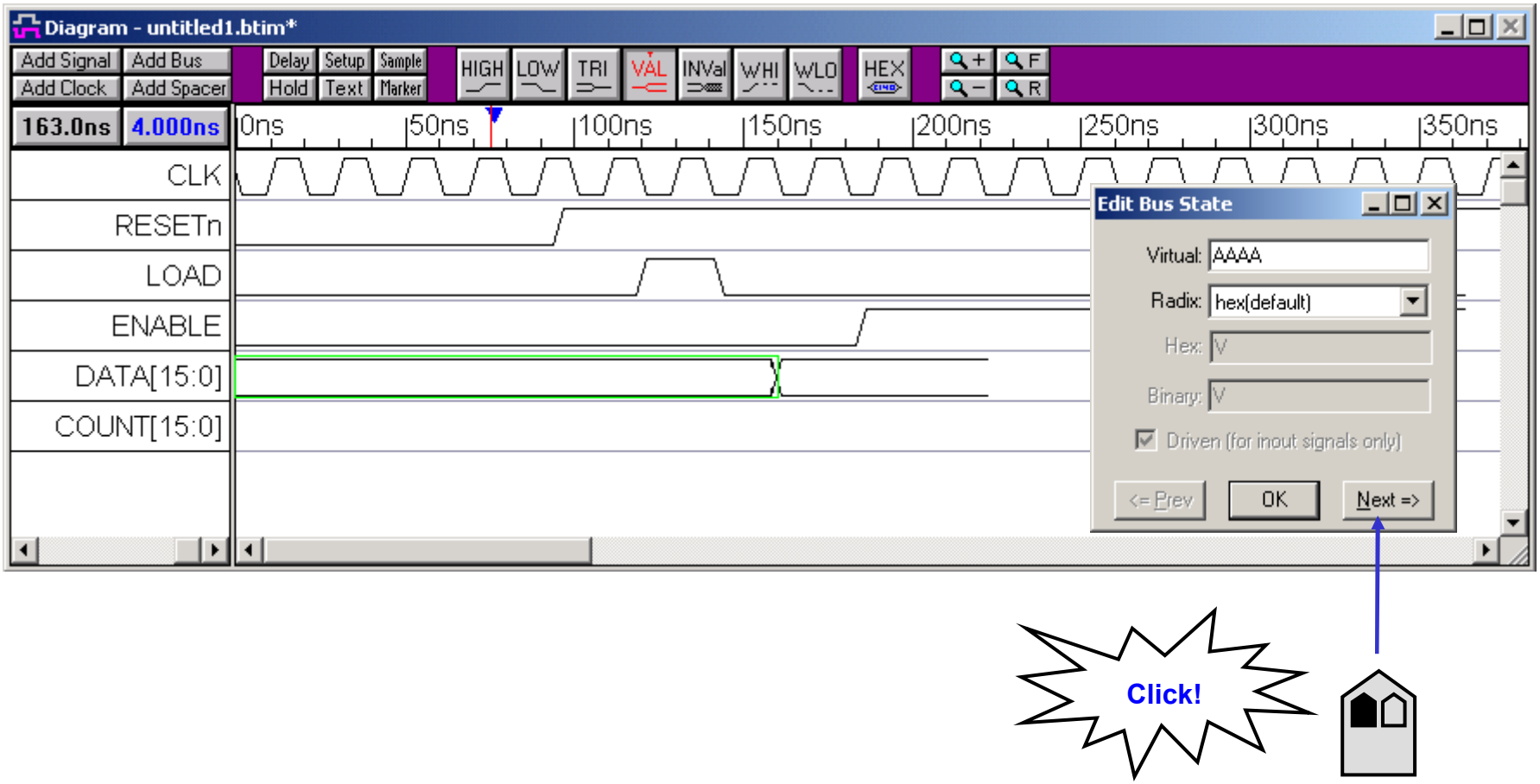
Drawing Busses



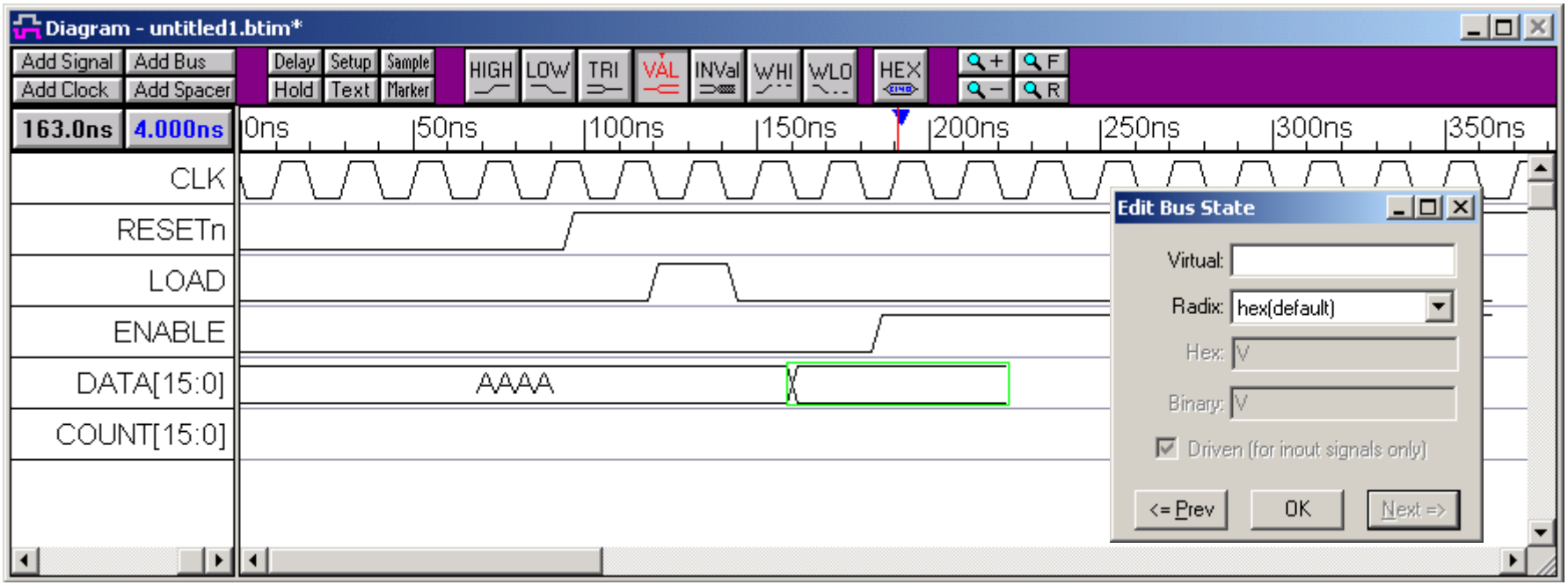
Drawing Busses



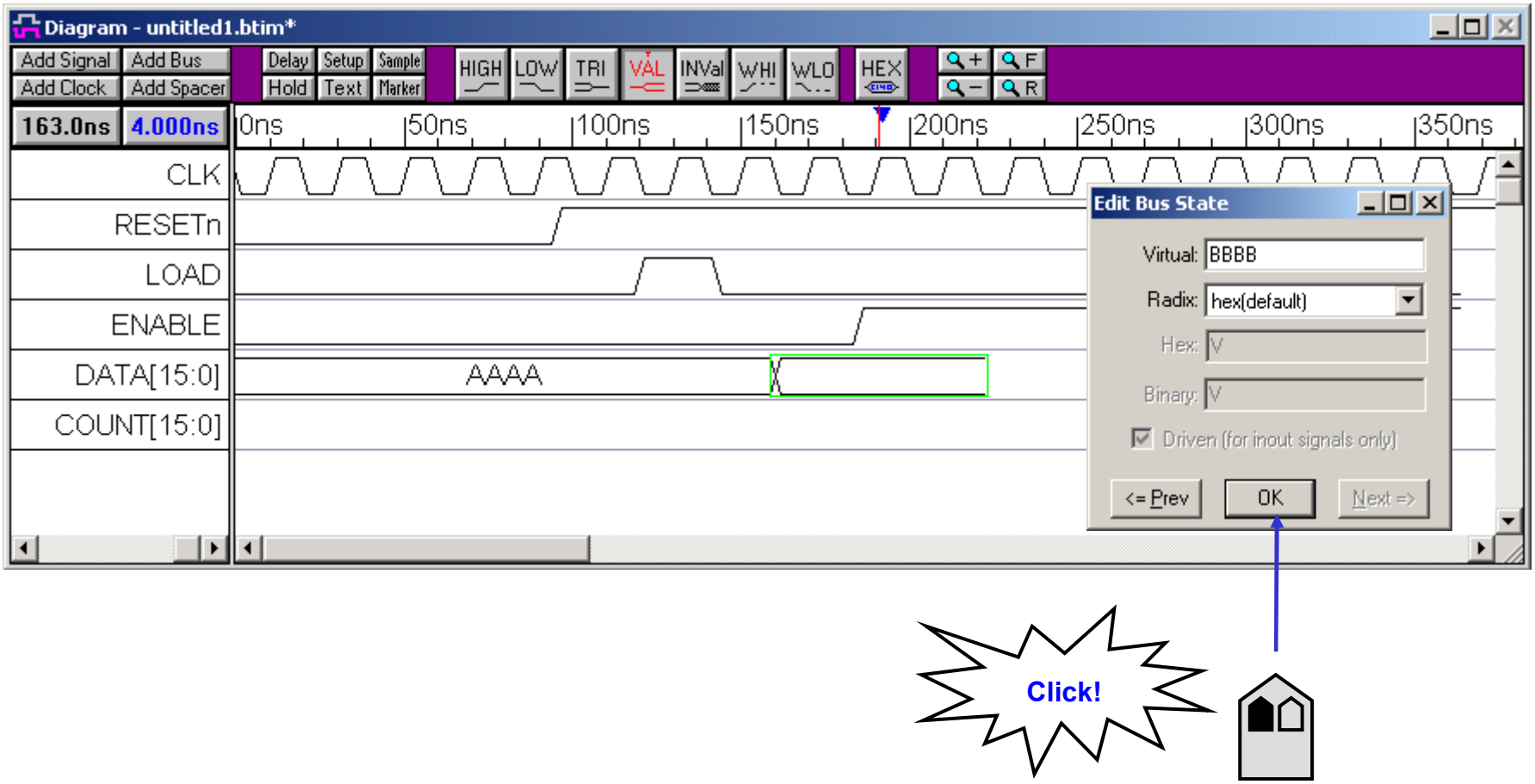
Drawing Busses



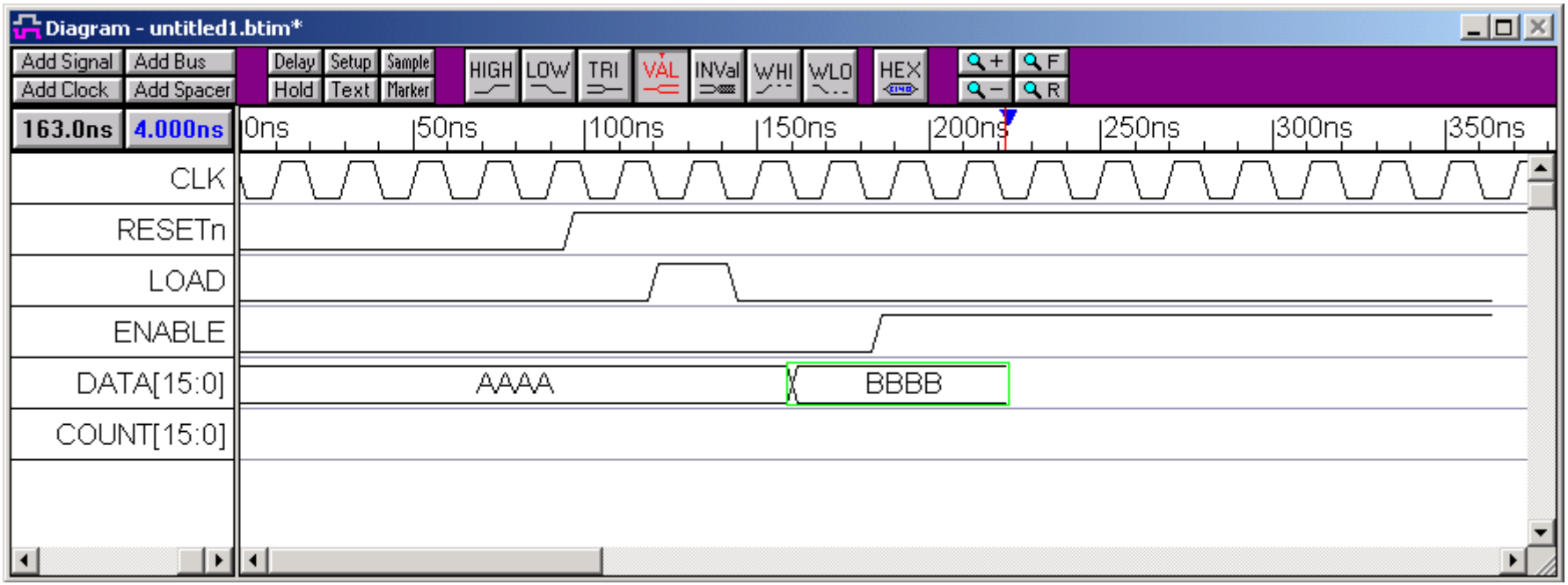
Drawing Busses



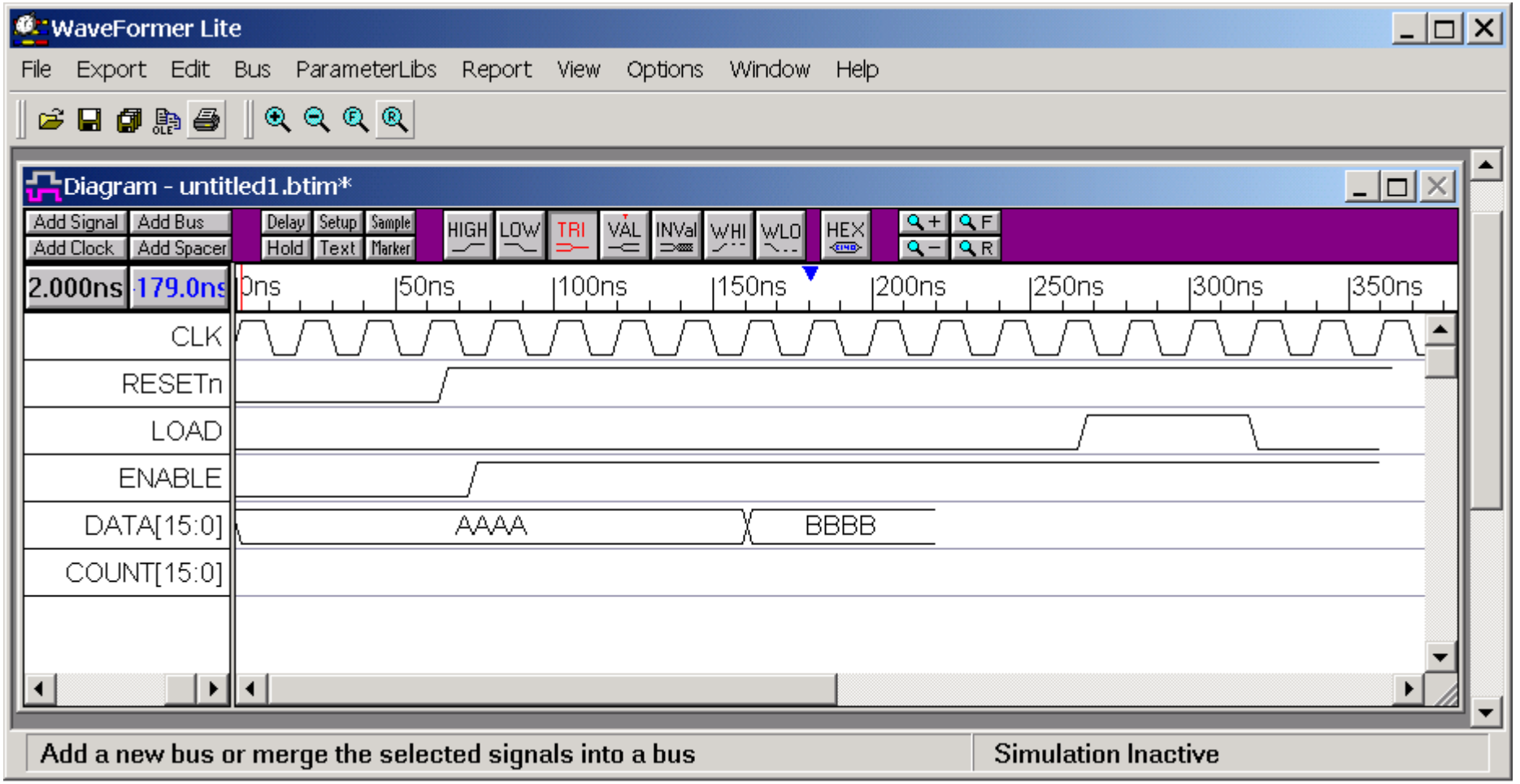
Drawing Busses



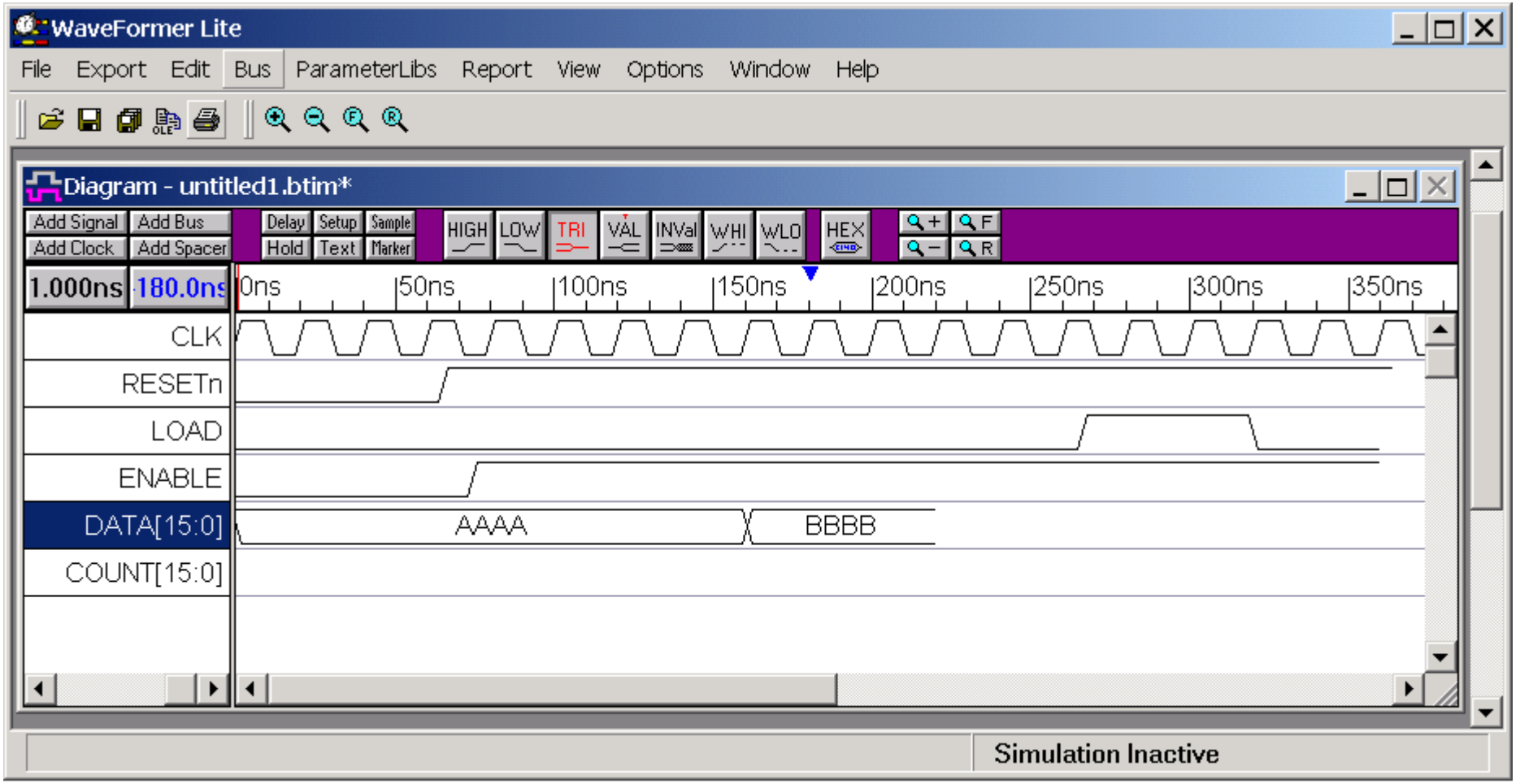
Drawing Busses



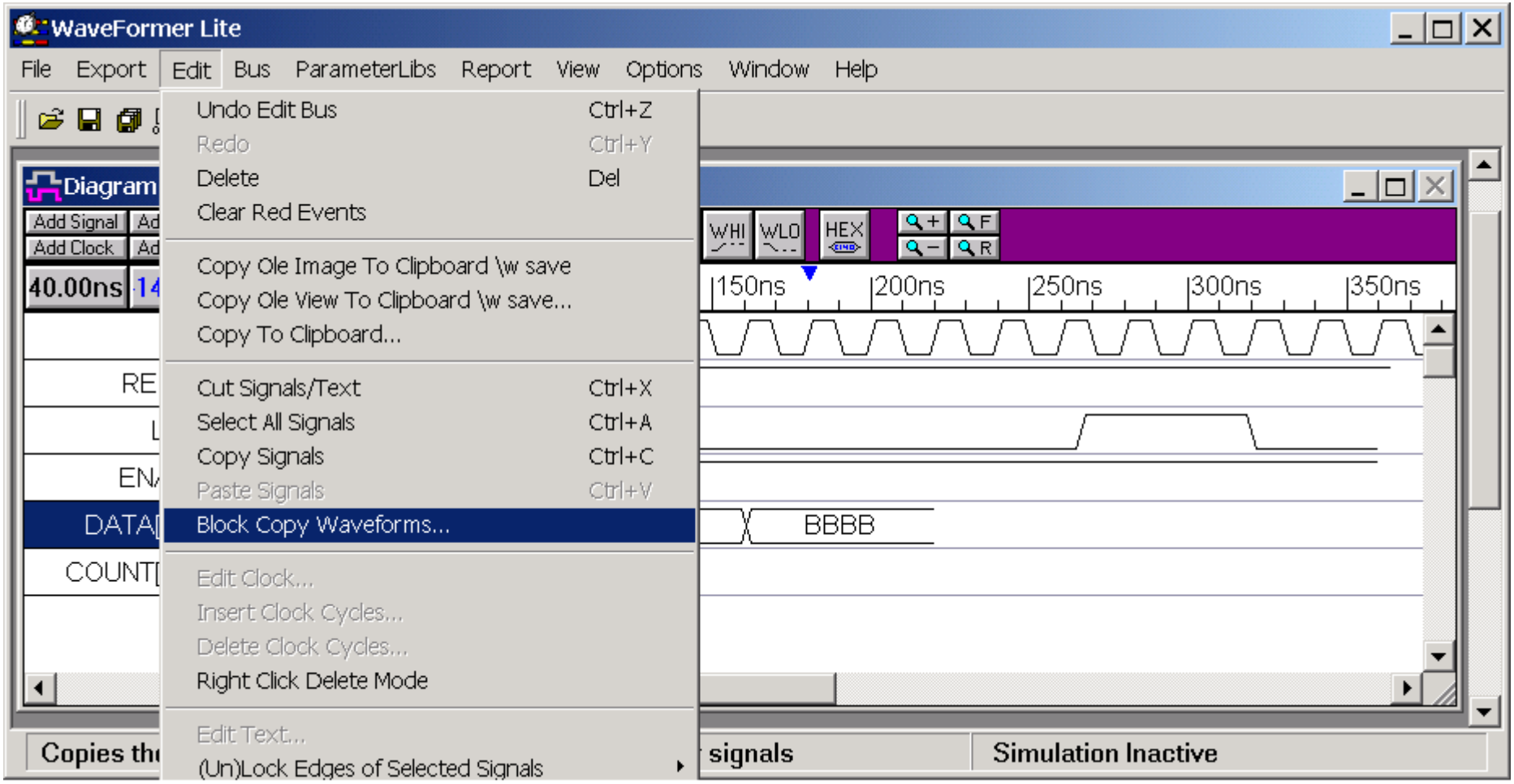
Editing Signals
Appending and Inserting



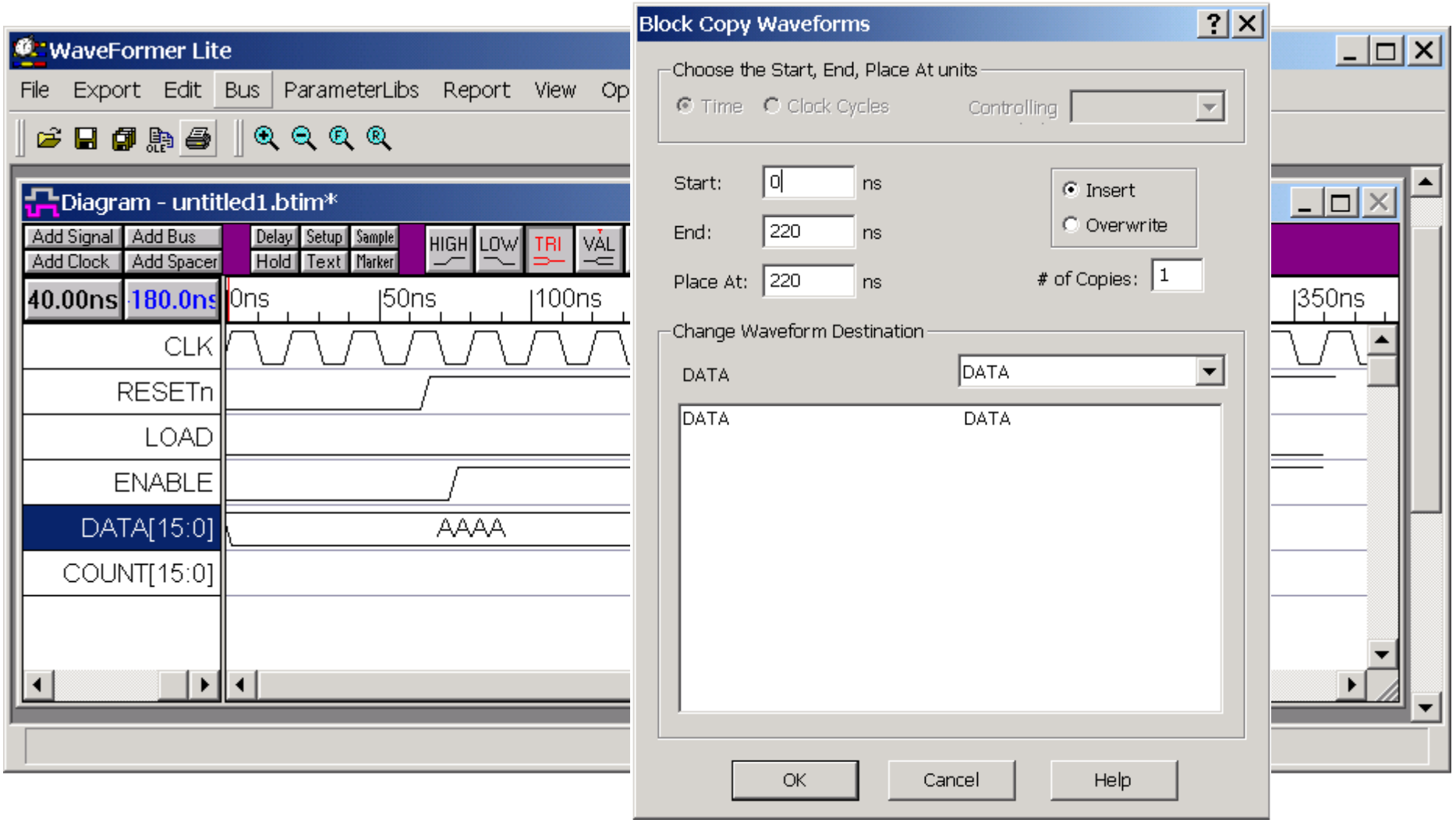
Editing Signals
Appending and Inserting



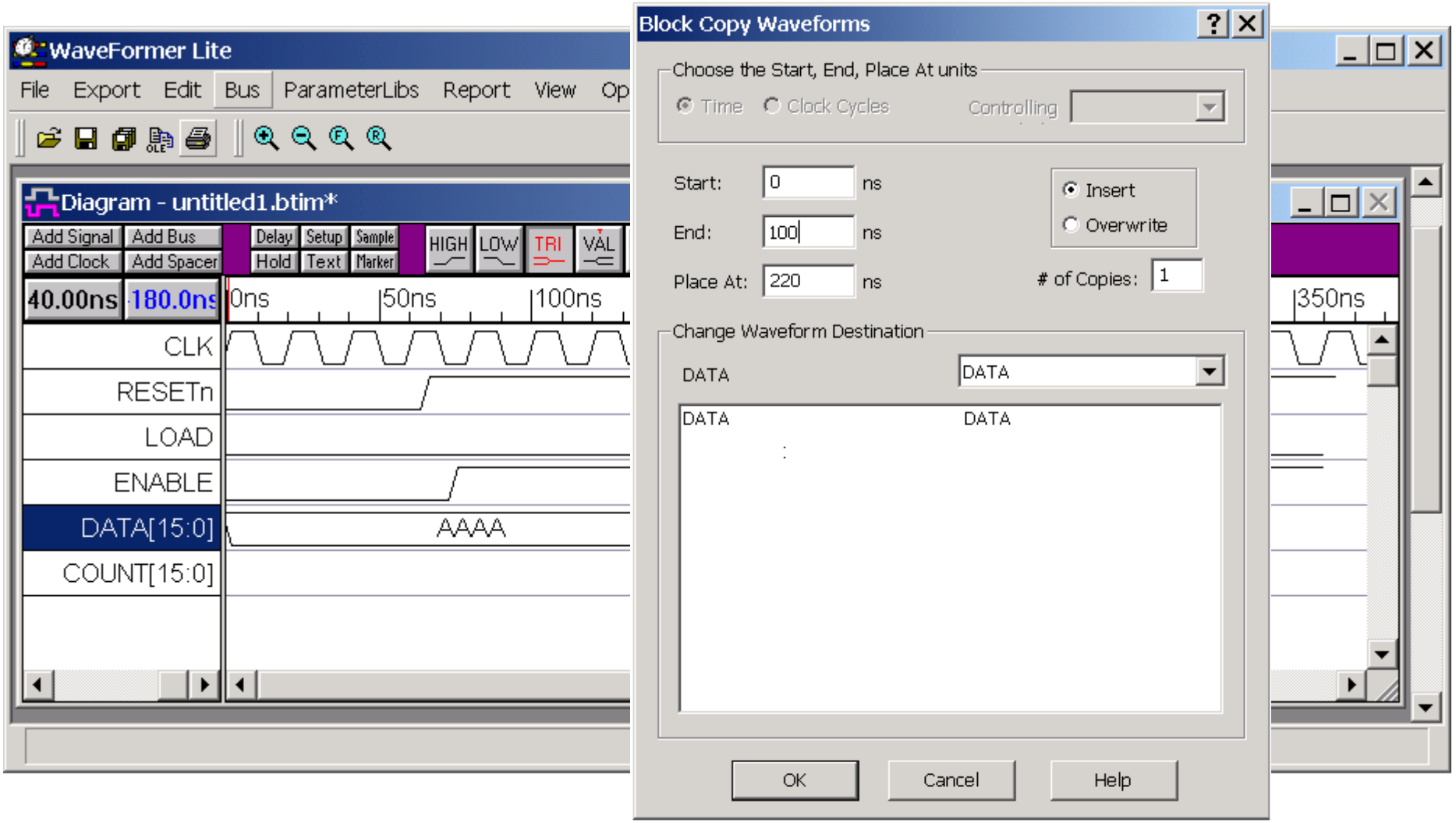
Editing Signals
Appending and Inserting



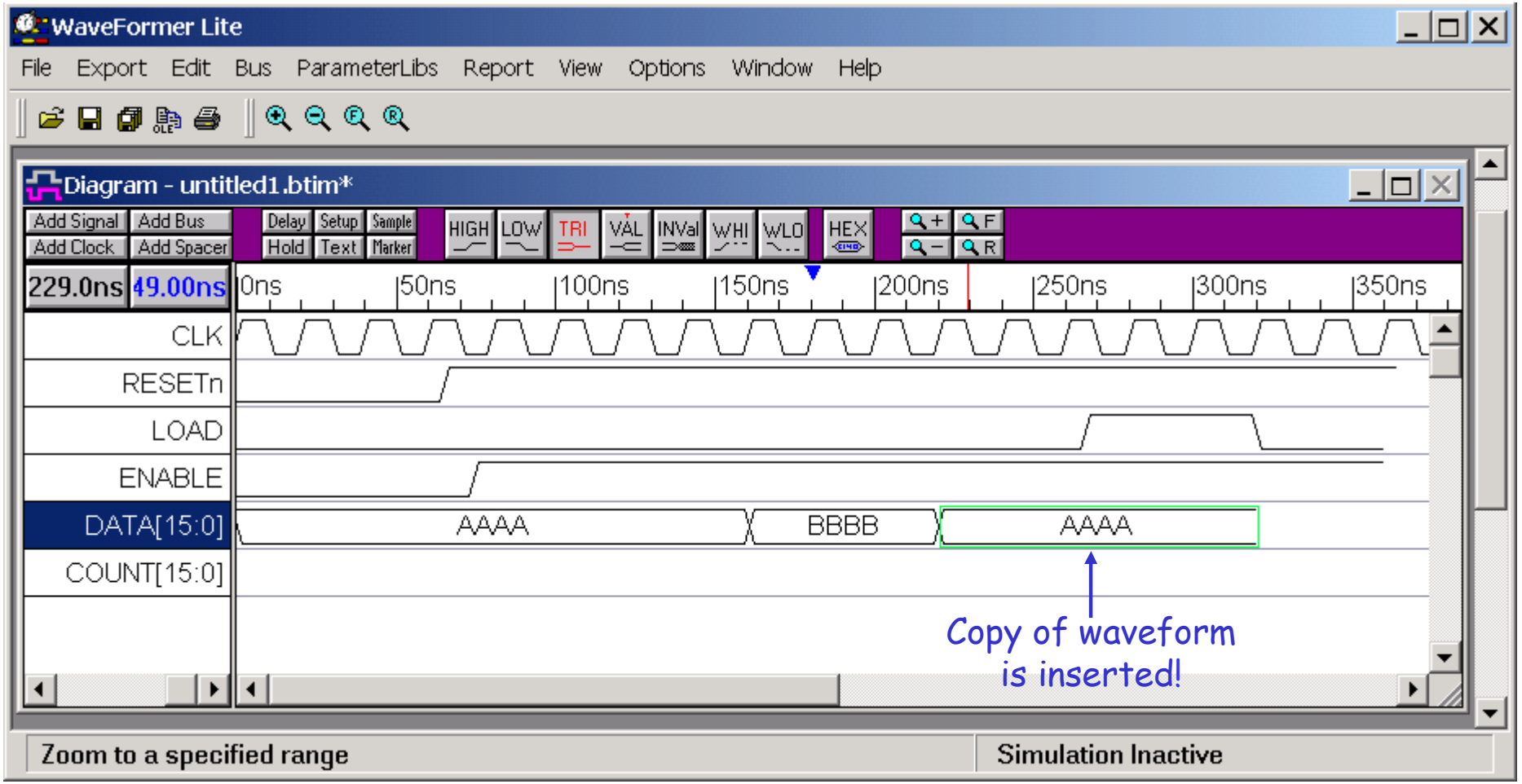
Editing Signals
Appending and Inserting



Editing Signals
Appending and Inserting

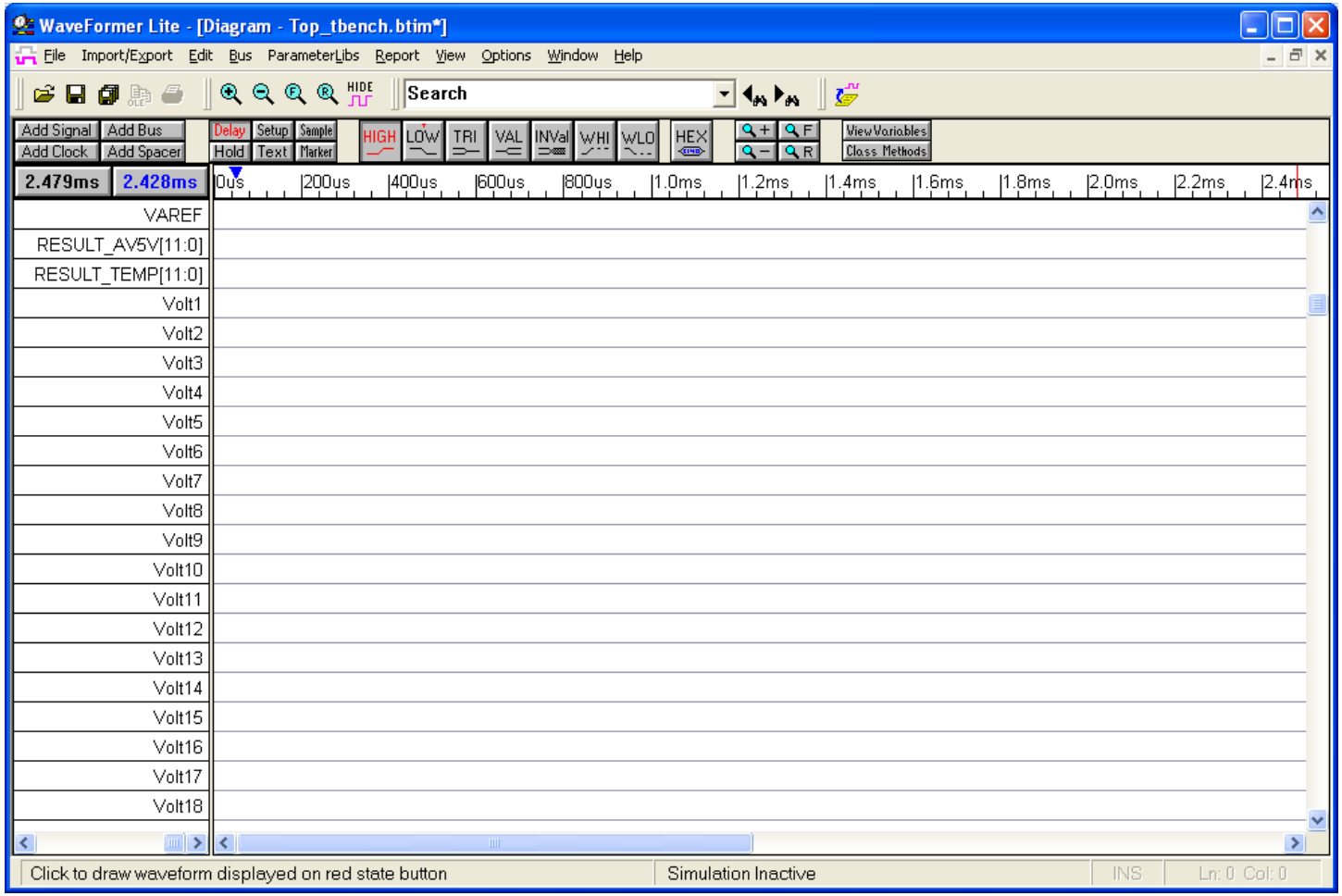


Editing Signals
Appending and Inserting



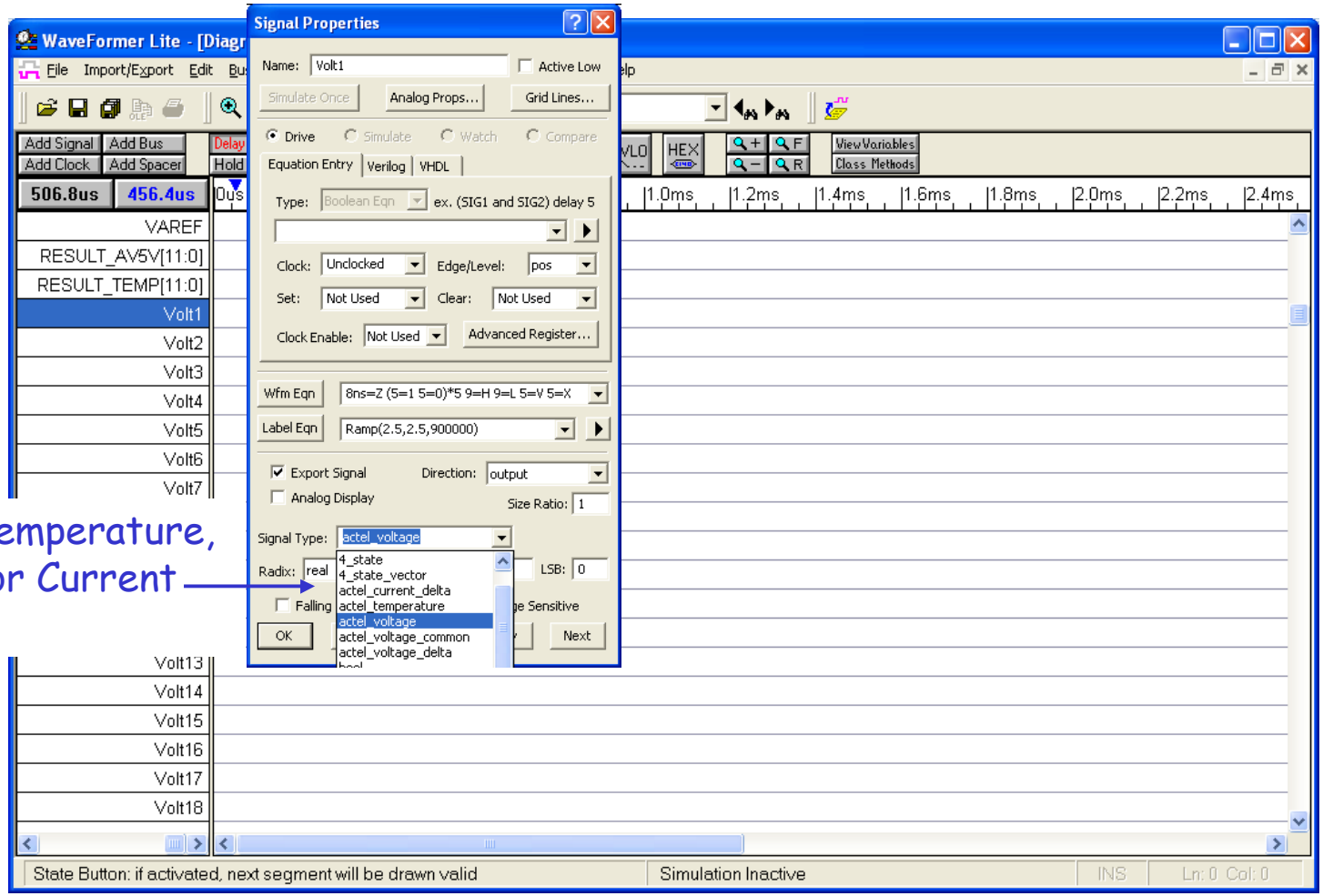
Fusion Analog Signals

- WaveFormer Lite Supports Creation of Analog Stimulus for Fusion FPGAs



Fusion Analog Signals

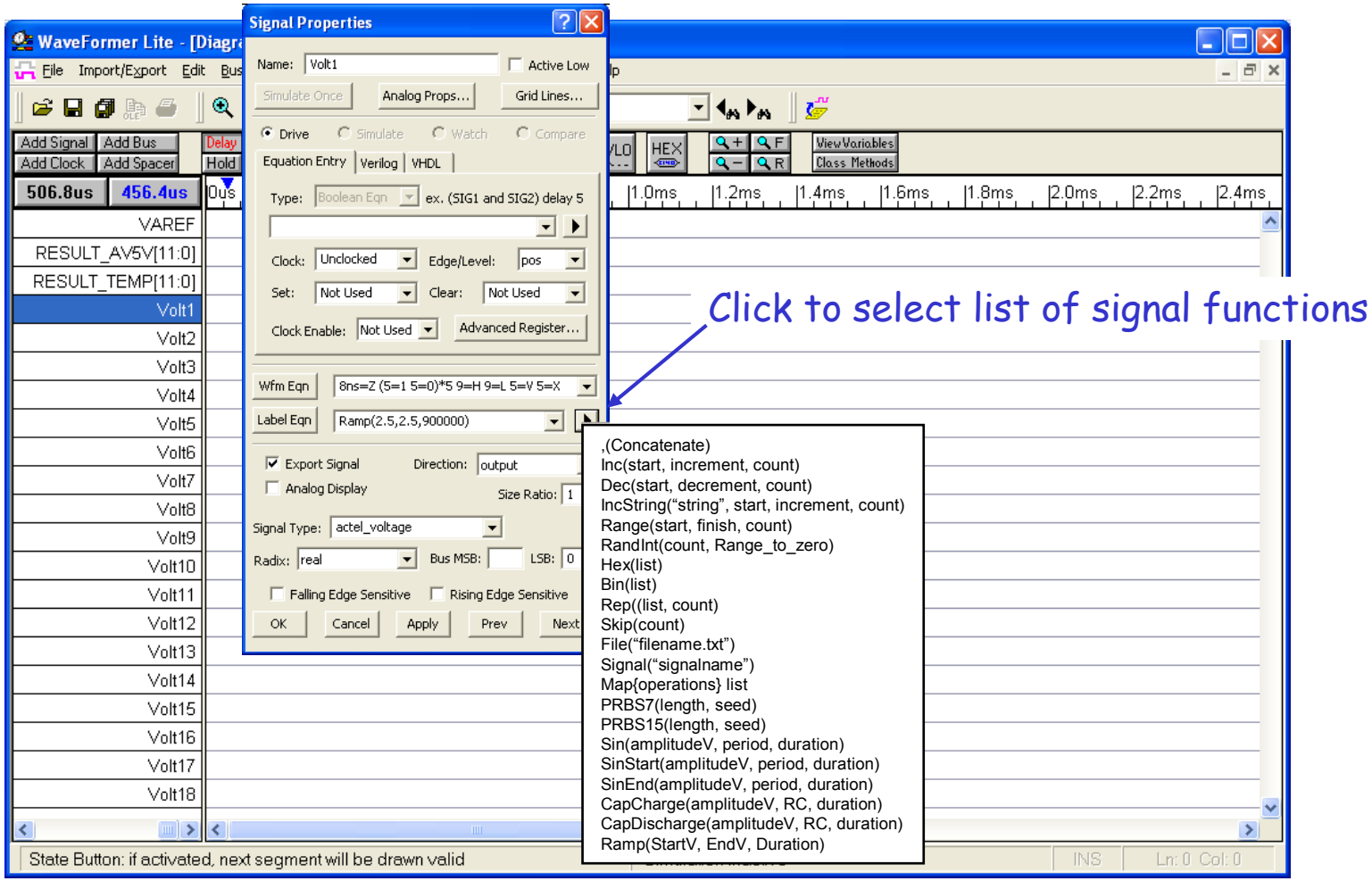
- Specify Analog Signal Type



Select Temperature, Voltage or Current from list

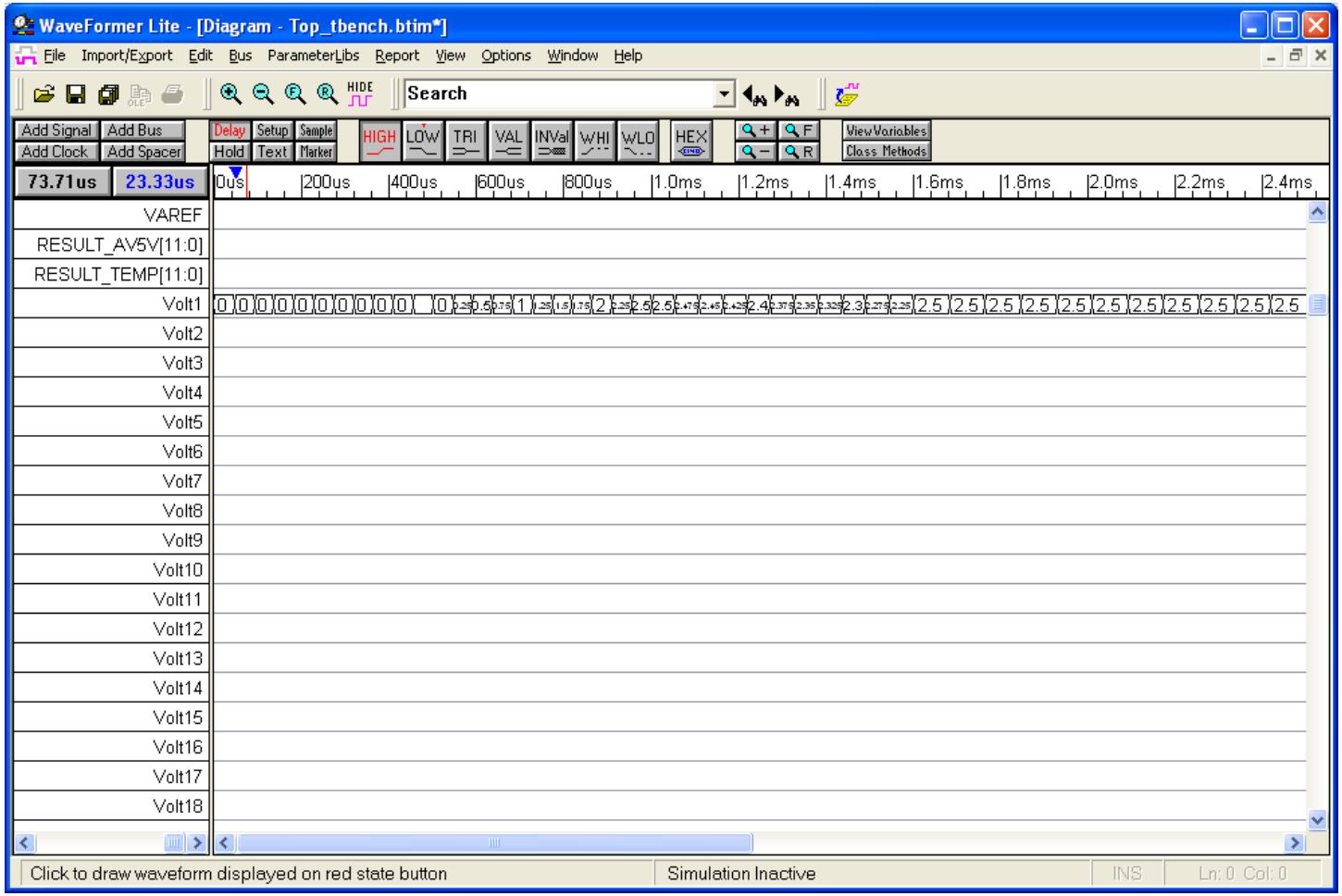
Fusion Analog Signals

- Using Built-in Functions



Fusion Analog Signals

- Analog Stimulus

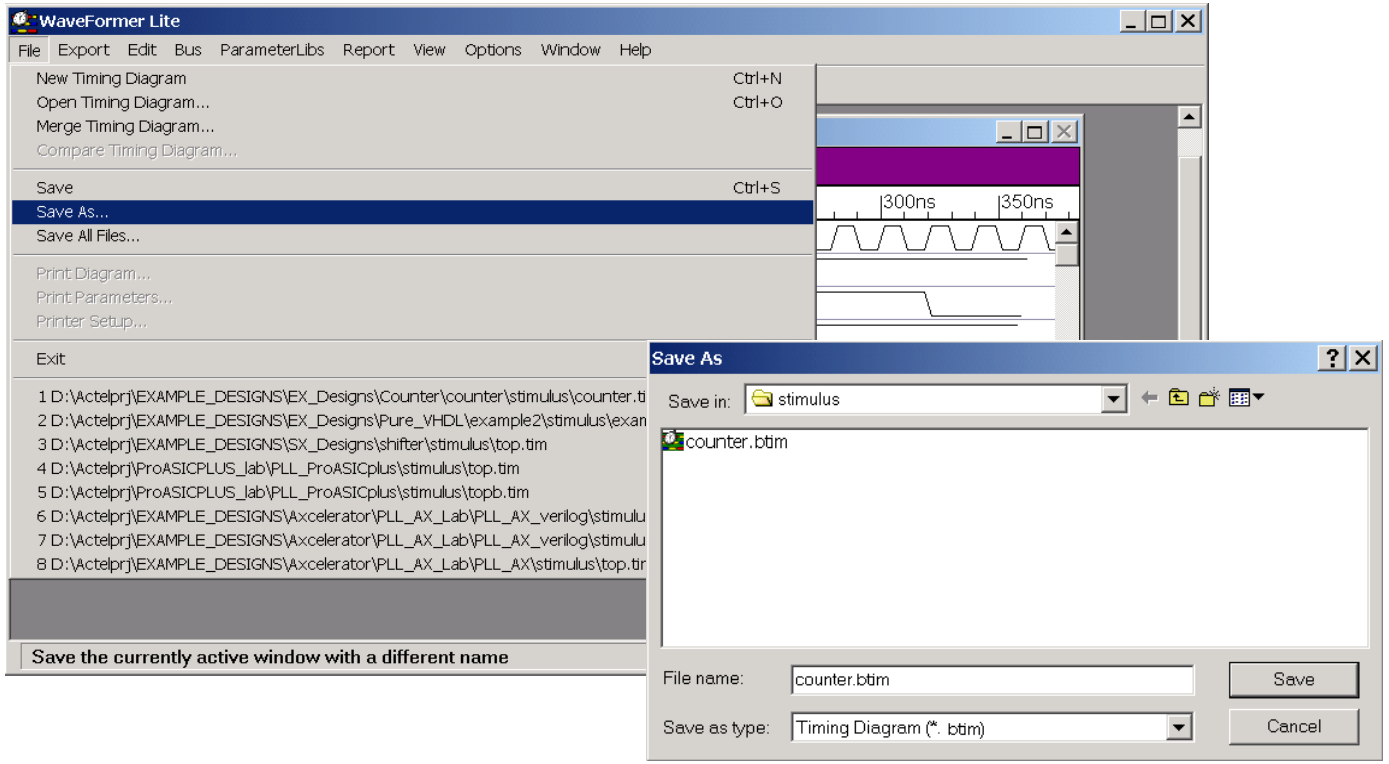


Saving Stimulus

- Save Stimulus

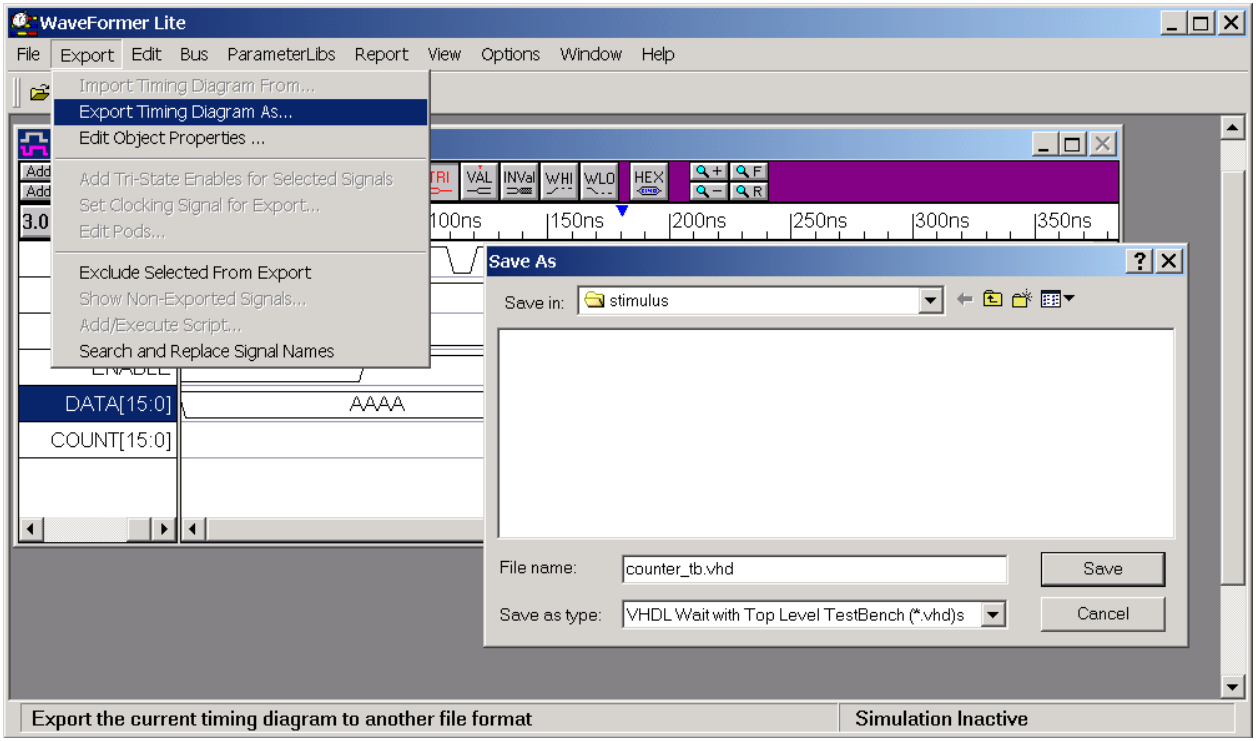
  File > Save

  - File Name May Contain Name of Top-level Module
  - Stimulus Appears on Libero File Manager Tab

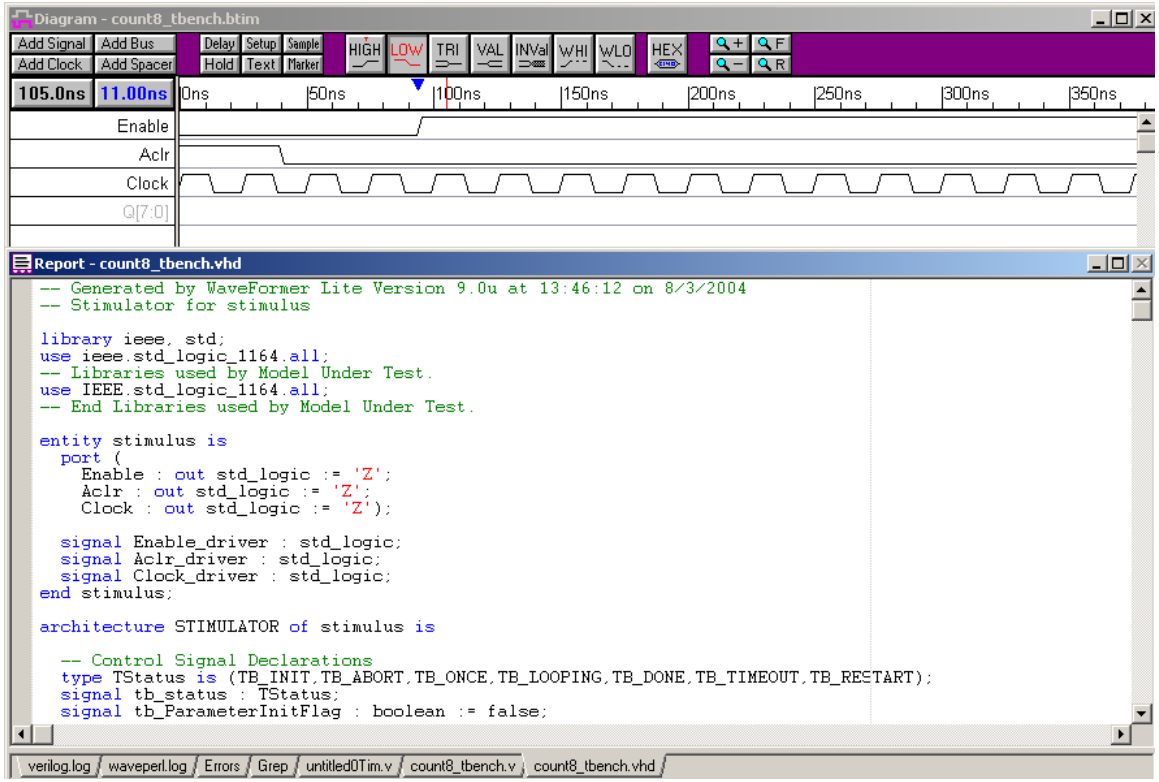


Generating the Testbench

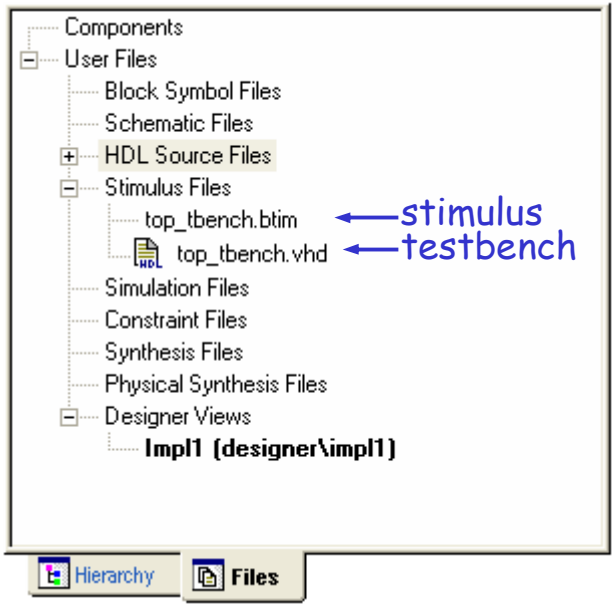
- Select Export from WaveFormer Lite Menu
  - WaveFormer Lite Has Many Export Options
  - Recommendations
    - VHDL Testbench - Select “VHDL with Top Level Testbench”
    - Verilog Testbench - Select “Verilog with Top Level Testbench”



Waveform and Testbench



Stimulus and testbench appear on File Manager tab in Libero



Design Verification with ModelSim AE

ModelSim AE, PE, LE, & SE

	Actel Edition	PE	LE	SE
Operating System	Windows	Windows	Linux	All
Advanced Optimizations	N/A	N/A	N/A	Included
Performance Analyzer	N/A	N/A	N/A	Included
C Debugger	N/A	N/A	N/A	Included
Dataflow Window	N/A	Optional	Included	Included
Waveform Comparison	N/A	Optional	Included	Included
SWIFT	N/A	Optional	Optional	Included
Code Coverage	N/A	Optional	Optional	Included
Stand-Alone Viewer	N/A	Optional	Optional	Included
Assertions (PSL)	N/A	Optional	Optional	Optional
SystemC	N/A	Optional	Optional	Optional

ModelSim AE

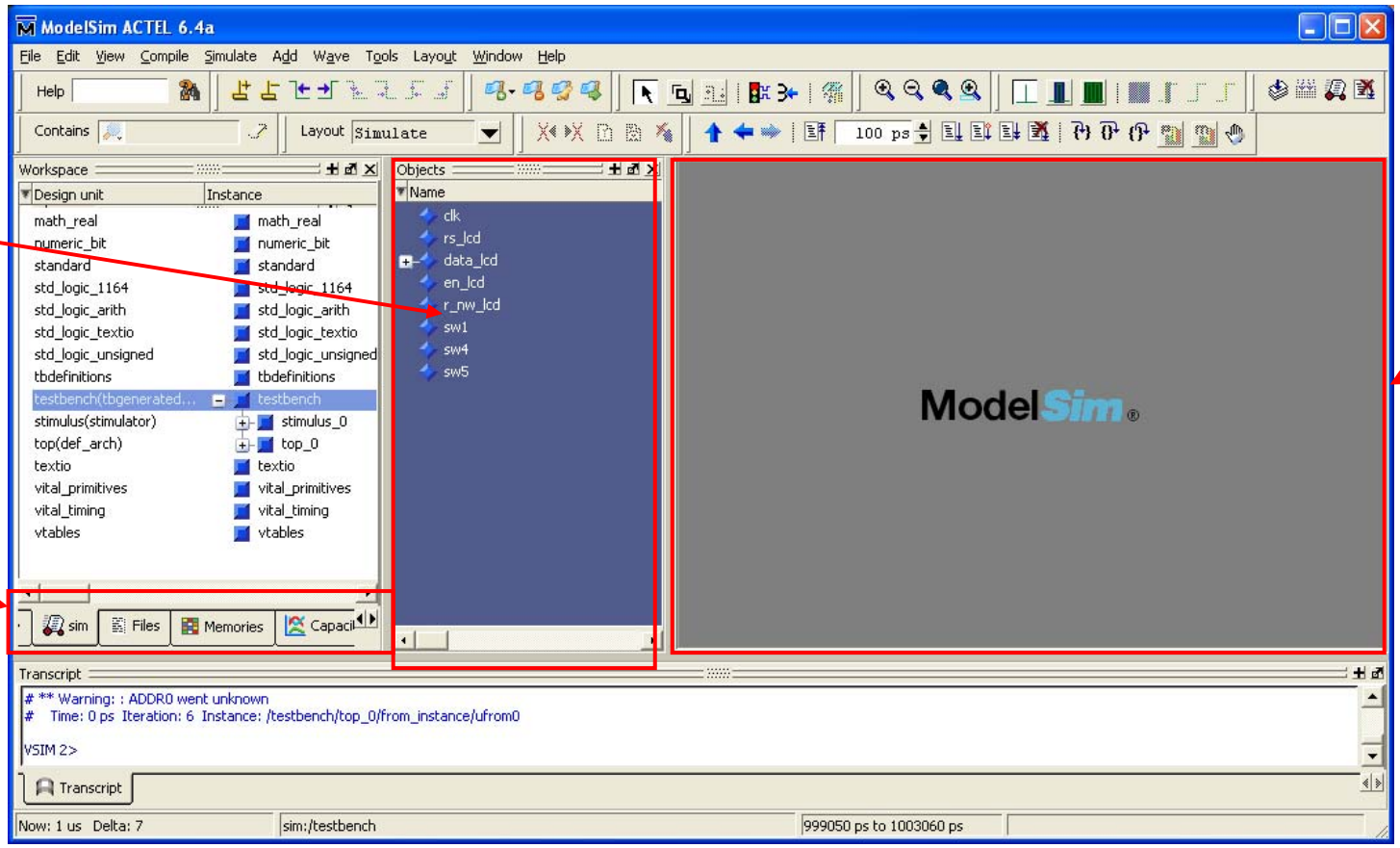
- Same Functionality as ModelSim PE
  - Windows XP and Windows Vista Business
    - Node Locked
- Reduced Performance
  - ~ 30% of ModelSim PE Performance
- Single Language
  - No co-simulation (VHDL and Verilog) capability
- Structural Simulation Limited to Actel's Gate-level Libraries
  - Pre-compiled libraries required
- Supported Through the Actel

ModelSim Default Window Layout

Dockable Window

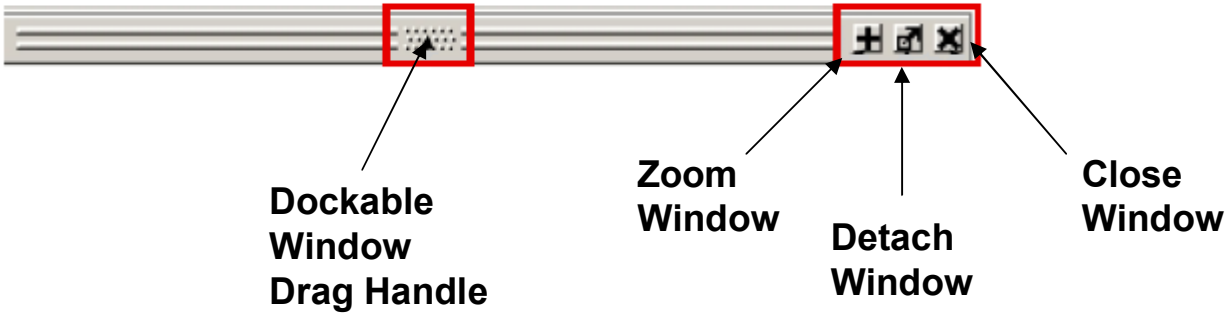
Workspace Tabbed Windows

MDI Pane



ModelSim Dockable Windows

- Windows Within The Main Window Are Dockable
  - Move The “Drag Handle” To The Desired Location



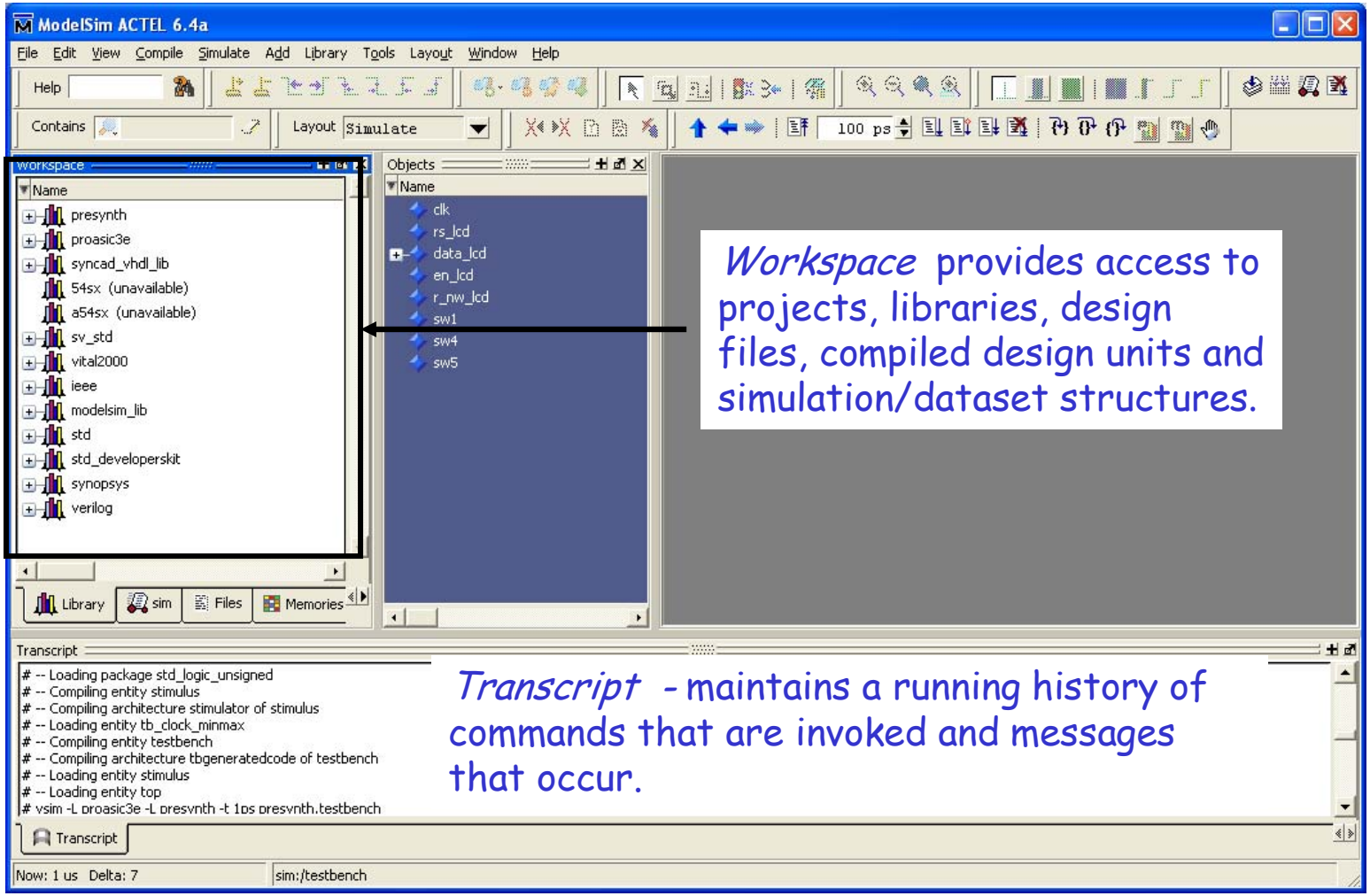
- Additional Window Features
  - Drag & Drop
    - HDL Items Can Be Dragged from Dataflow, List, Signals, Source, Structure, Variables, and Wave Windows …
    - … And Dropped into either List or Wave Window
  - Automatic Window Updating

Common Window Features

- Drag & Drop
  - The HDL items can be dragged from the Dataflow, List, Signals, Source, Structure, Variables, and Wave windows and dropped into the List or Wave window
- Automatic window updating
  - Dataflow, Process and Objects windows
- Find Feature
- Sorting HDL items
  - Sort ascending, descending or appearance order
  - Process, Signals, Source, Structure, Variables and Wave windows
- Combining Items

ModelSim Main Window

- Main Window Appears when ModelSim is Launched



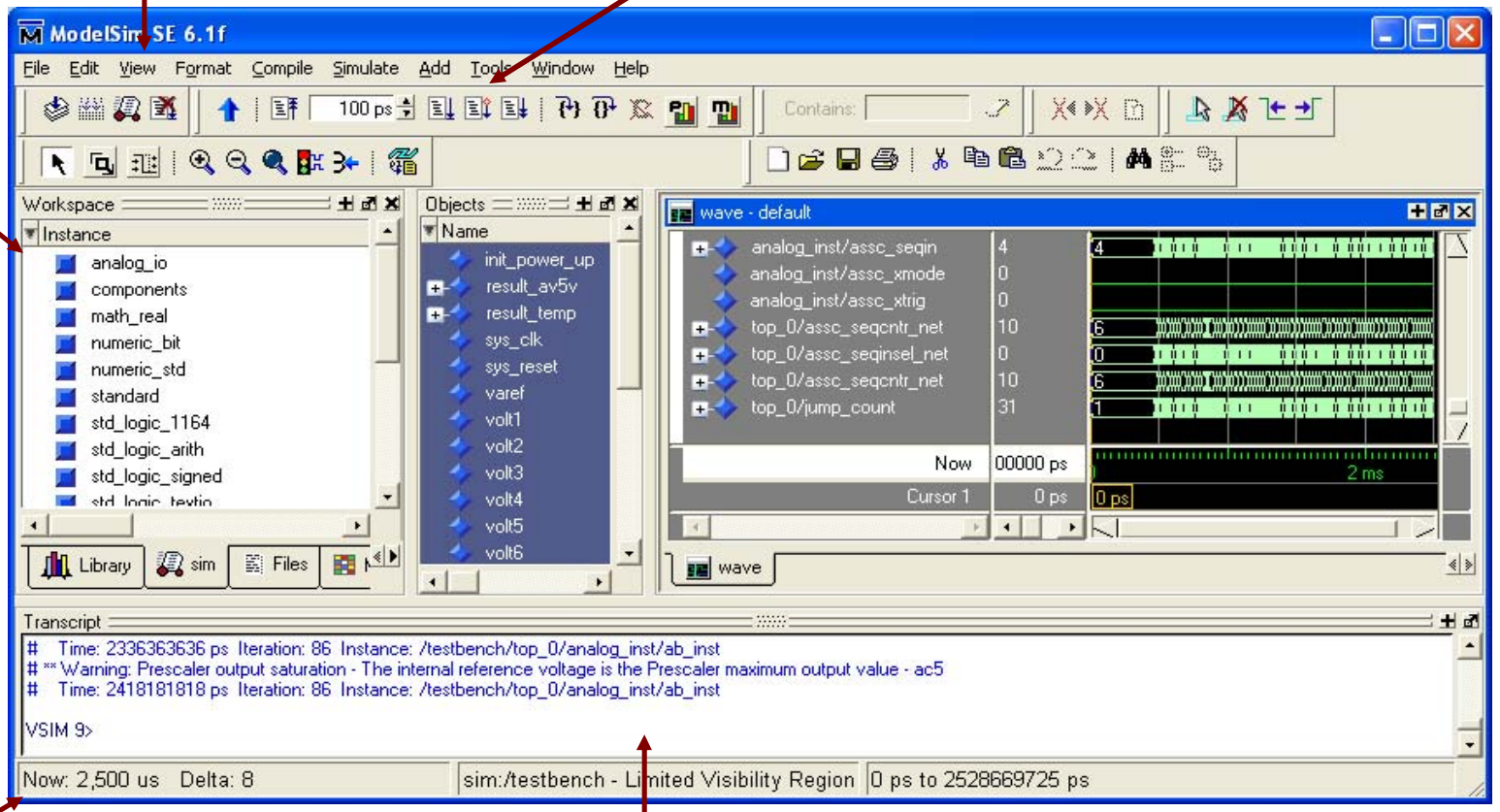
ModelSim Main Window

Active Project with Design Loaded

Menu Bar

Tool Bar (***Break, Run, Cont, Step,*** and ***Step Over***)

Design Hierarchy



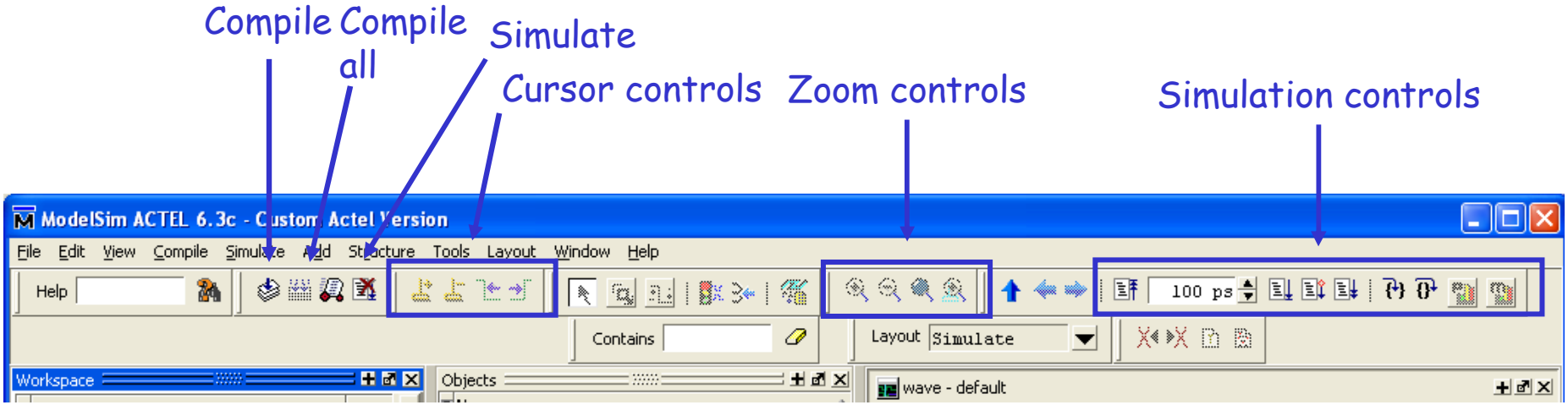
Status Bar (***current time, delta time step, environment)***

TCL Interpreter

ModelSim> prompt before design is loaded.

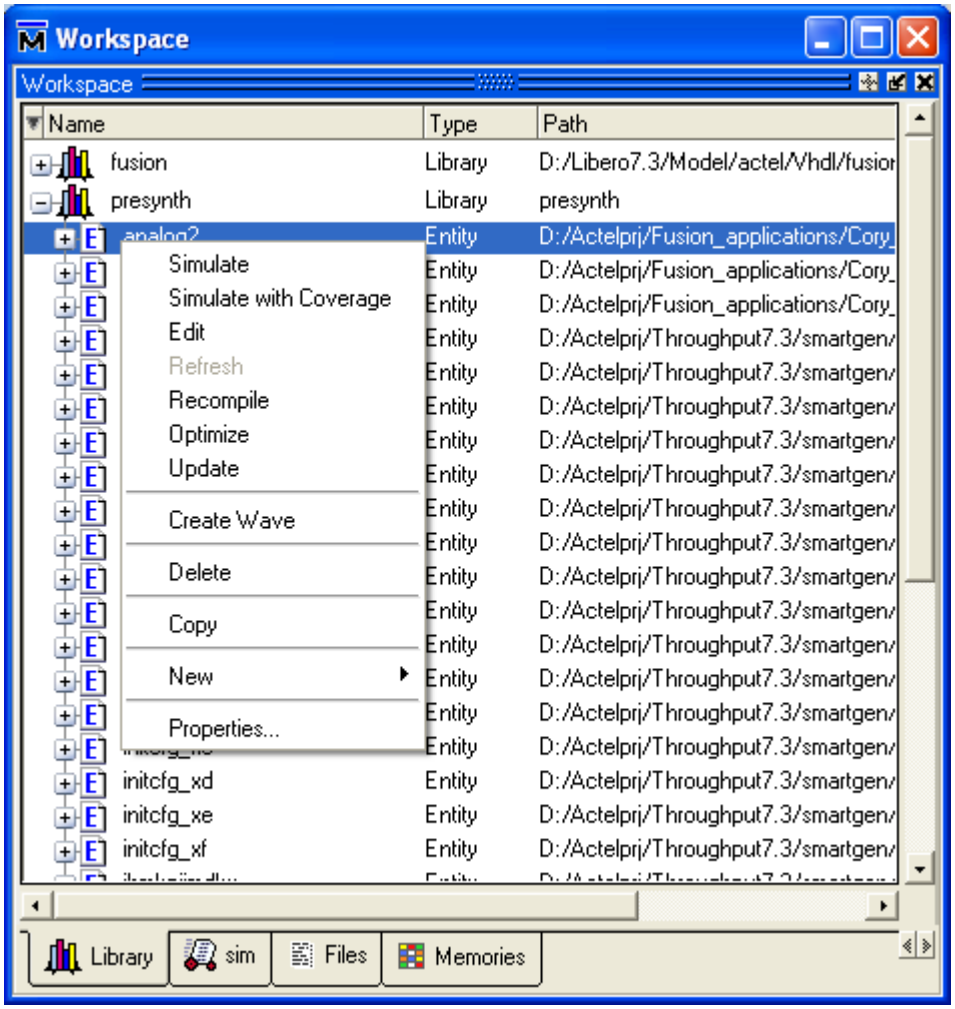
VSIM> prompt is displayed after design is loaded

ModelSim Toolbar



Workspace Window

- Workspace Provides Access to:
  - Libraries
  - Compiled Design Units
  - Design Source Files
  - Memory Modules



Source Window

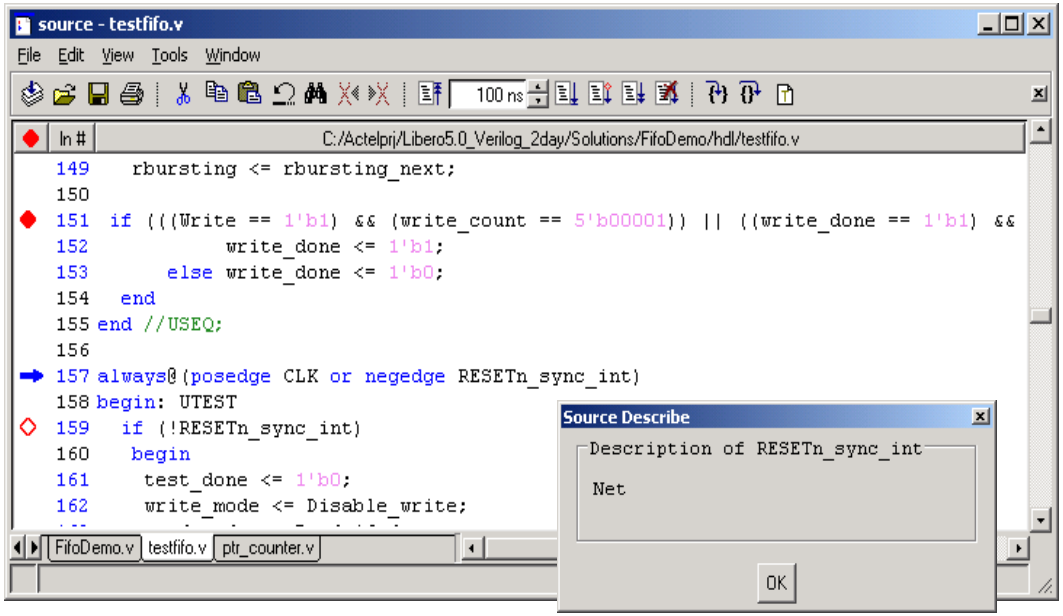
- View and Edit HDL source code
  - Color-coded comments, keywords, strings, numbers, executable lines, identifiers, system tasks, text
  - Save, compile and restart simulation from Source window
- Features
  - Describe and Examine selected items
  - Find and Replace items
  - Set Breakpoints

•Blue line numbers – executable lines where a breakpoint can be set

•Red diamonds – enabled breakpoints

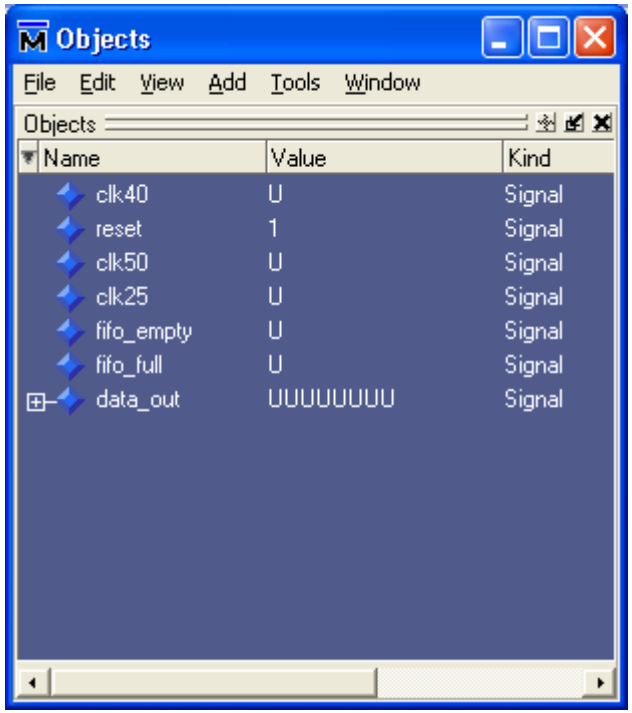
•Hollow diamonds – disabled breakpoints

•Blue arrow – denotes a process selected in the Process window



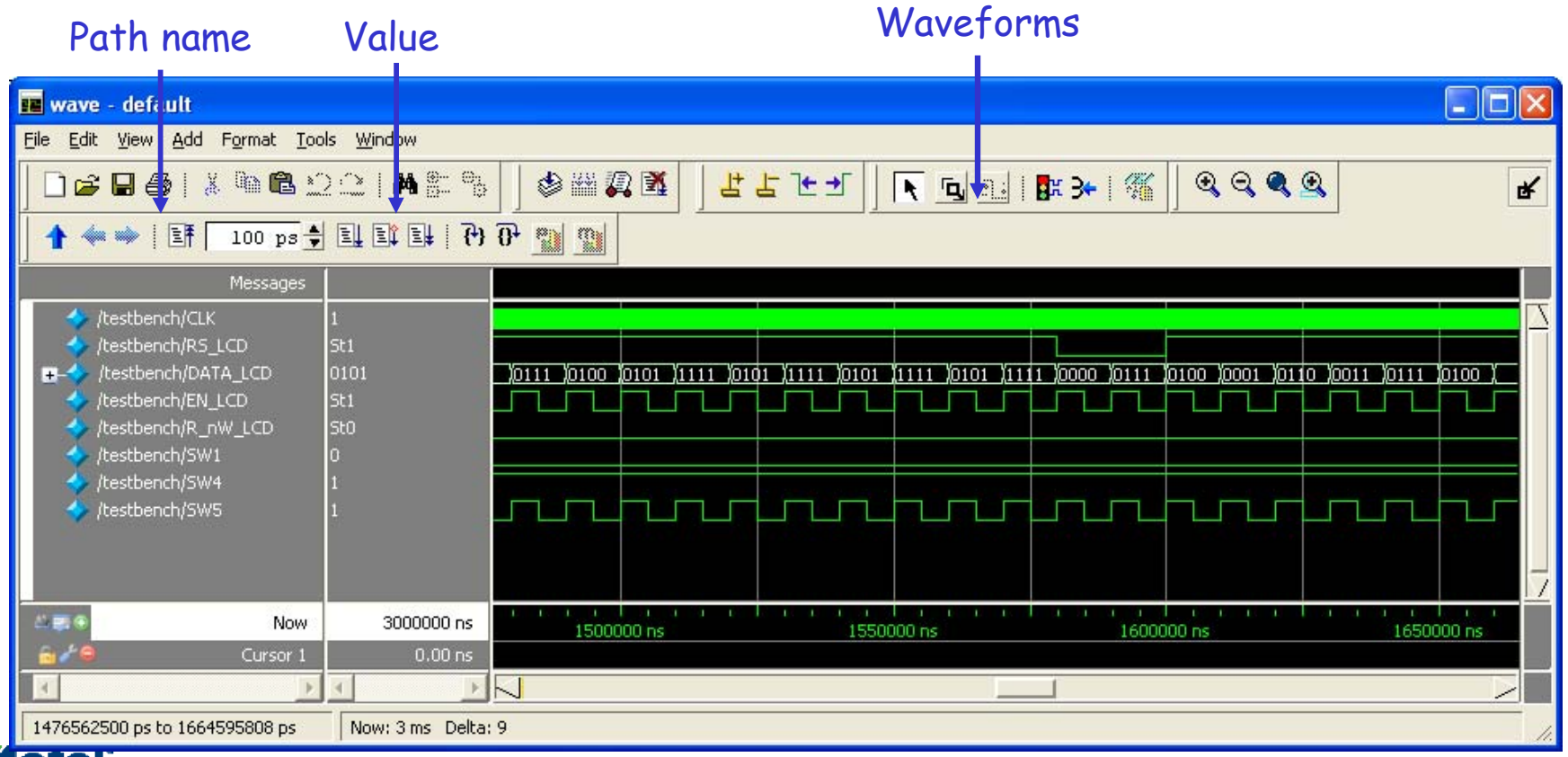
Objects Window

- Shows Names and Values of HDL Items in Current Region Selected in the Workspace
  - Objects include signals, nets, registers, constants and variables not declared in a process, generics and parameters
- Items Can Be Sorted in Ascending, Descending or Declaration Order
- Hierarchy (+) Expandable
  - VHDL Items - Signals
  - Verilog Items - Nets, Register Variables
  - Named Events
- "Drag & Drop"
  - Wave & List windows
    - Force Apply Stimulus
    - Filter Signal Types (input,output
    - etc)
    - Find HDL Items



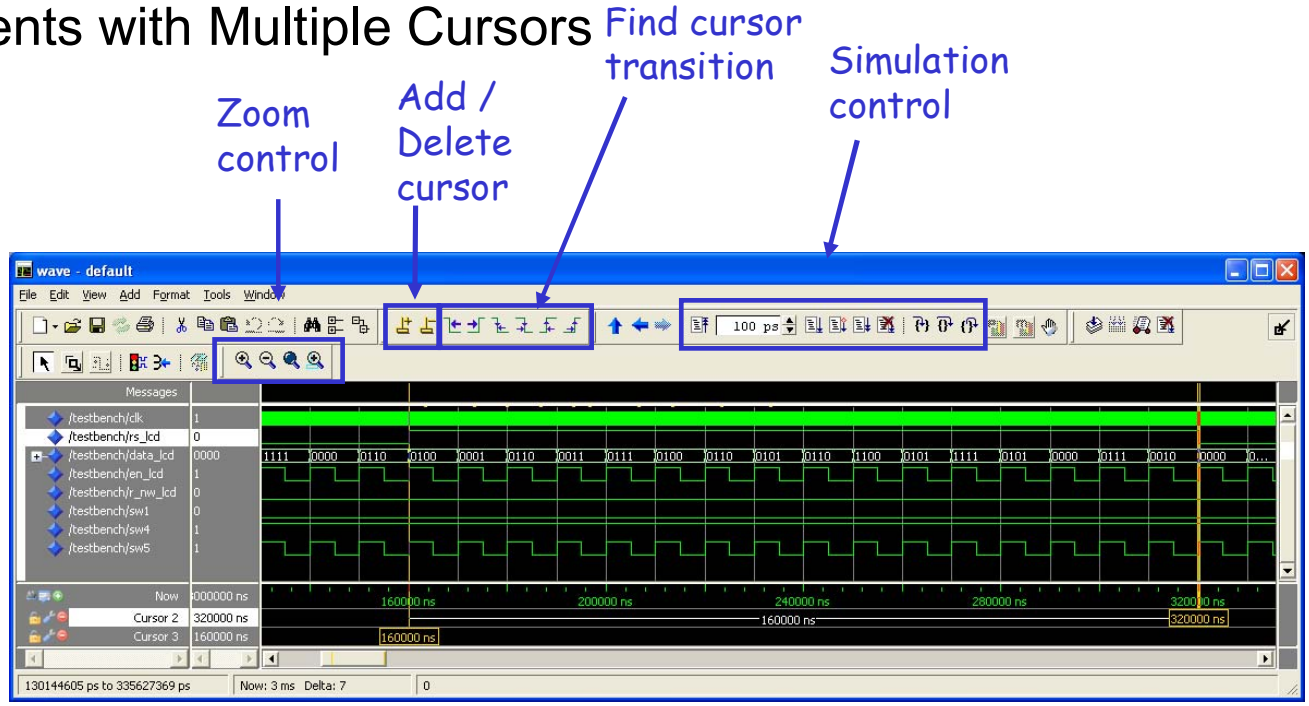
Wave Window

- Displays Waveforms of Simulation Results
  - VHDL – signals, processes and shared variables
  - Verilog – nets, registers, variables and named events
  - Virtuals – virtual signals, busses and functions



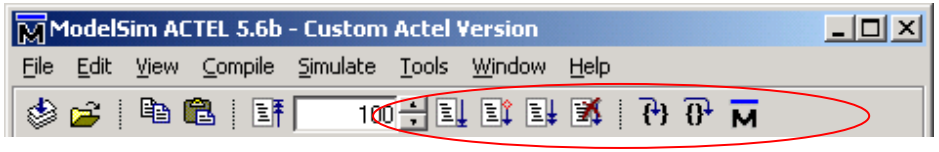
Wave Window Features

- Drag and Drop from from the List, Process, Signals, Source, Structure, or Variables window
- Timing Measurements with Multiple Cursors
- Display Formatting
  - Dividers
  - Window Panes
- Signal Formatting
  - Radix
  - Combining
- Simulation Control
  - Breakpoints
  - Restart, Run
  - Run length
- Sort
  - Ascending or descending
- Zoom
- Find

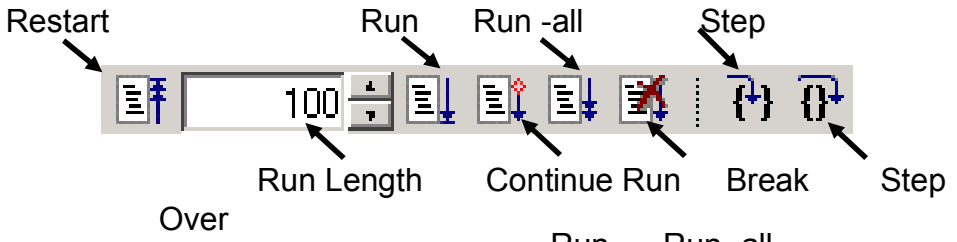


Advancing Simulation Time
Main and Wave Windows

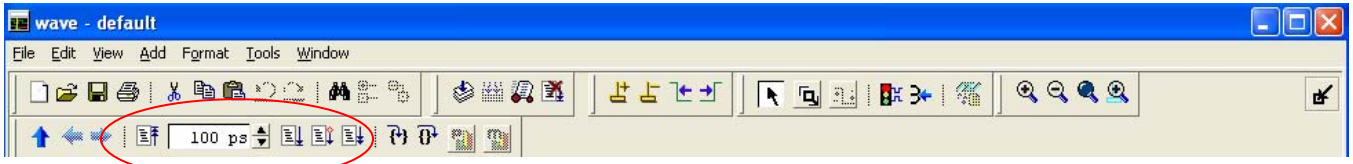
- **From Main Window**
  - **At the VSIM prompt:**
    - `VSIM 12> run 100 ms`
  - **In The Main Window Tool Bar:**

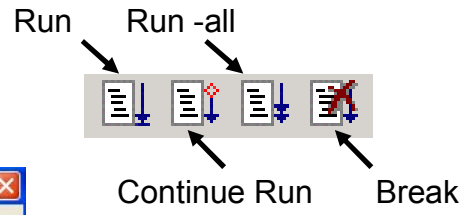


- **•Run** - run the current simulation for the specified run length
- **•Continue Run** - continue the current simulation run until the end of the specified run length or until it hits a breakpoint or specified break event
- •Run -All - run the current simulation forever, or until it hits a breakpoint or specified break event



- **From Wave Window**
  - **In The Wave Window Tool Bar:**





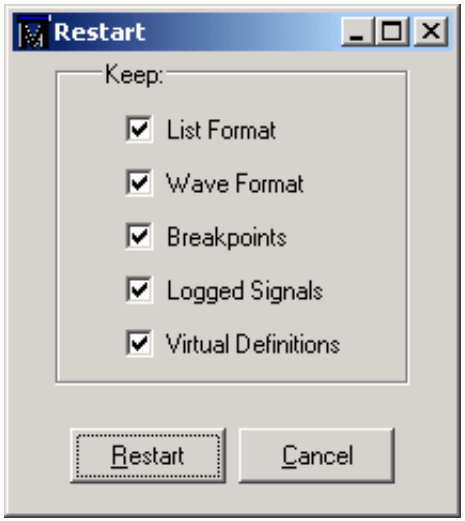
Restarting Simulation

- **Restart to Zero**
  - At the VSIM prompt force a re-start

    ```
    VSIM 12> restart –f
    ```

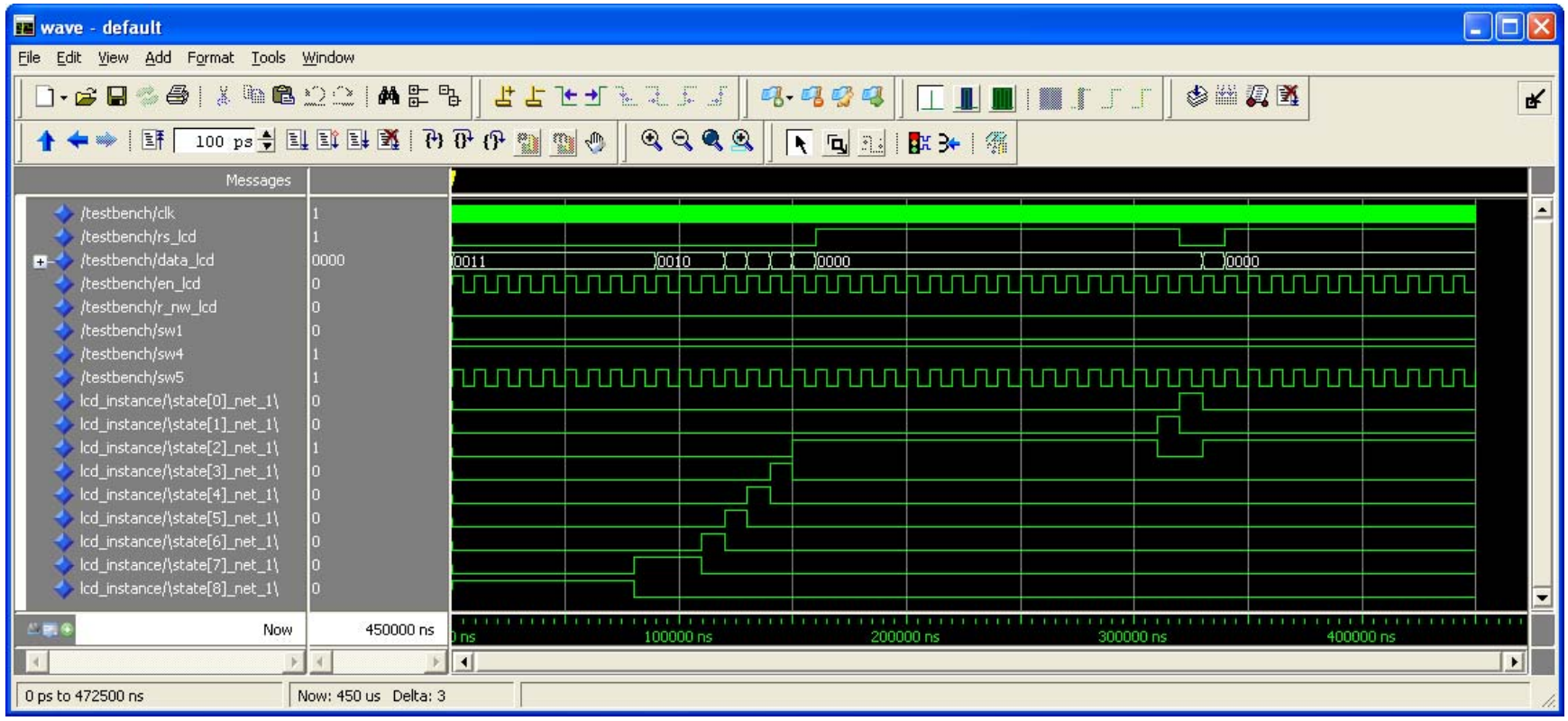
  - In The Main Window Run > Restart or Restart Button
    - Will display restart dialog.

- **Restart Dialog**
  - Keep current
    - Listed Signals
    - Waved Signals
    - Breakpoints
    - Logged Signals
    - Virtual Signals



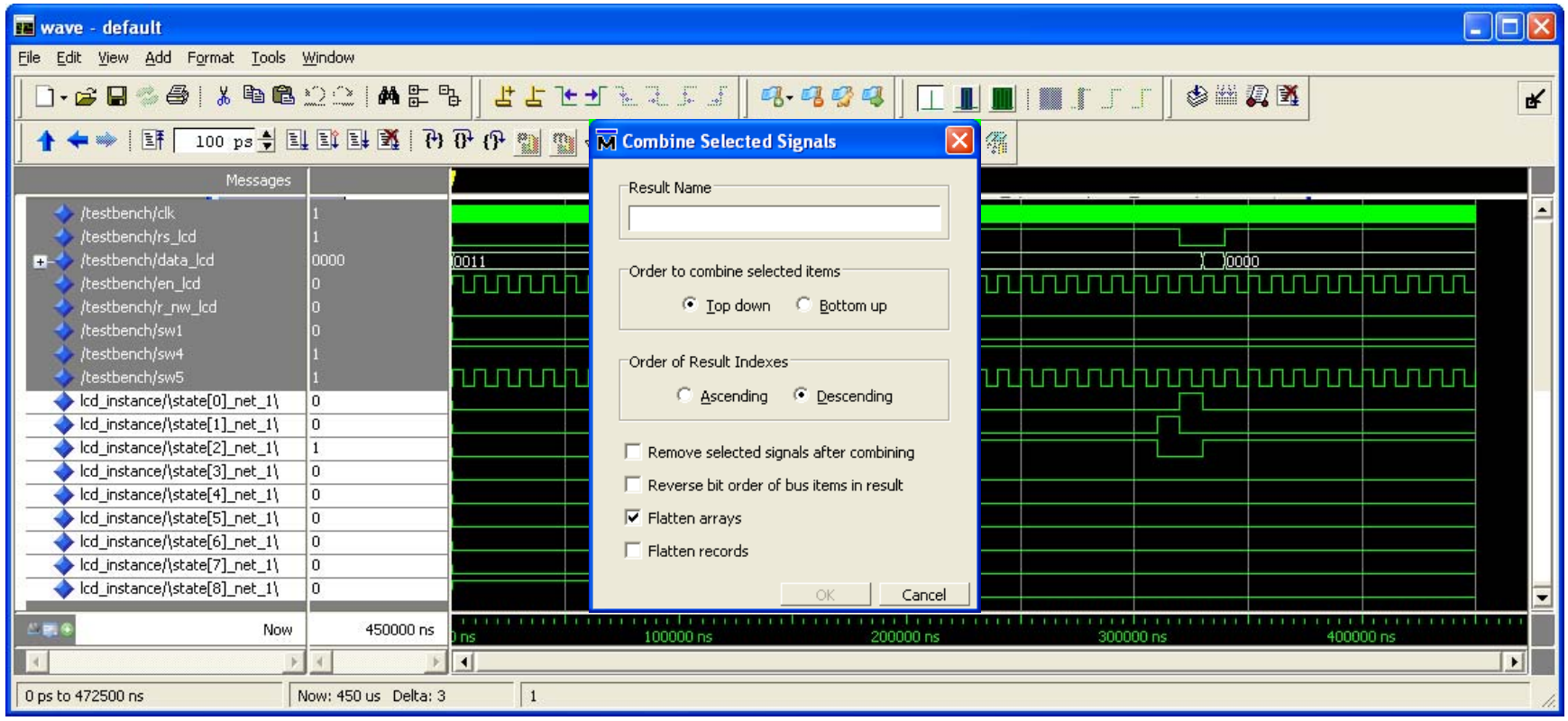
Wave Window
Virtual Busses

- Scalar Signals can be Combined into Vectors



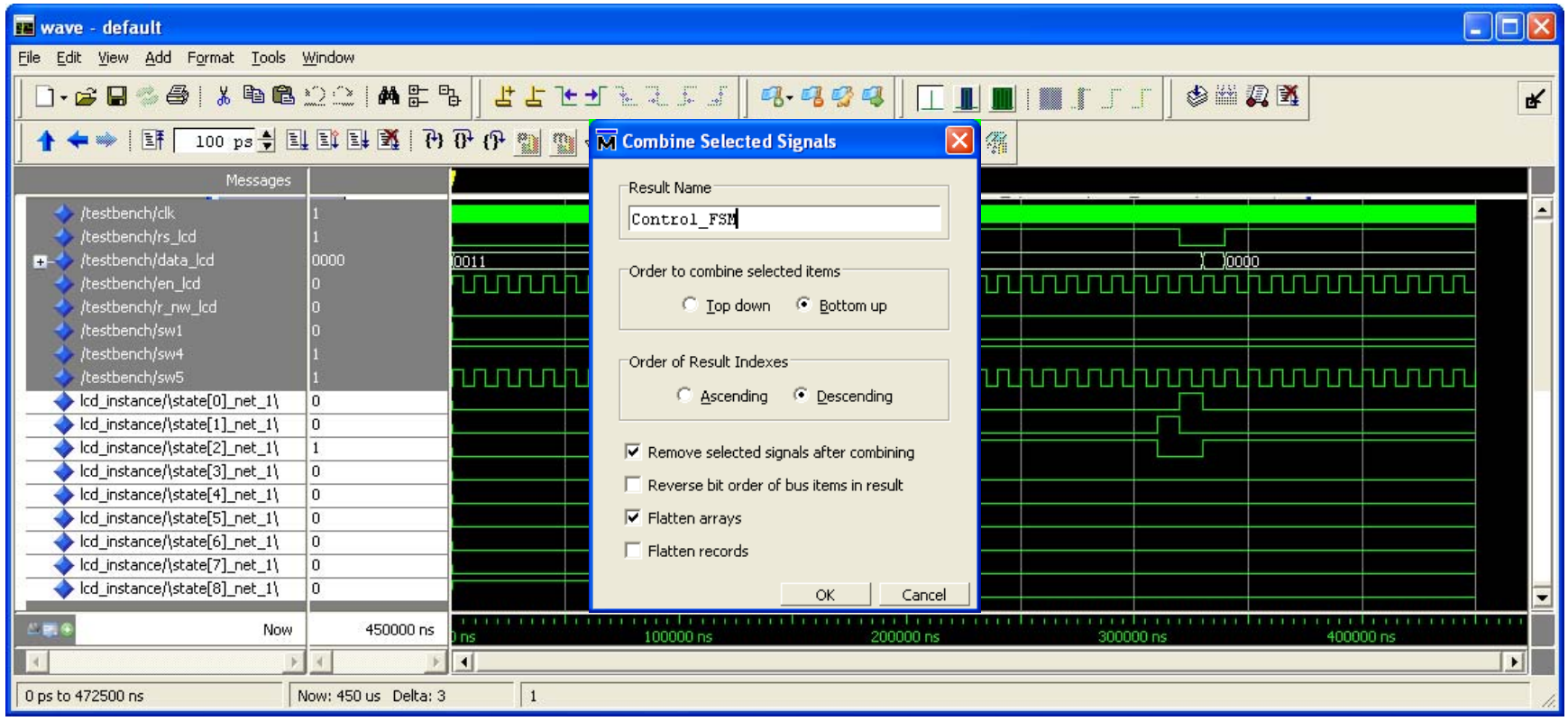
Wave Window
Virtual Busses

- Scalar Signals can be Combined into Vectors



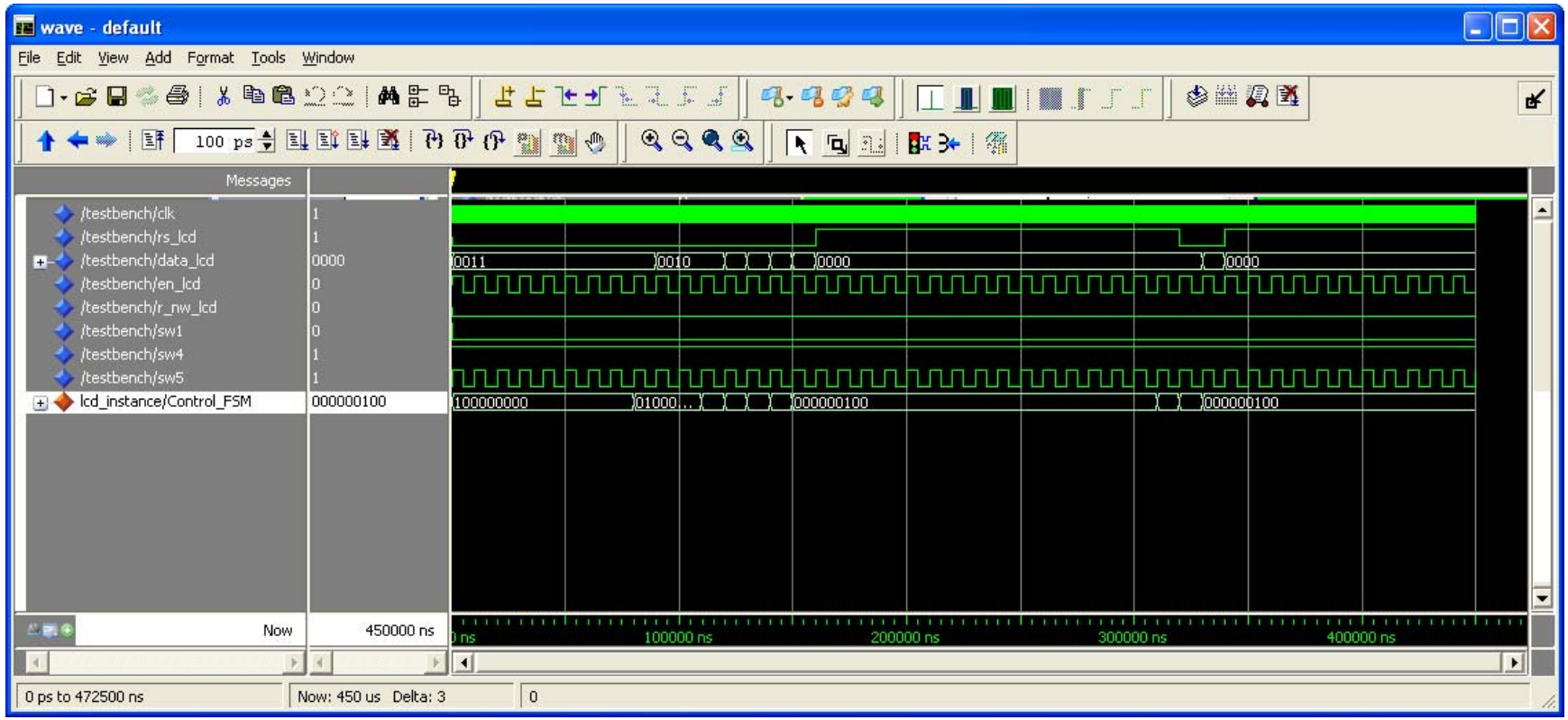
Wave Window
Virtual Busses

- Scalar Signals can be Combined into Vectors



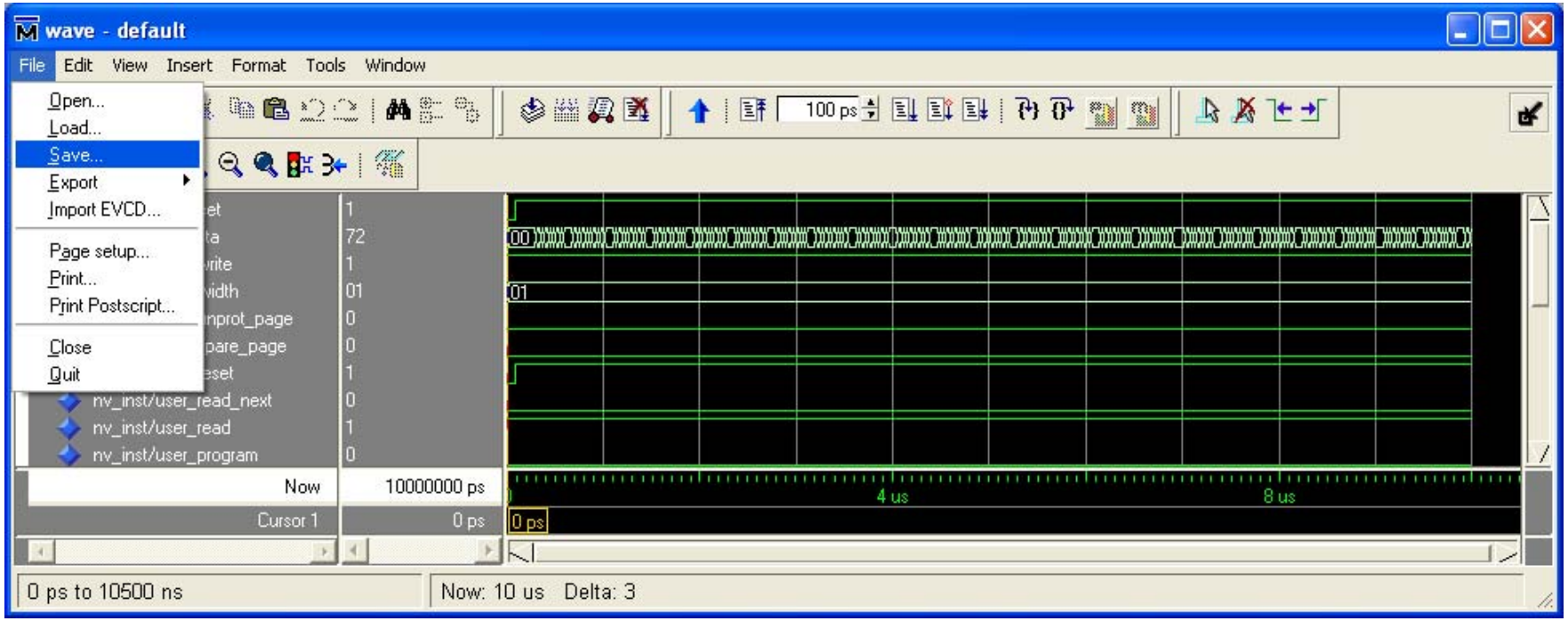
Wave Window
Virtual Busses

- Scalar Signals can be Combined into Vectors



Wave Window
Saving Wave Data

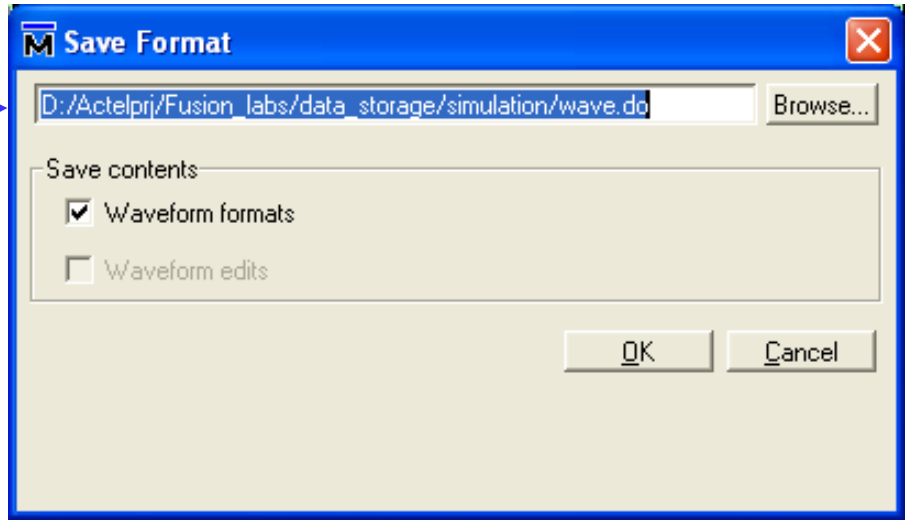
- Signals Added to Wave Window can be Saved
  - File > Save from Wave window
- Select Saved Wave Format in Libero Simulation Options for Future Simulation Runs



Wave Window
Saving Wave Data (cont.)

- Enter File Name in Save Format Dialog box
  - Enter name on Libero simulation tab to include signals in future simulation runs

Enter file name →



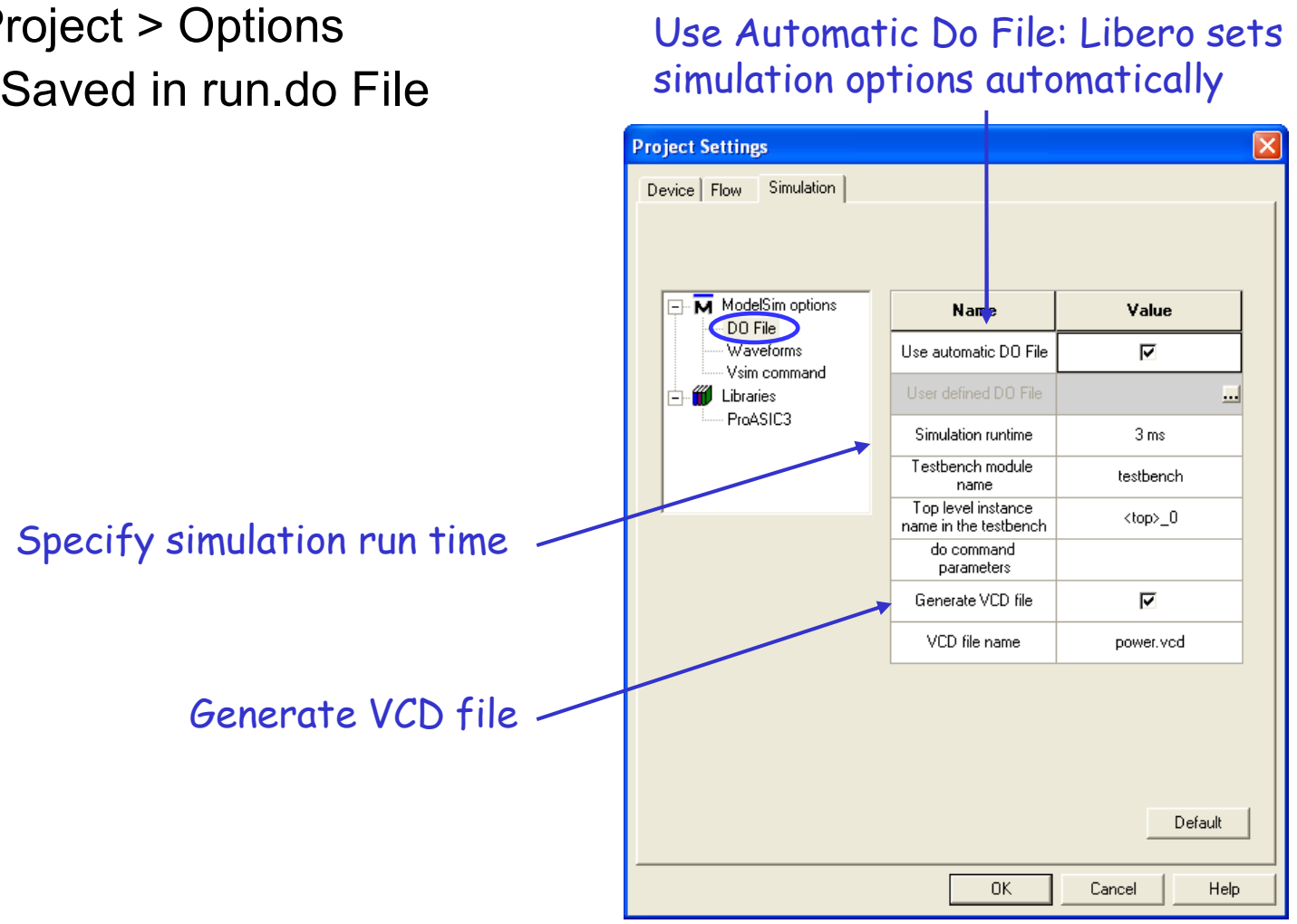
ModelSim Macro Files

- ModelSim Commands can be Saved in a Macro File
  - The do command executes commands.
  - A macro file can have any name and extension.
- Command line Syntax:
  - do<filename> [<parameter_value>]
  - Example: do run.do
- Macros can also be run from the Main Window Toolbar
  - Tools > Execute Macro

```
vlib presynth
vmap presynth ./presynth
vcom -93 -work presynth D:/Actelprj/count32/hdl/count32.vhd
vcom -93 -work presynth D:/Actelprj/count32/stimulus/count32.vhd
vsim presynth.testbench
add wave /testbench/*
run 1000ns
```

Libero Simulation Options
ModelSim Do File Options

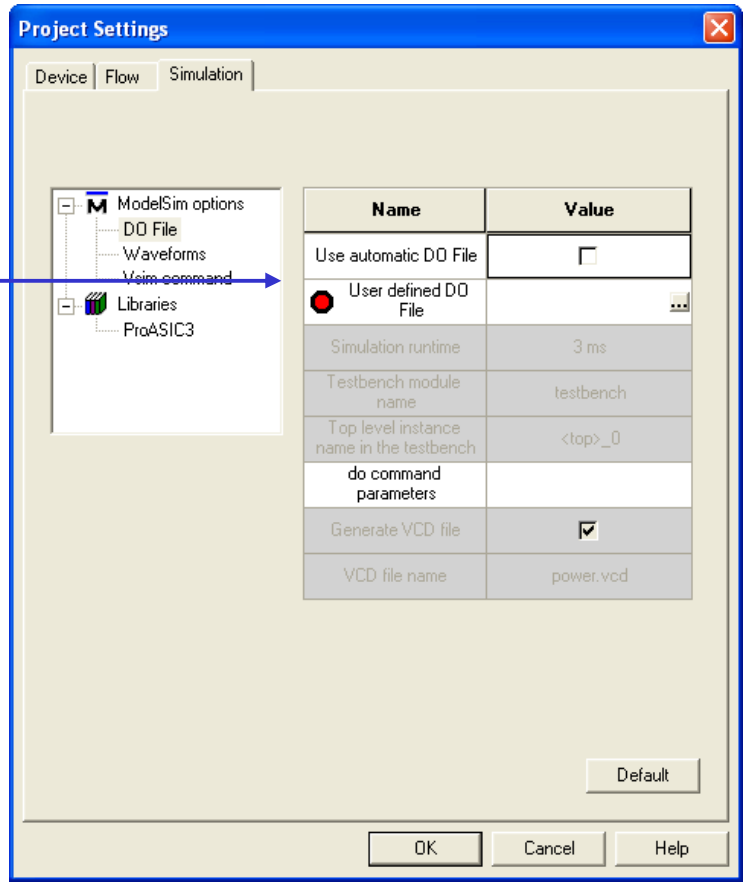
- Simulation Options Can Be Set from Simulation Tab
  - Select Project > Options
  - Results Saved in run.do File



Libero Simulation Options

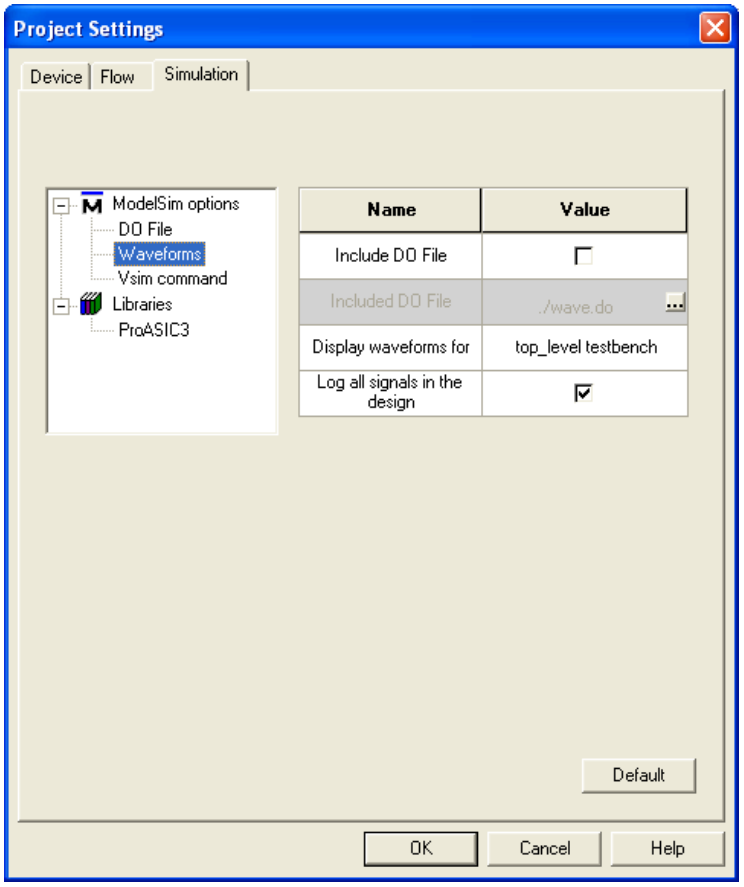
User run.do File

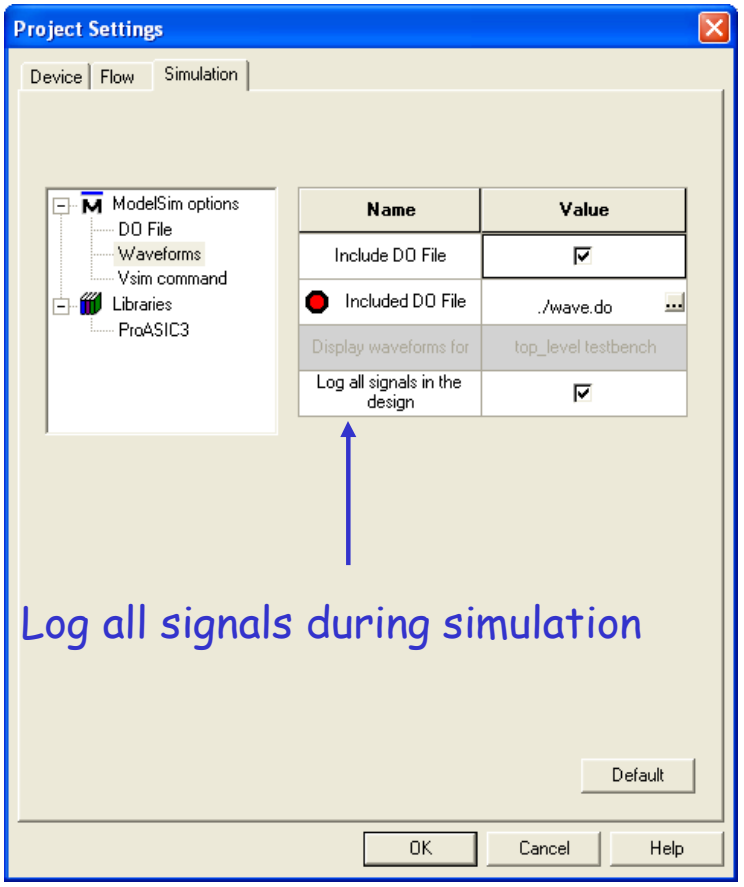
Include user-defined script. Browse to script file location



Libero Simulation Options
Waveform Options

- Specify Signals to be Displayed in ModelSim Wave Window





Libero Simulation Options
ModelSim VSIM Commands

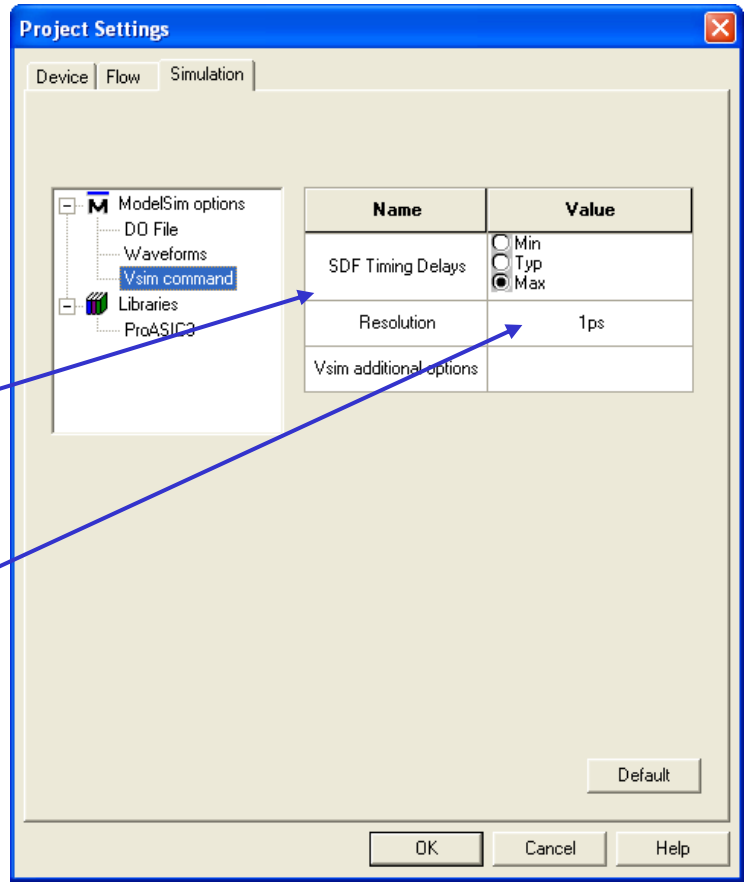
- Specify Simulator Resolution and Post-Layout Simulation Operating Conditions

Select min, typ, max simulation conditions for post-layout simulation

Default resolution based on family choice

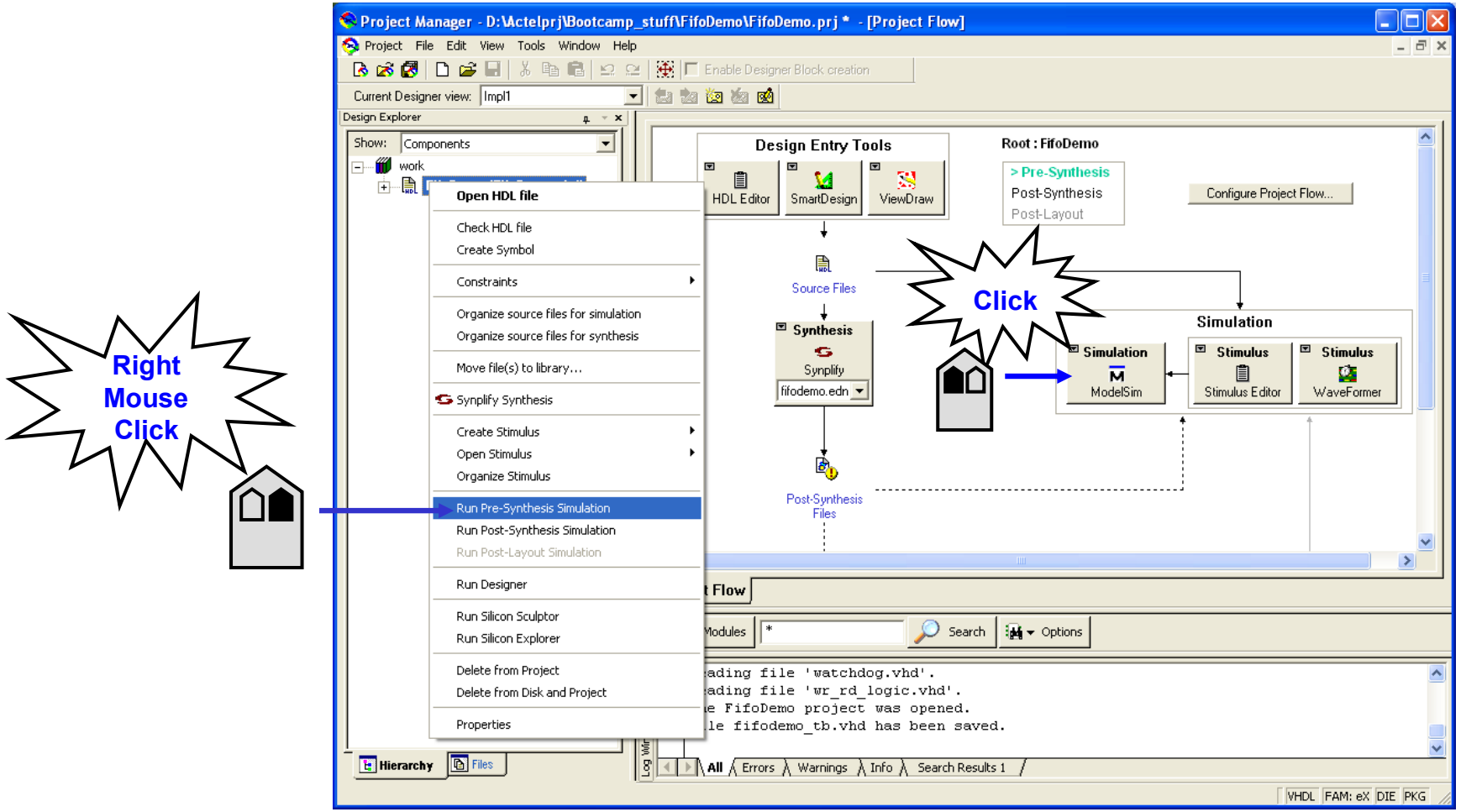
1ps for 500K, APA, 54SXA, AX, ProASIC3/E, Fusion, IGLOO/e

1ns for all other families



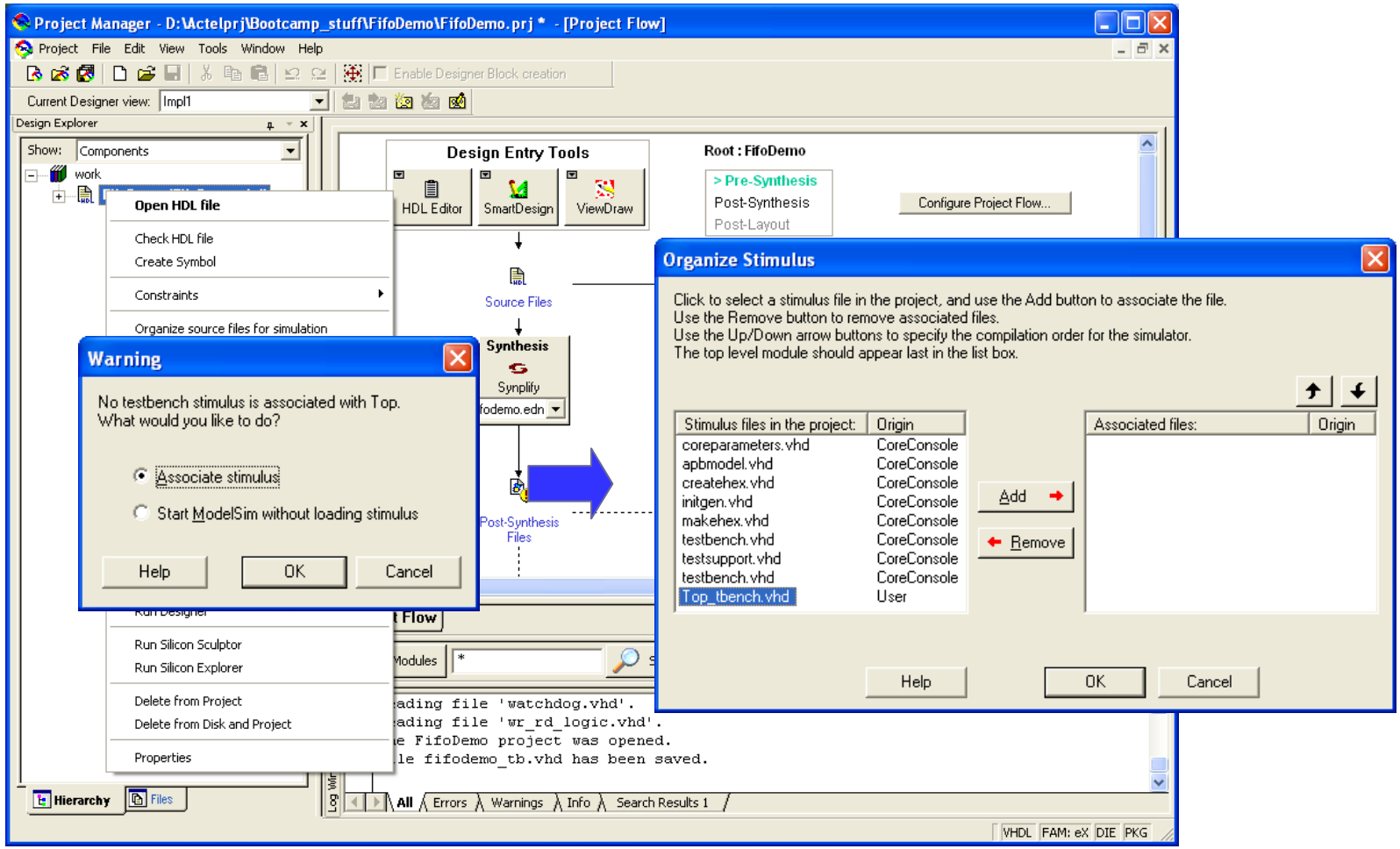
Invoking ModelSim
Pre- or Post-Synthesis

- Click “Simulation” Project Flow Window or …



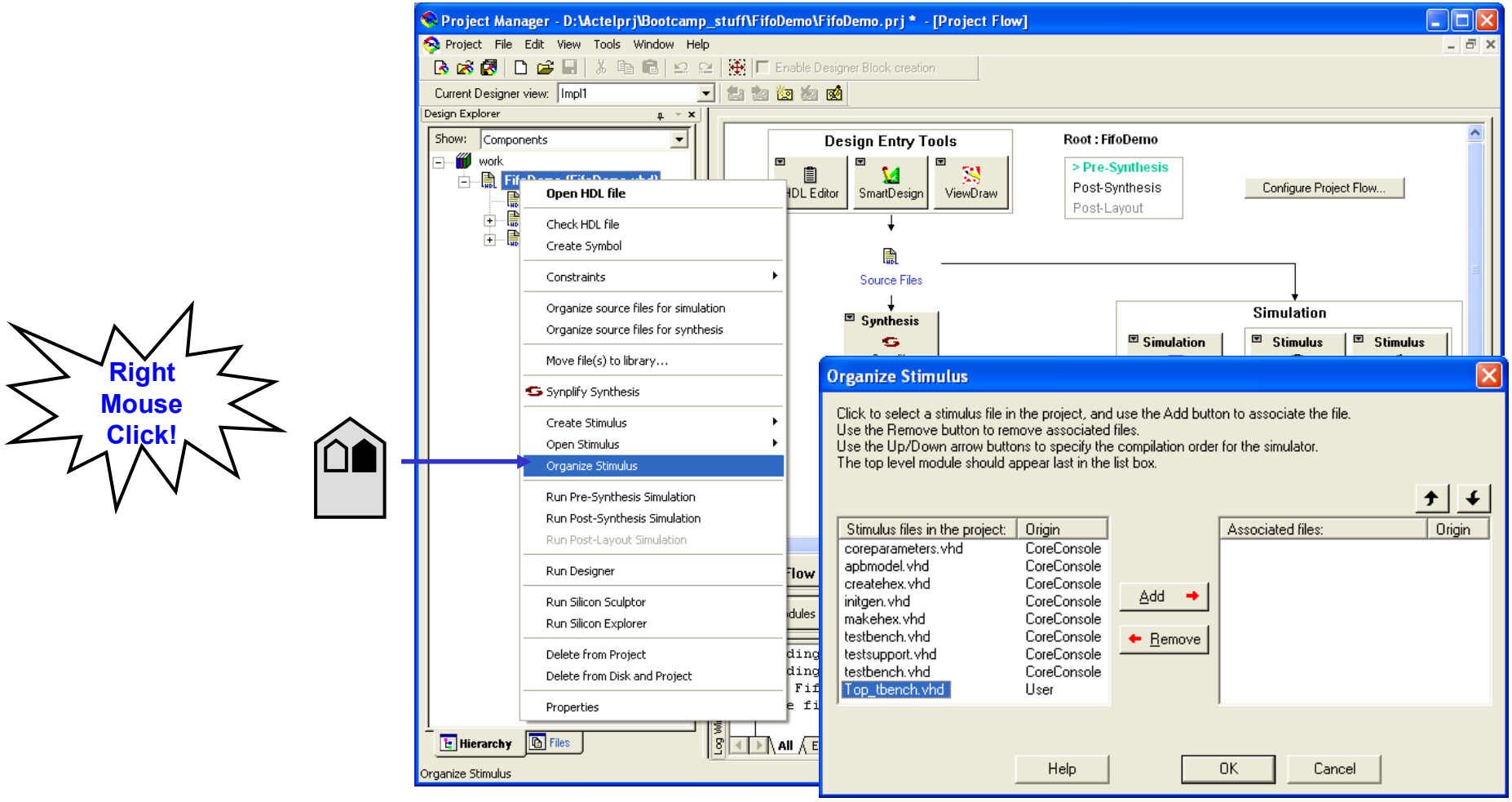
Associating Stimulus

- Libero Prompts for Testbench if none is Associated
  - Select testbench from Organize Stimulus dialog



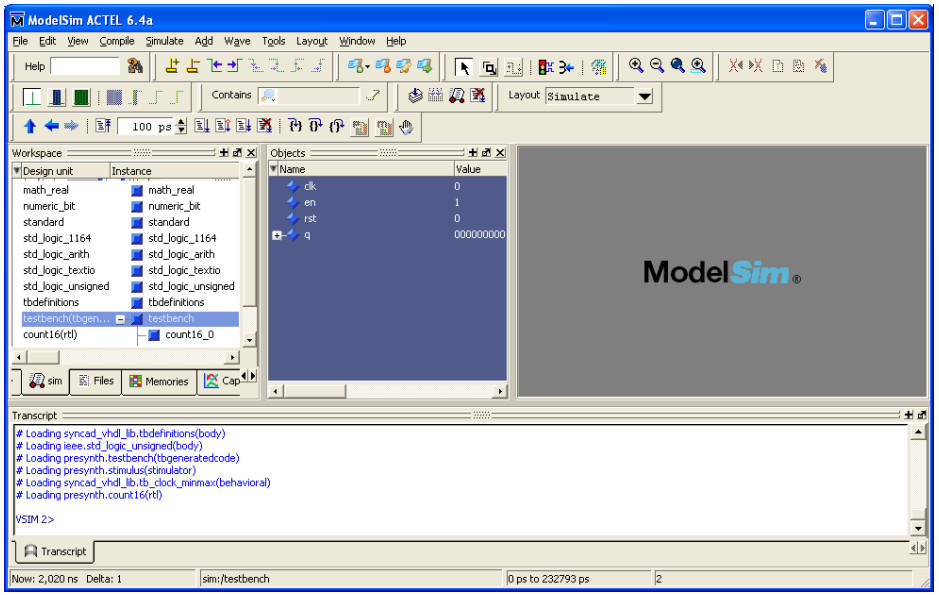
Associating Stimulus Manually

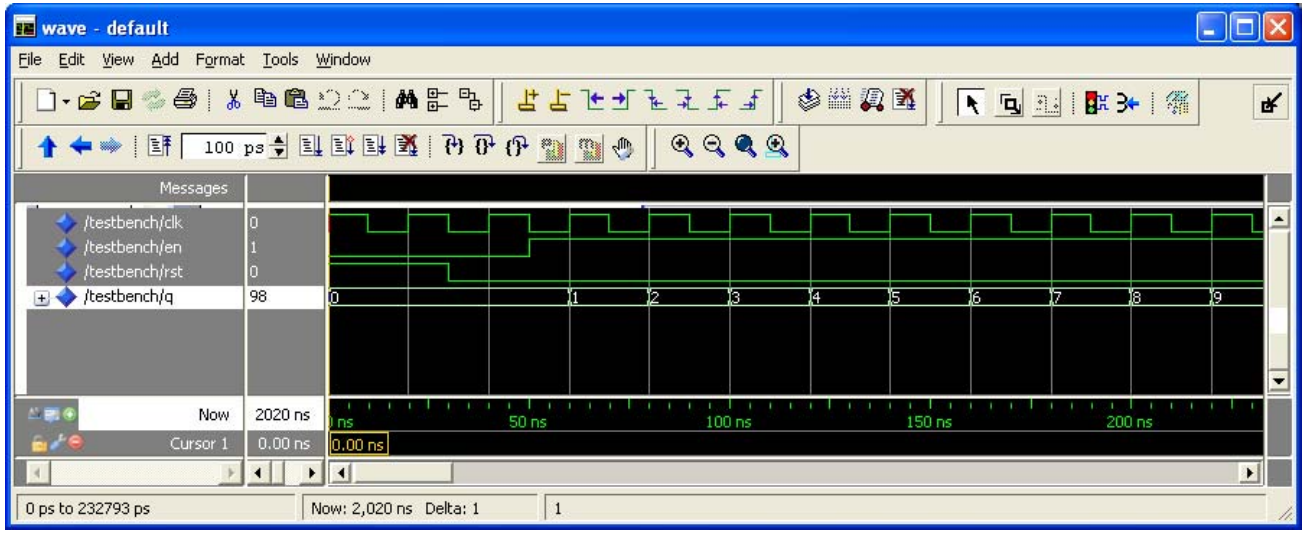
- Select a Different Testbench for Simulation



Pre-Synthesis Simulation

- ModelSim Automatically Compiles Design and Runs Simulation
  - Default run time is 1 µS
  - (External) Signals from Testbench Automatically Added to Wave Window
  - Additional (Internal) Signals Can Be Added by User





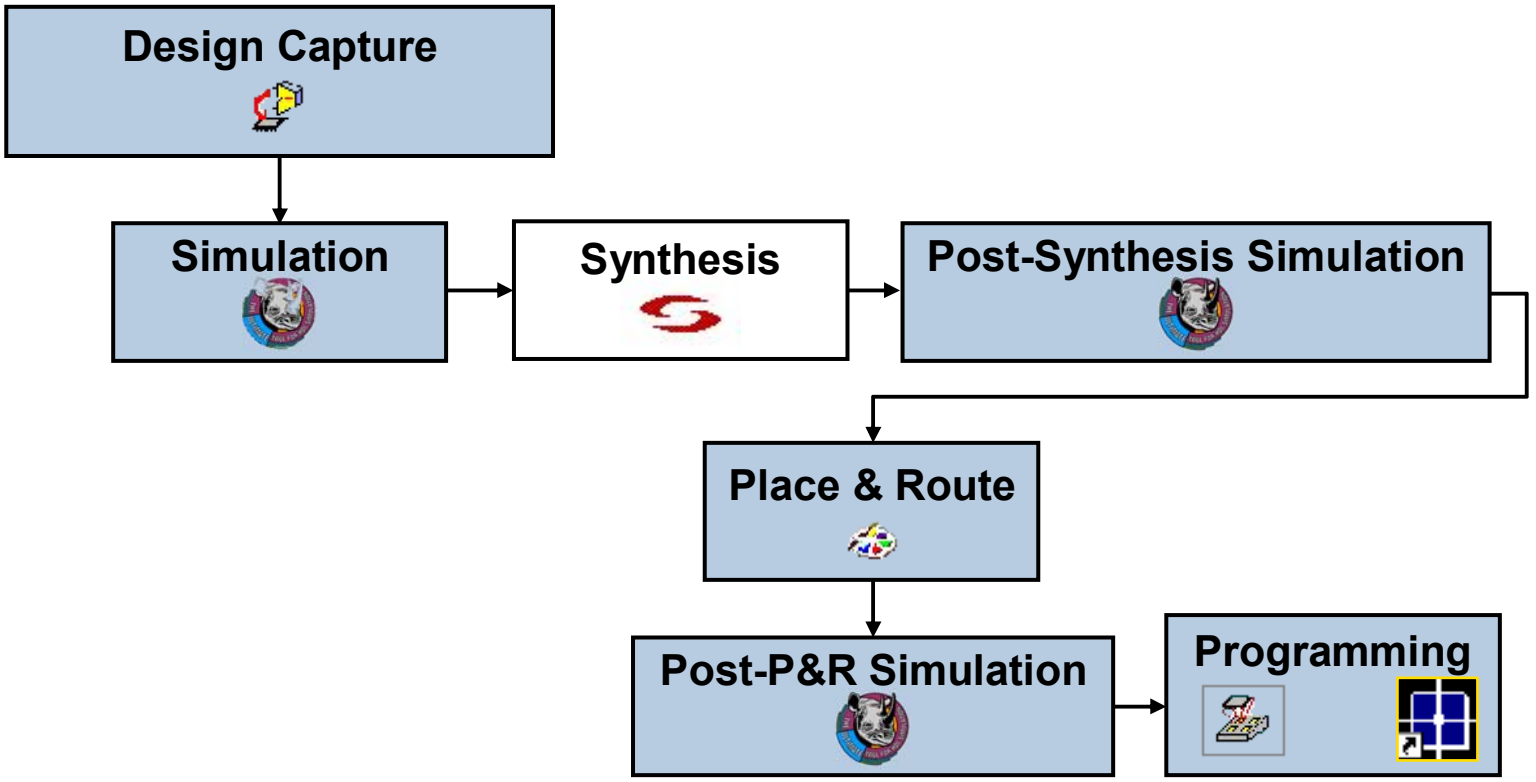
Simulating Designs
Summary

- Capture Design
  - Generate RTL Netlist (VHDL or Verilog)
  - OR
  - Create Schematic
    - May Include RTL blocks
    - Structural VHDL or Verilog Netlist Automatically Created before Simulation
- Create Testbench
  - Use WaveFormer Lite, or Text Editor
- Associate Stimulus
- Run Pre-Synthesis Simulation

Hands-on Lab 3

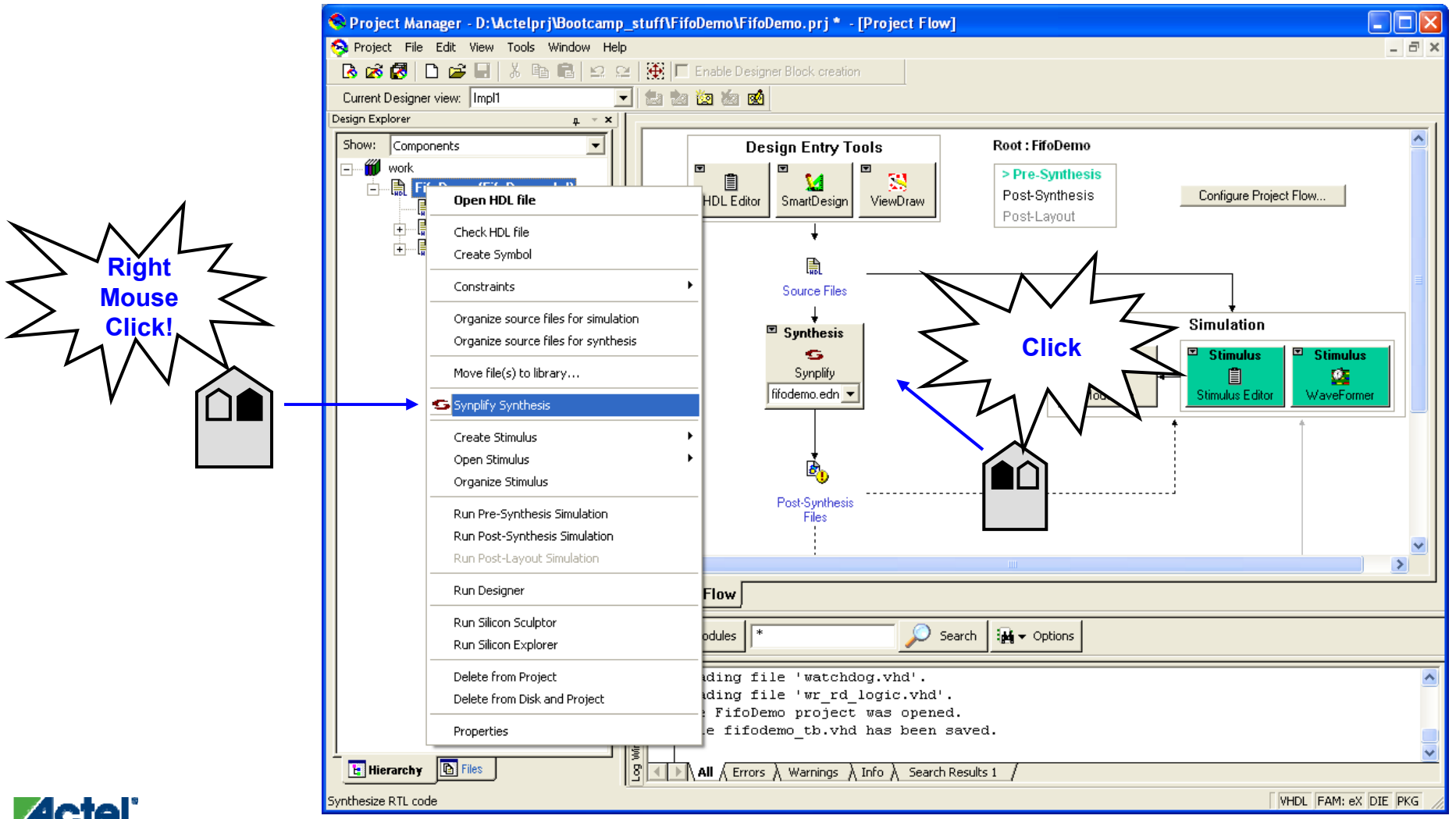
- Complete Lab 3 in the Lab Guide
  - Perform Pre-synthesis Design Simulation
  - Save Your Work

Synthesis


Design Capture
Simulation
Synthesis
Post-Synthesis Simulation
Place & Route
Post-P&R Simulation
Programming

Invoking Synplicity

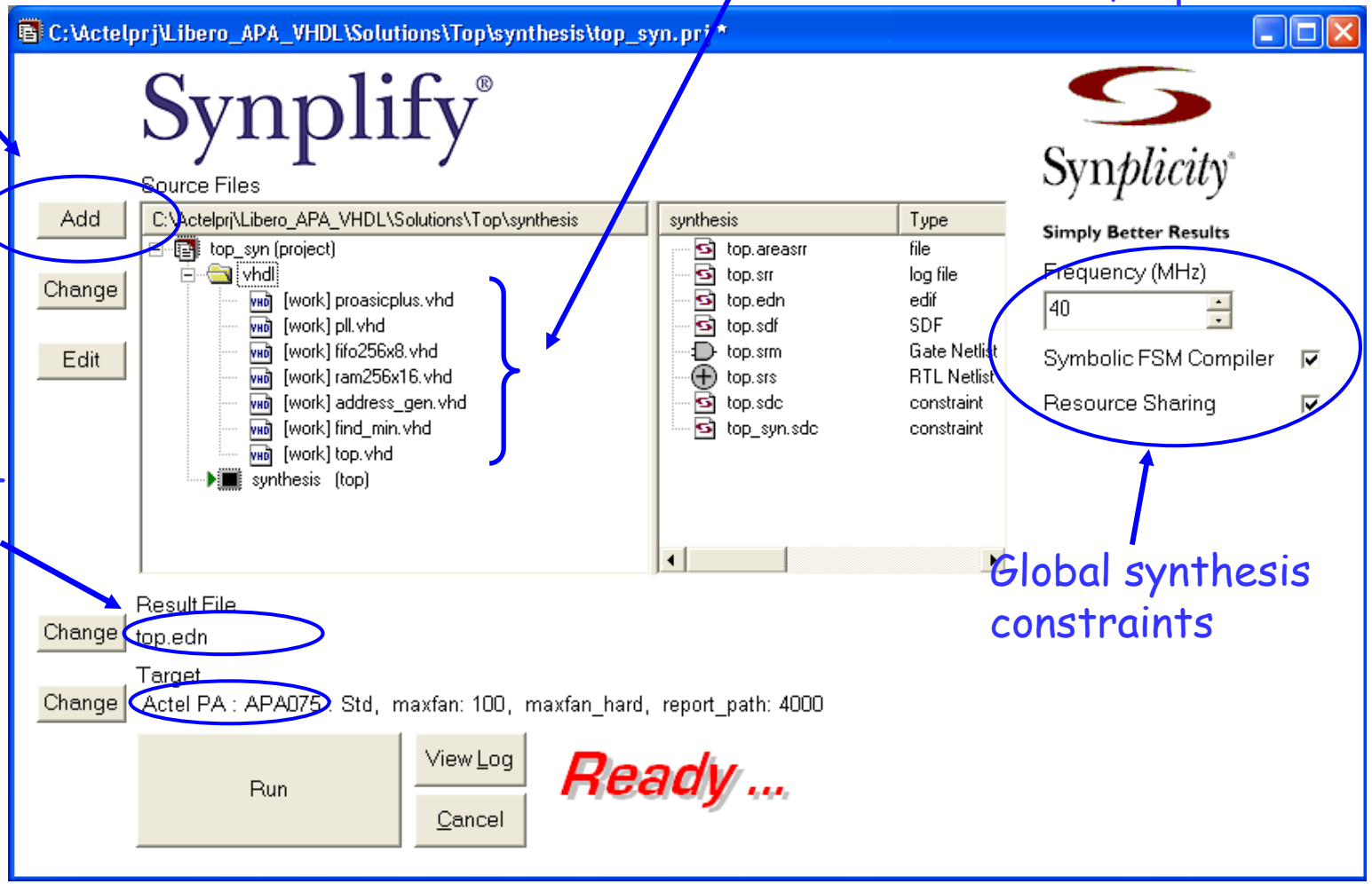
- Click “Synthesis” in Project Flow Window or ..



Synplify Interface

Add constraint files or VHDL packages

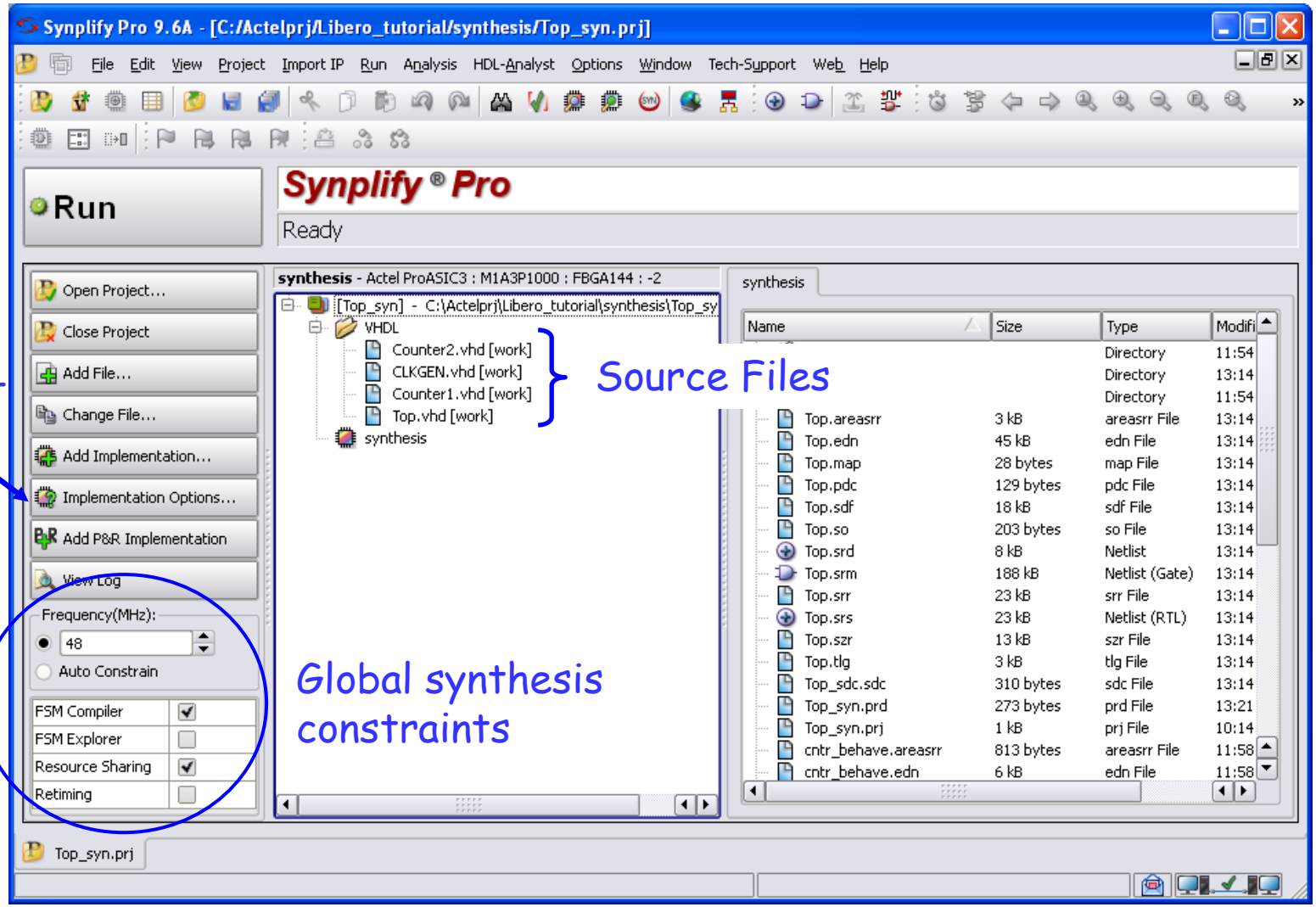
Libero automatically lists files lowest levels first, top last



Change target and result file

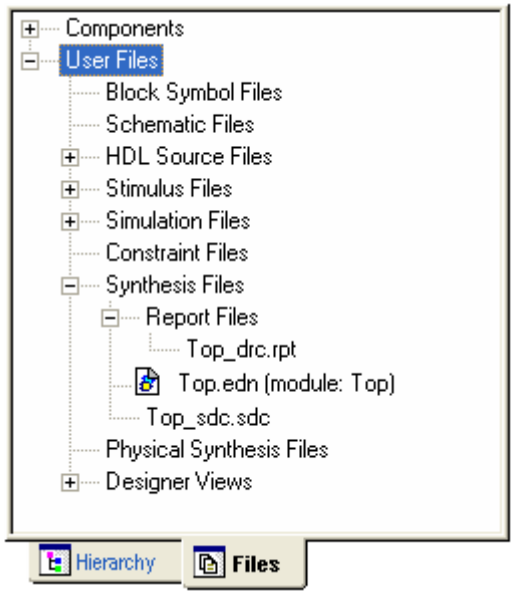
Global synthesis constraints

Synplify Pro Interface

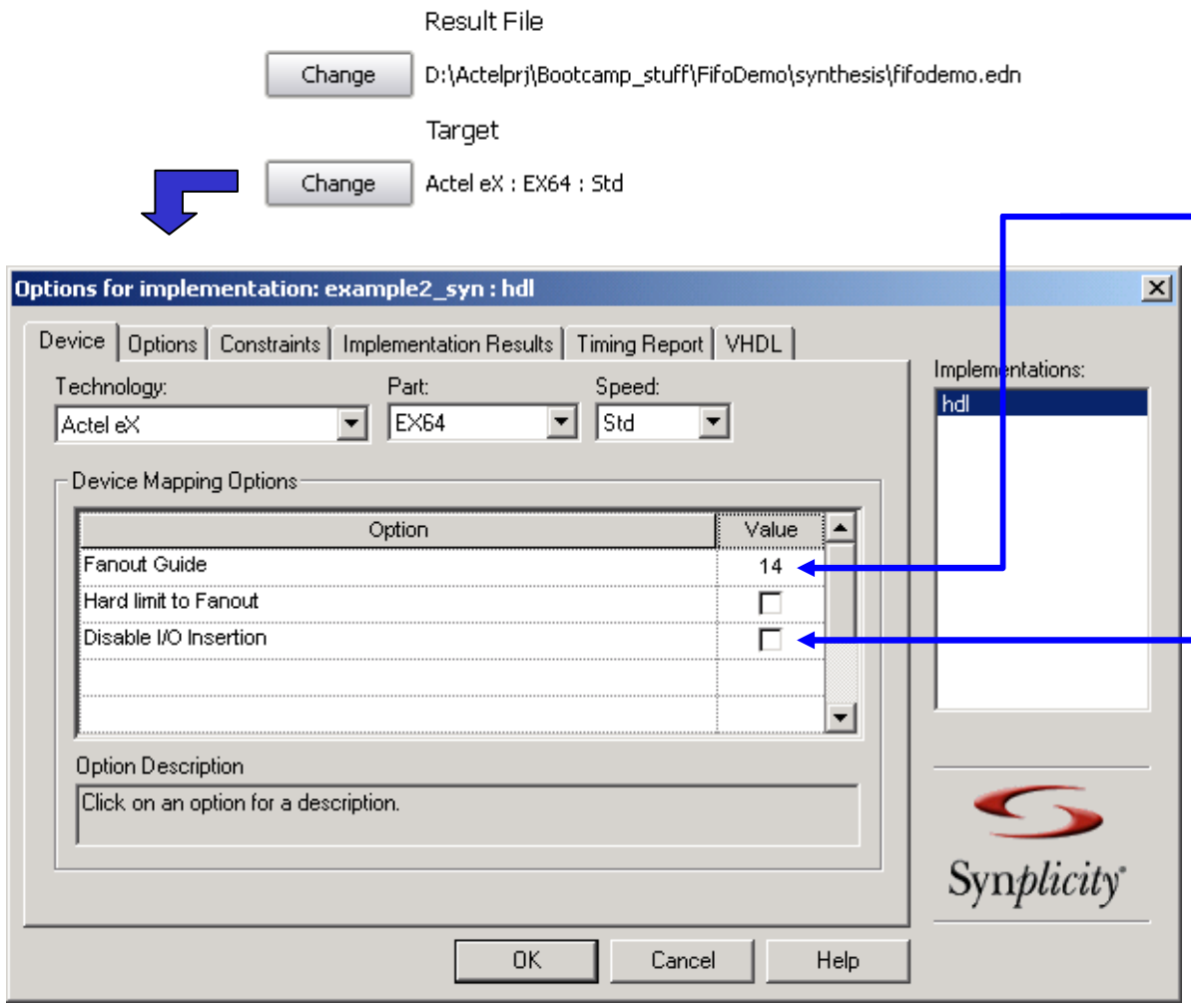


Result File

- Synplicity Produces an EDIF Netlist
  - <design>.edn
- Libero Automatically Produces Structural VHDL or Verilog Netlist
  - <design>.vhd or <design>.v
  - Results Appear on Files Tab under Synthesis Files



Setting the Target Options



High fanout = slow, small designs
Low fanout = fast, large designs
Use defaults for first pass

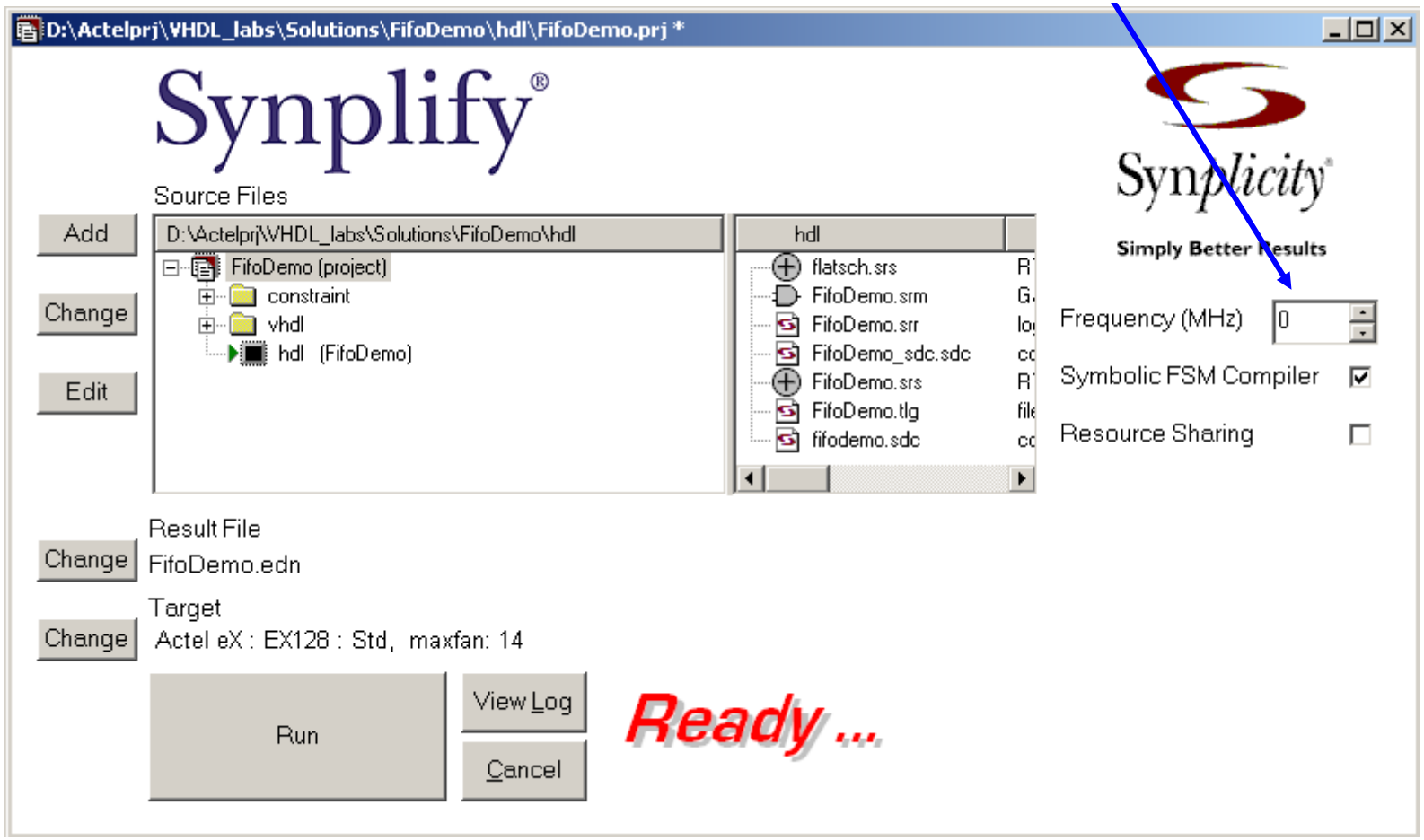
By default, Synplify will insert Actel I/O macros on all the HDL I/O ports.
When synthesizing lower-level blocks, this must be disabled.

Global Constraints

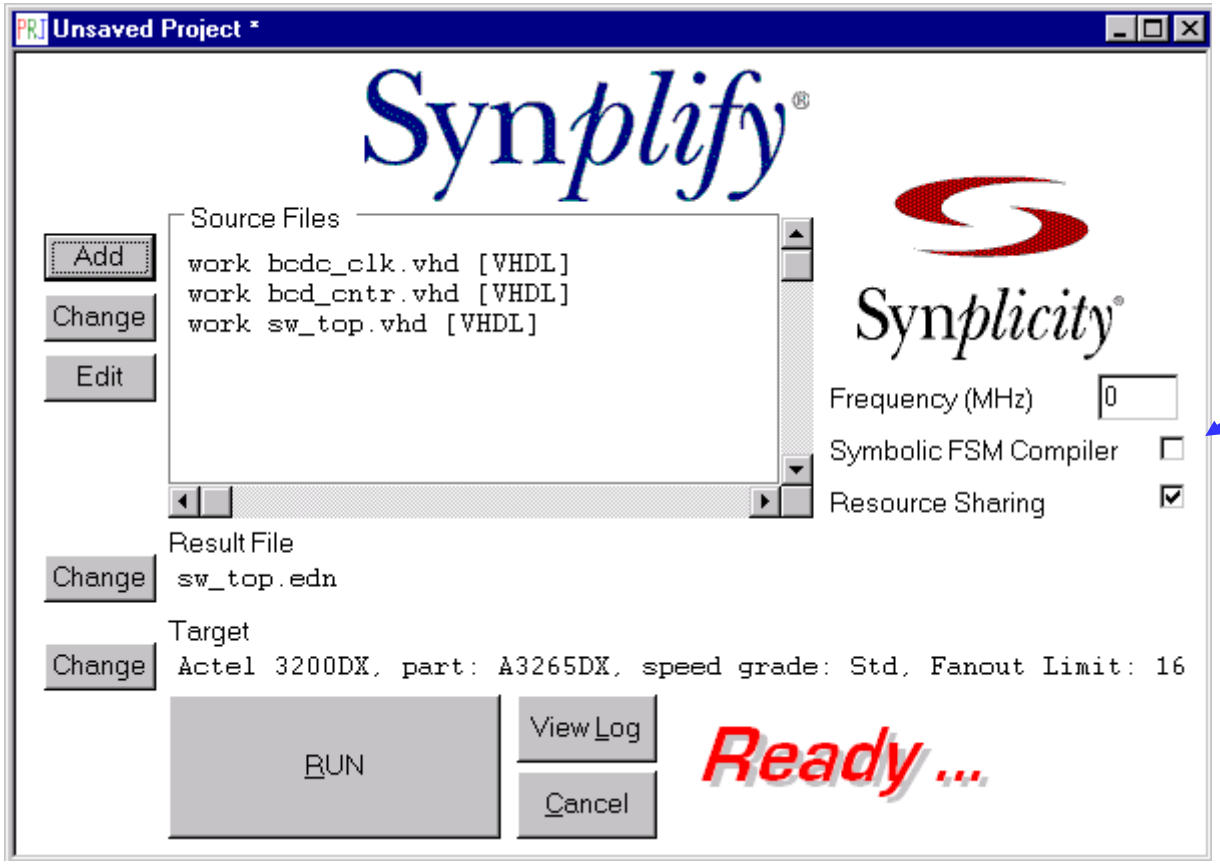
- Frequency
- Symbolic FSM Compiler
- Configure HDL Compiler
- Resource Sharing

Global Frequency

Global clock frequency is applied to all clock domains in the design
Low value means optimize for area



Symbolic FSM Compiler



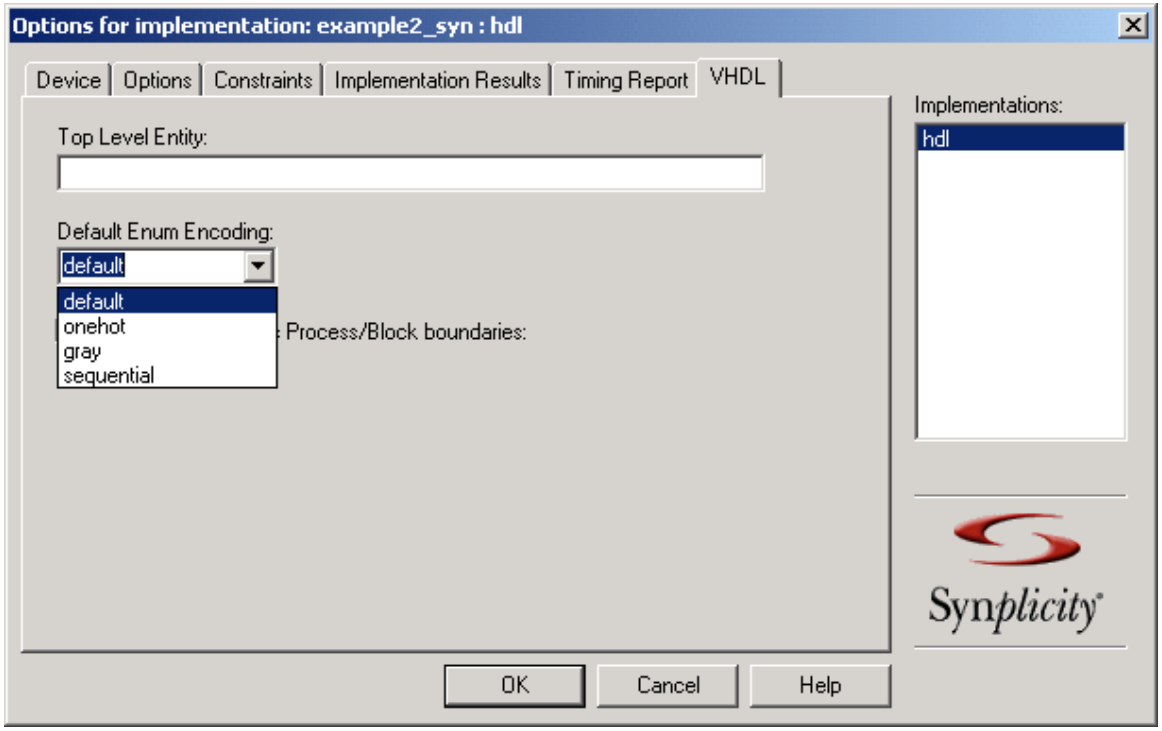
When checked Synplify identifies state machined in design and sets encoding

Encoding method can be set on individual state machines with **syn_encoding** attribute in the HDL code

State Machine Encoding (VHDL)

Options > Configure VHDL Compiler

Sets the default encoding style for enumerated types



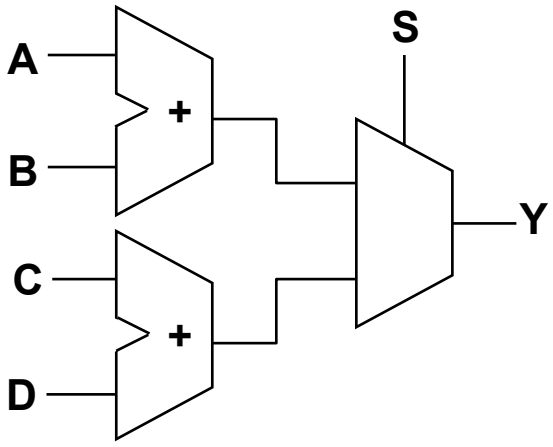
Override encoding style on an individual basis using syn_encoding directive in constraint editor or the HDL source code

# of states	default encoding
1 - 4	sequential
5 - 24	one-hot
> 24	gray

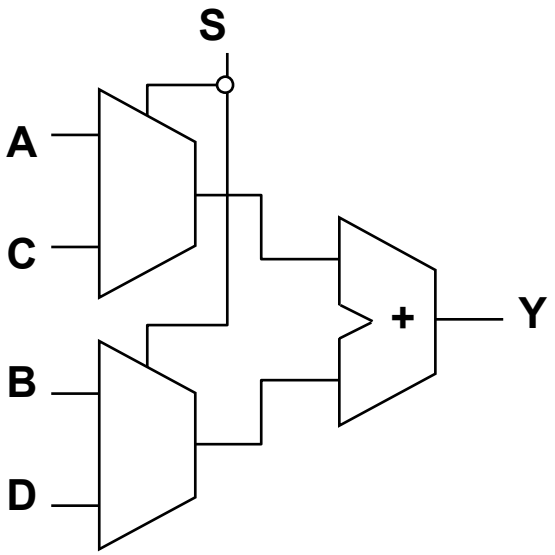
Resource Sharing

- When enabled, Synplify performs automatic sharing of operator resources, including adders, subtractors, incrementers, and decrementers.

```
if (s = '0') then
Y <= A + B;
else
Y <= C + D;
end if;
```

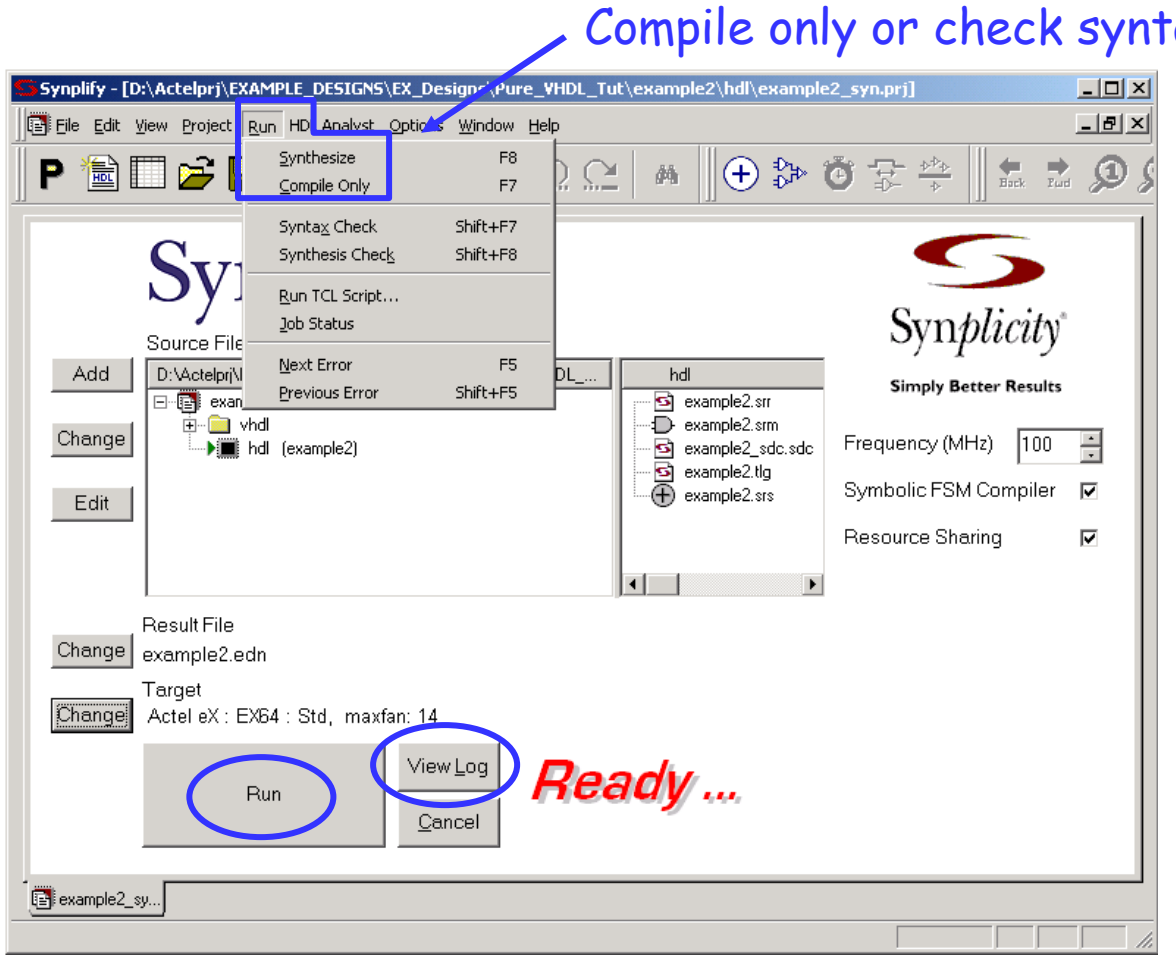


Without resource sharing

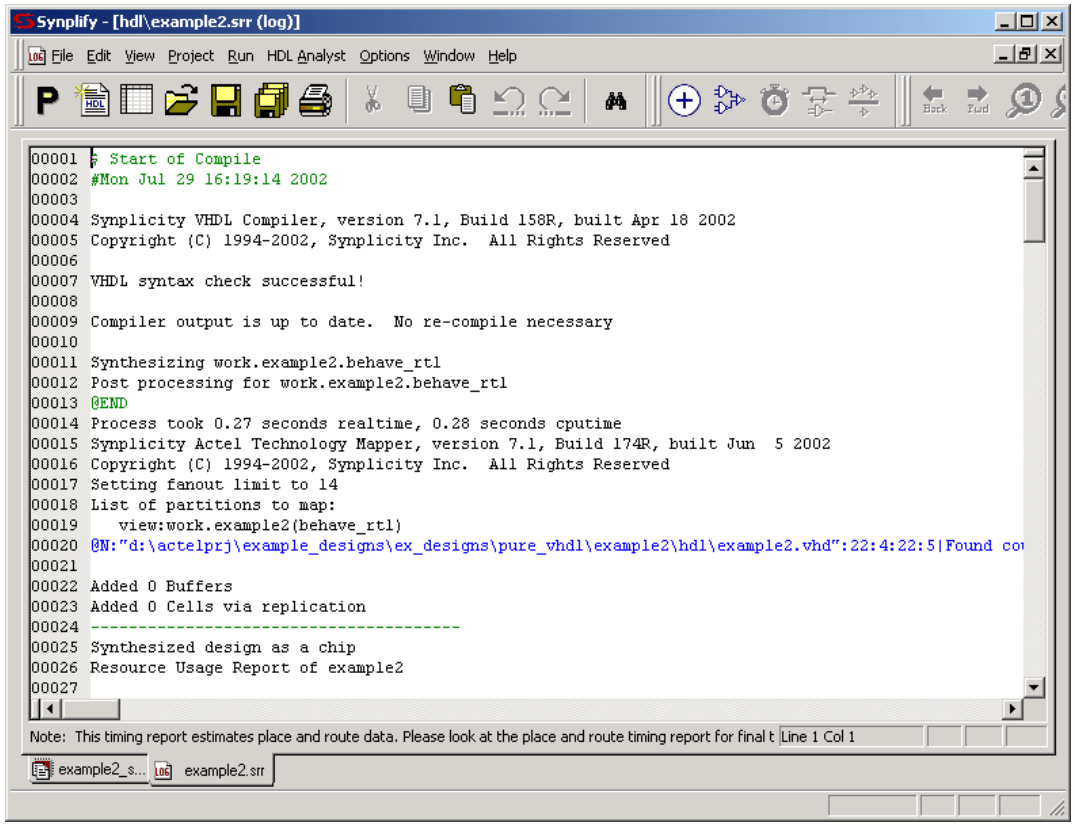


With resource sharing

Performing Synthesis

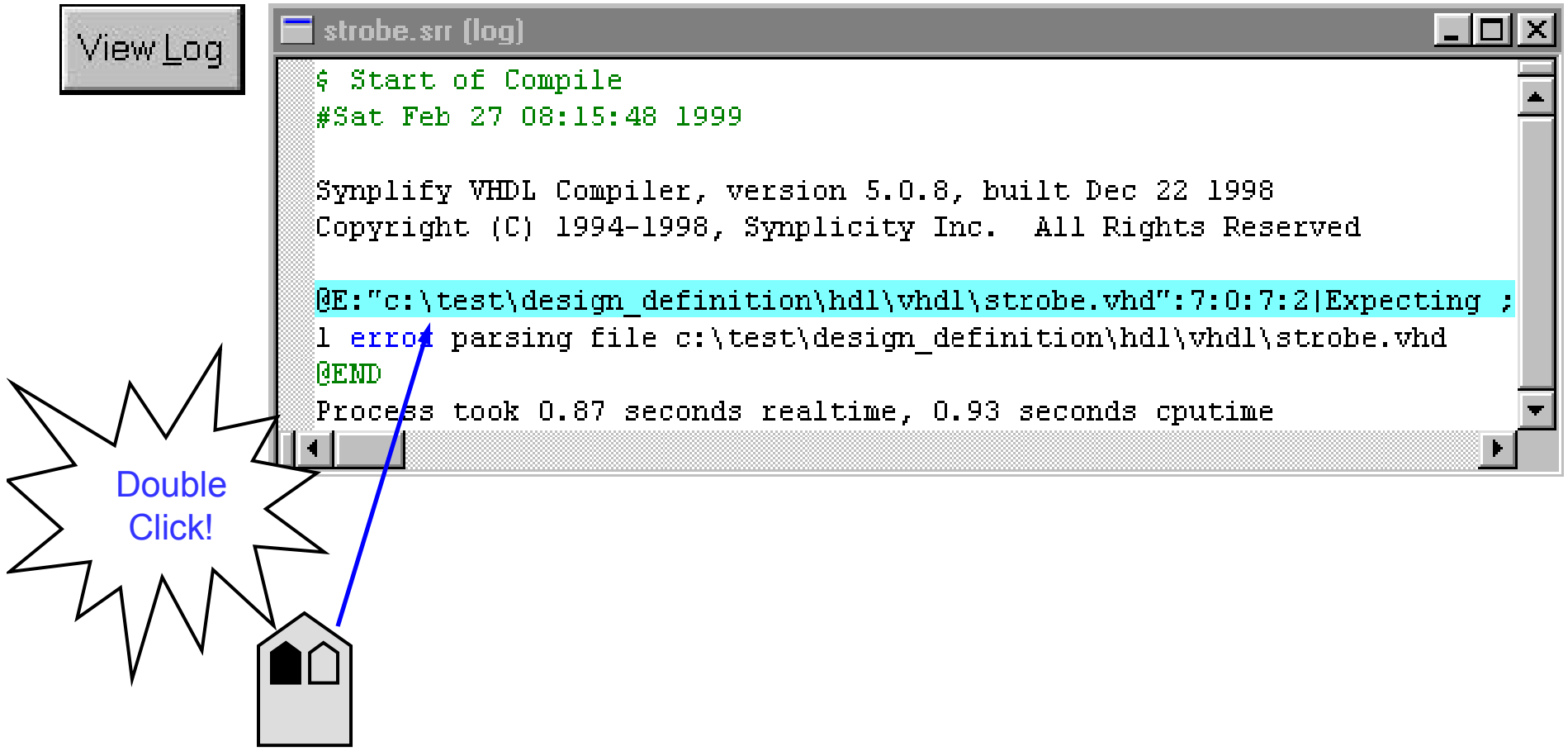


Synplify Log File

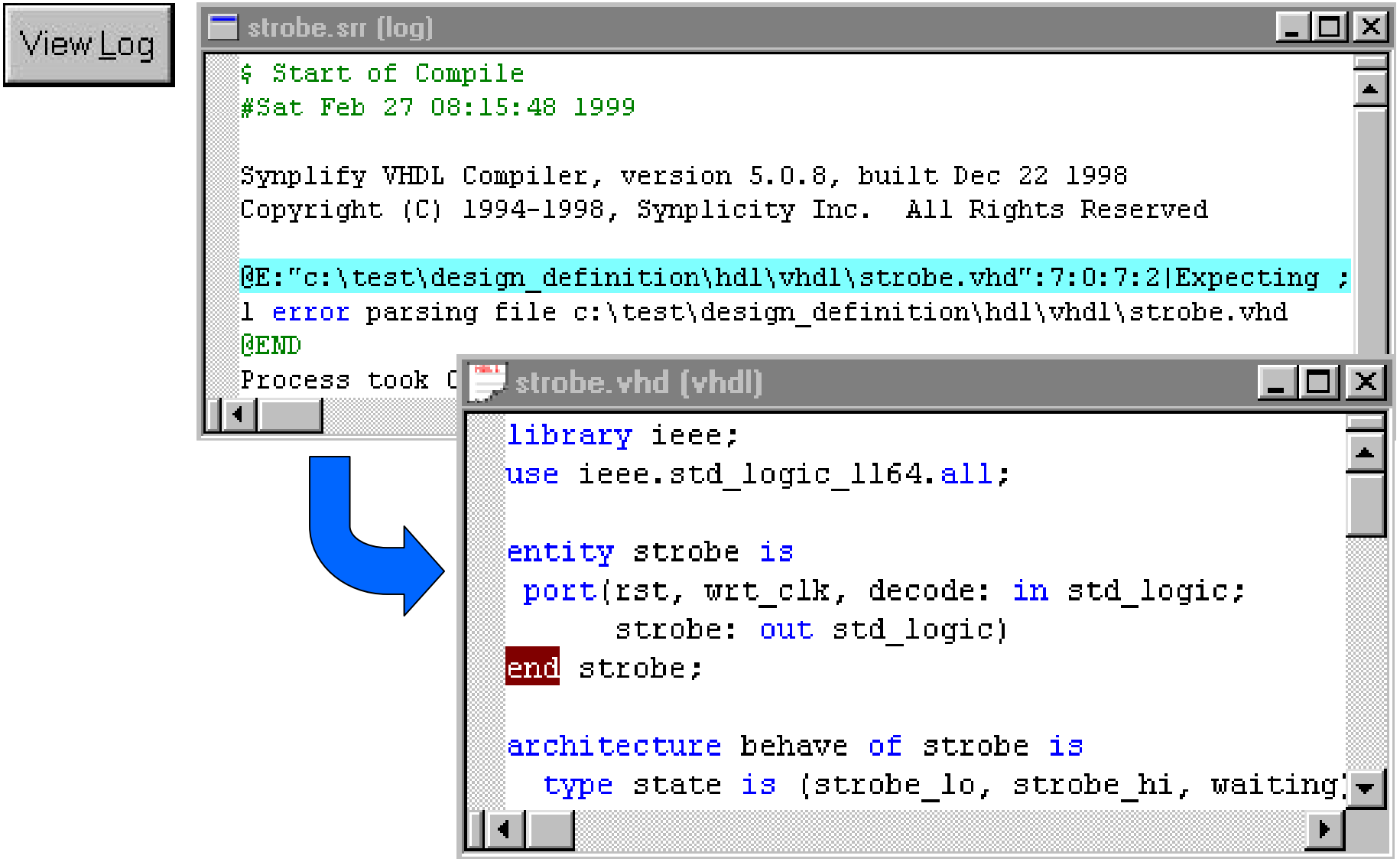


- Synplify Log Contains Valuable Information:
  - Warnings and Errors
    - Double-click and Jump to Code!
  - Fanout Limit
  - Extraction Information (Found Counter, FSM, Adder, etc.)
  - Net Loading
  - Logic Buffering and Replication Information
  - Resource Usage
  - Critical Path Timing Analysis

Reading the Log File
Errors

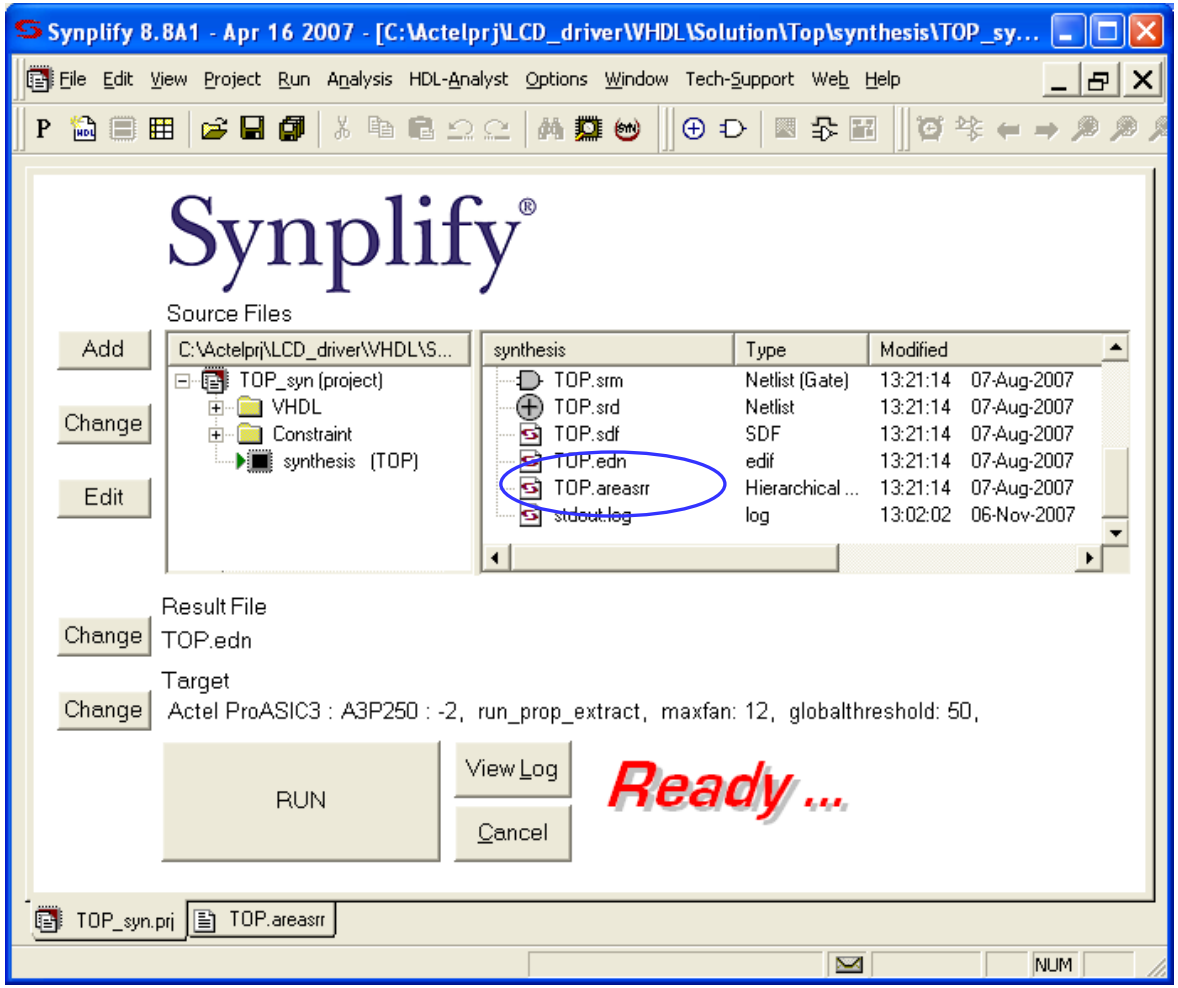


Reading the Log File
Errors

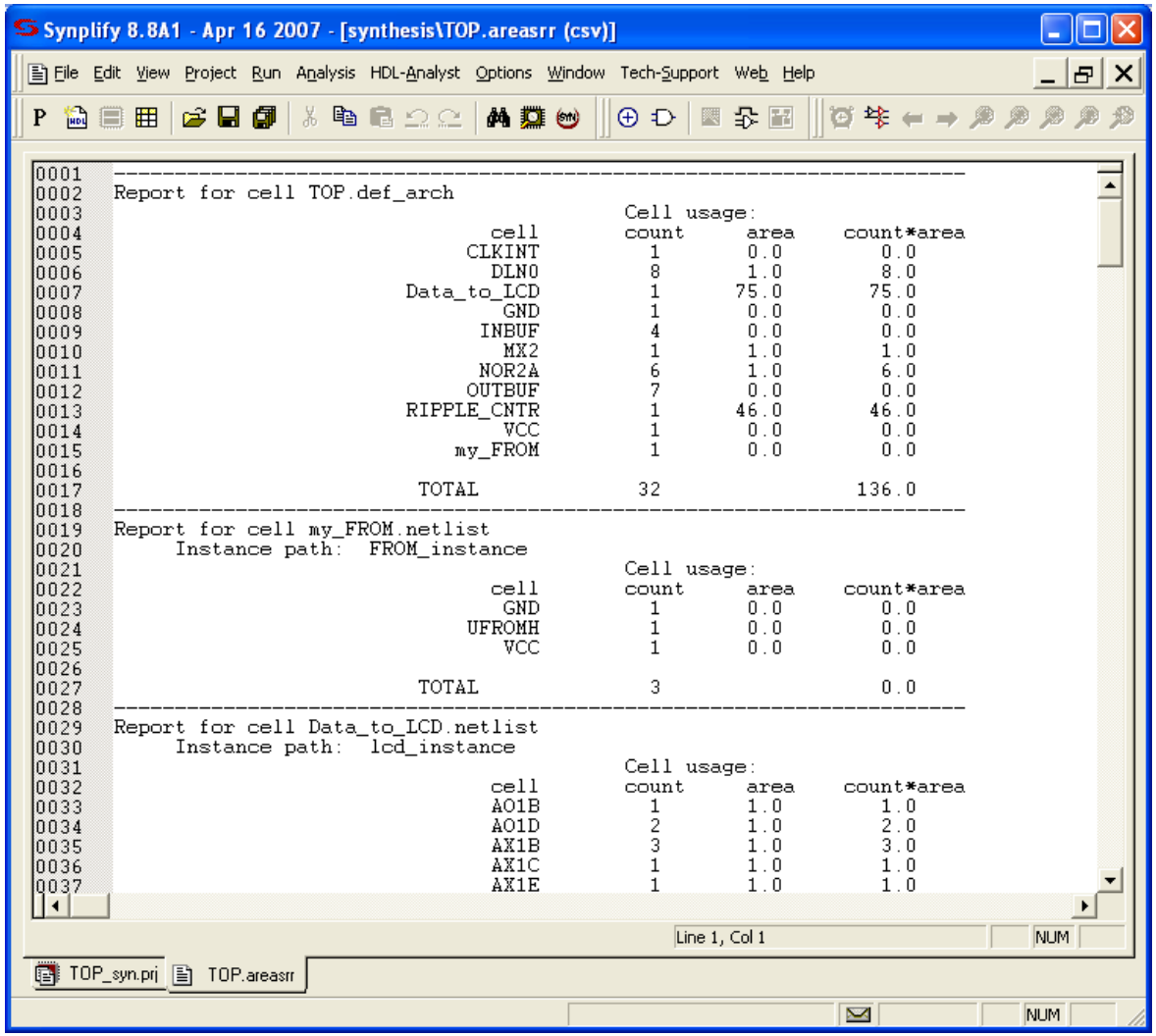


Synplify Area Log File

- Synplify Area Log Gives Hierarchical Breakdown of Area

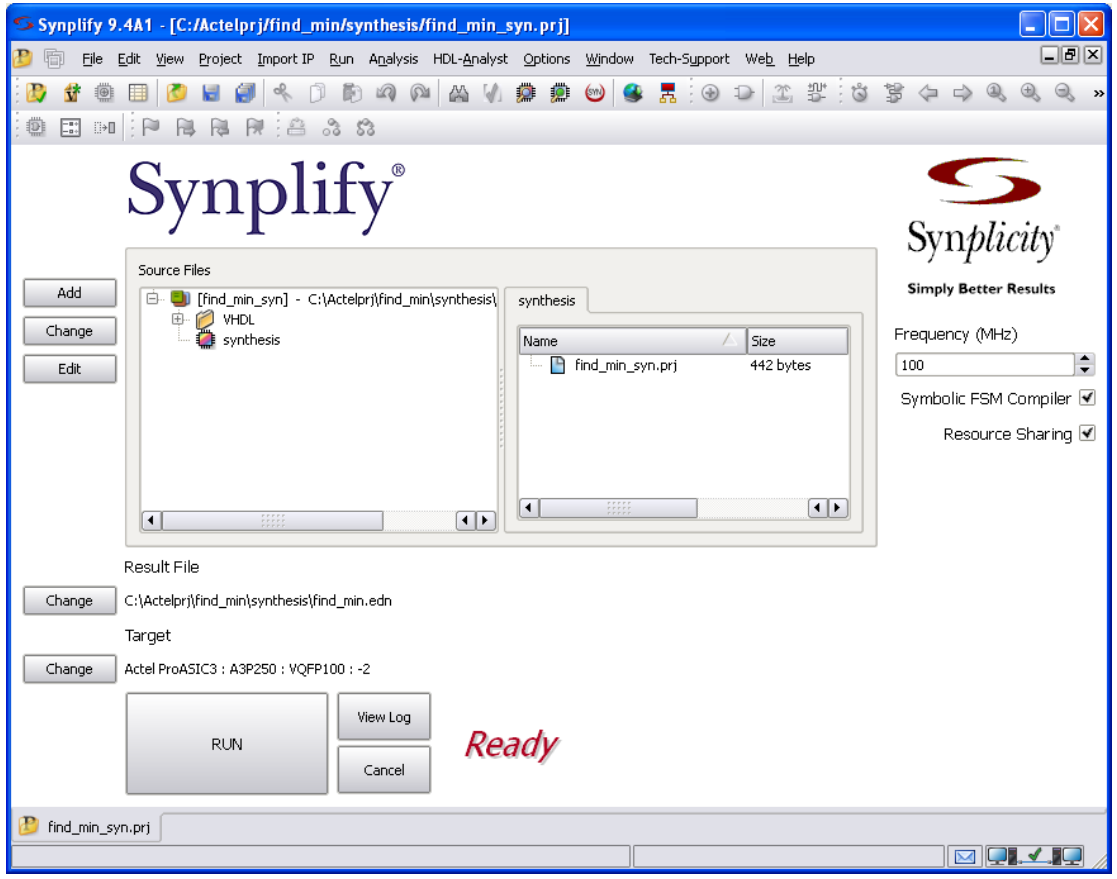


Synplify Area Log File



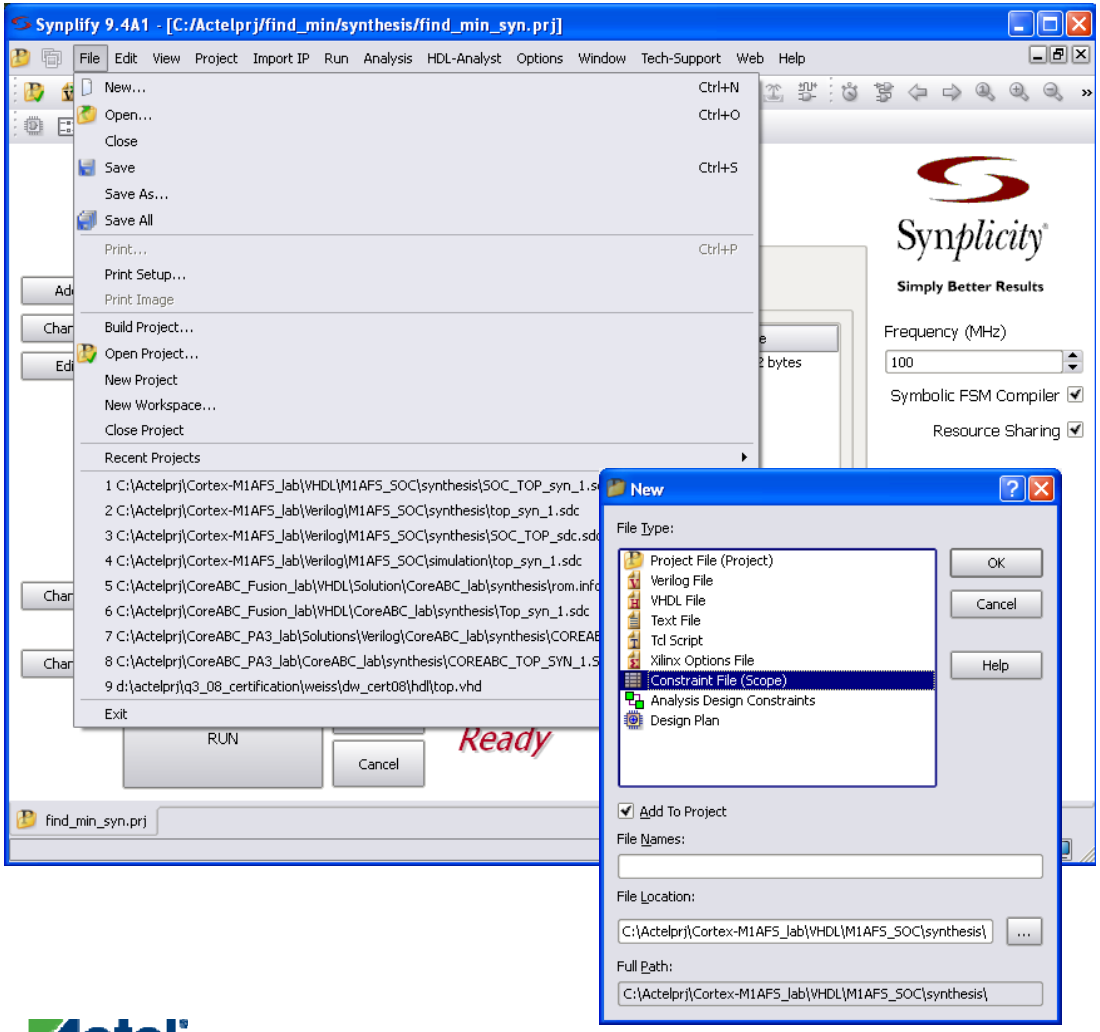

```
0001  ------------------------------------------------------------------------
0002  Report for cell TOP.def_arch
0003                                        Cell usage:
0004                                  cell       count      area     count*area
0005                                CLKINT           1       0.0        0.0
0006                                   DLN0          8       1.0        8.0
0007                           Data_to_LCD           1      75.0       75.0
0008                                   GND           1       0.0        0.0
0009                                 INBUF           4       0.0        0.0
0010                                   MX2           1       1.0        1.0
0011                                 NOR2A           6       1.0        6.0
0012                                OUTBUF           7       0.0        0.0
0013                            RIPPLE_CNTR          1      46.0       46.0
0014                                   VCC           1       0.0        0.0
0015                                my_FROM          1       0.0        0.0
0016
0017                                 TOTAL             32                136.0
0018  ------------------------------------------------------------------------
0019  Report for cell my_FROM.netlist
0020       Instance path:  FROM_instance
0021                                        Cell usage:
0022                                  cell       count      area     count*area
0023                                   GND           1       0.0        0.0
0024                                UFROMH           1       0.0        0.0
0025                                   VCC           1       0.0        0.0
0026
0027                                 TOTAL             3                 0.0
0028  ------------------------------------------------------------------------
0029  Report for cell Data_to_LCD.netlist
0030       Instance path:  lcd_instance
0031                                        Cell usage:
0032                                  cell       count      area     count*area
0033                                  AO1B           1       1.0        1.0
0034                                  AO1D           2       1.0        2.0
0035                                  AX1B           3       1.0        3.0
0036                                  AX1C           1       1.0        1.0
0037                                  AX1E           1       1.0        1.0
```

Synplify Constraint Editor (SCOPE)



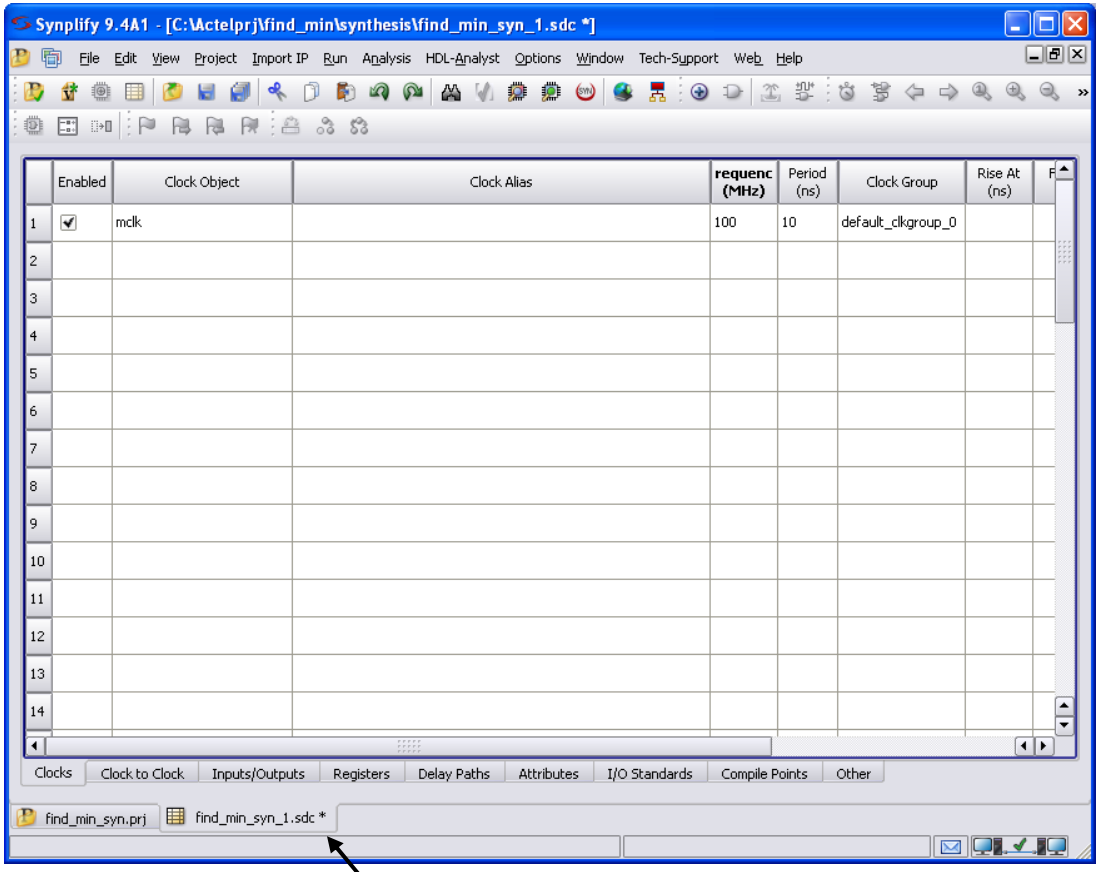
- Synplify and Synplify Pro Include a Spreadsheet-like Constraint Editor
- Extremely Easy to Use

Synplify Constraint Editor (SCOPE)



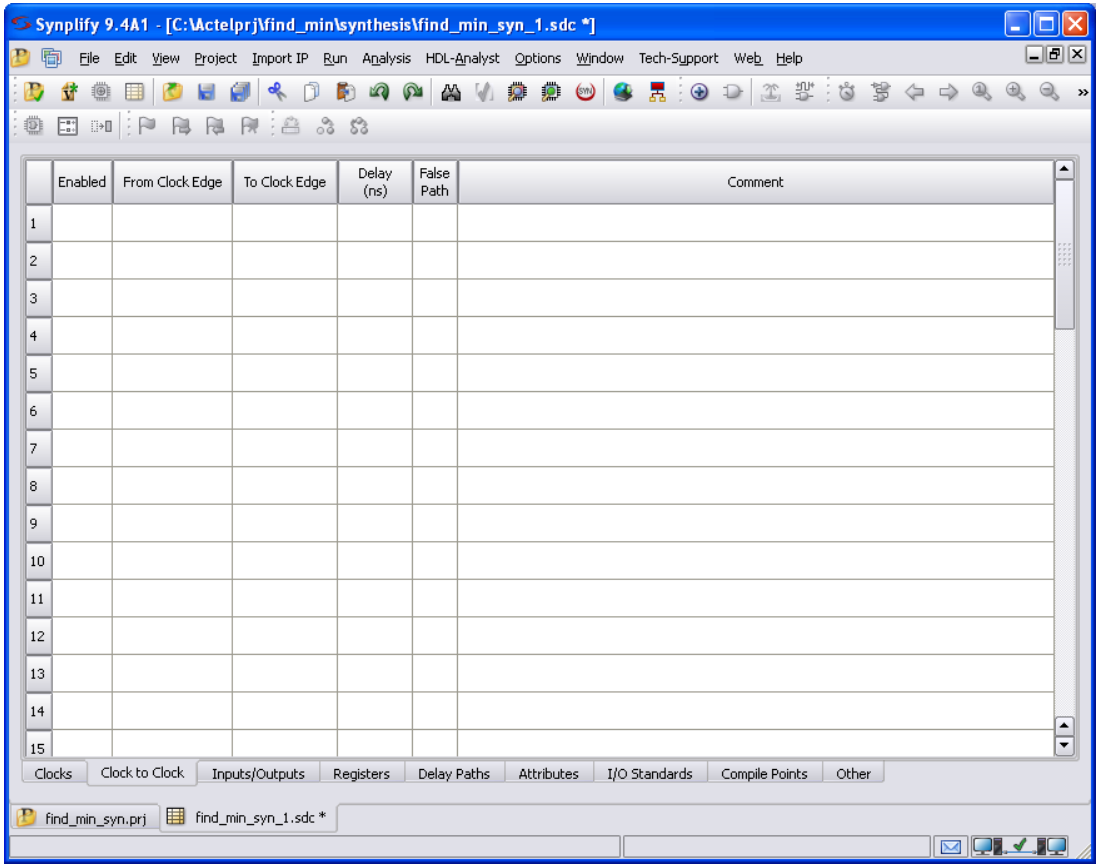
- Select File >New
- Select Constraint File (Spreadsheet)
  - Select Files of type Constraint Files
- Press OK

Constraint Editor (Cont.)



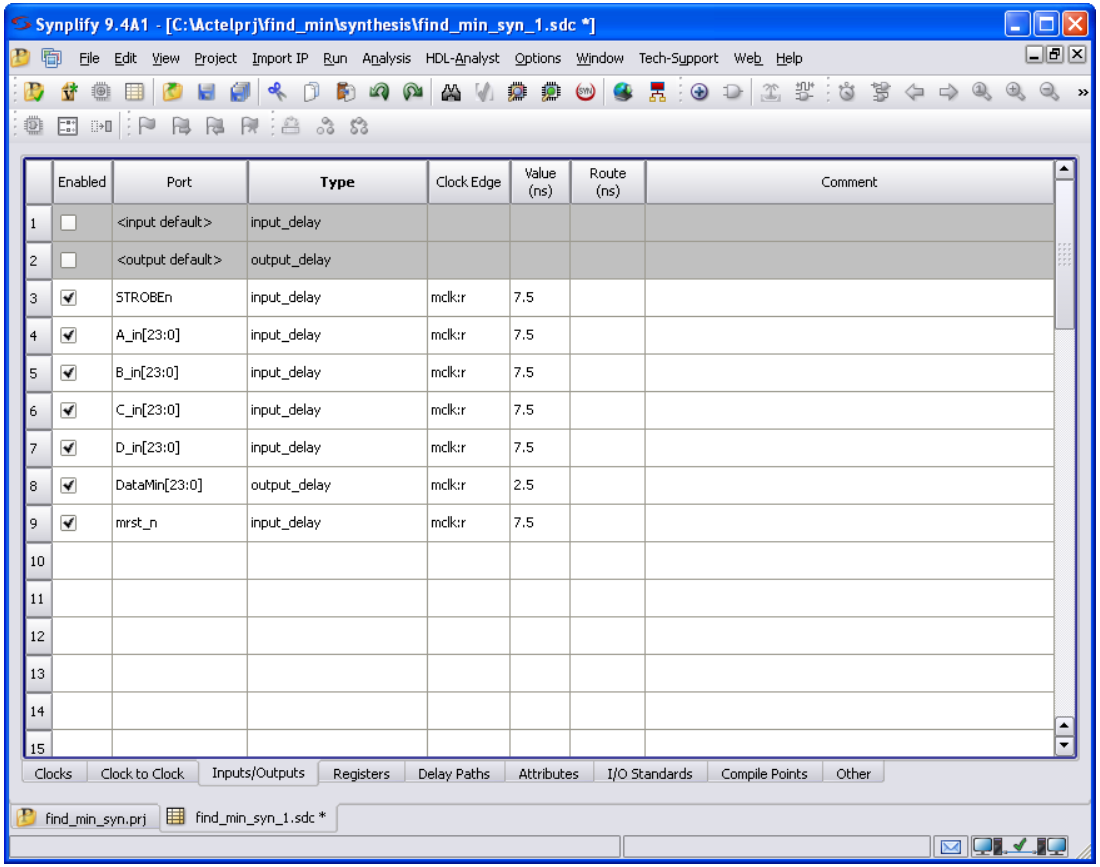
- Constraint Editor tabs Support:
  - Clock Frequency or Period

Constraint Editor (Cont.)



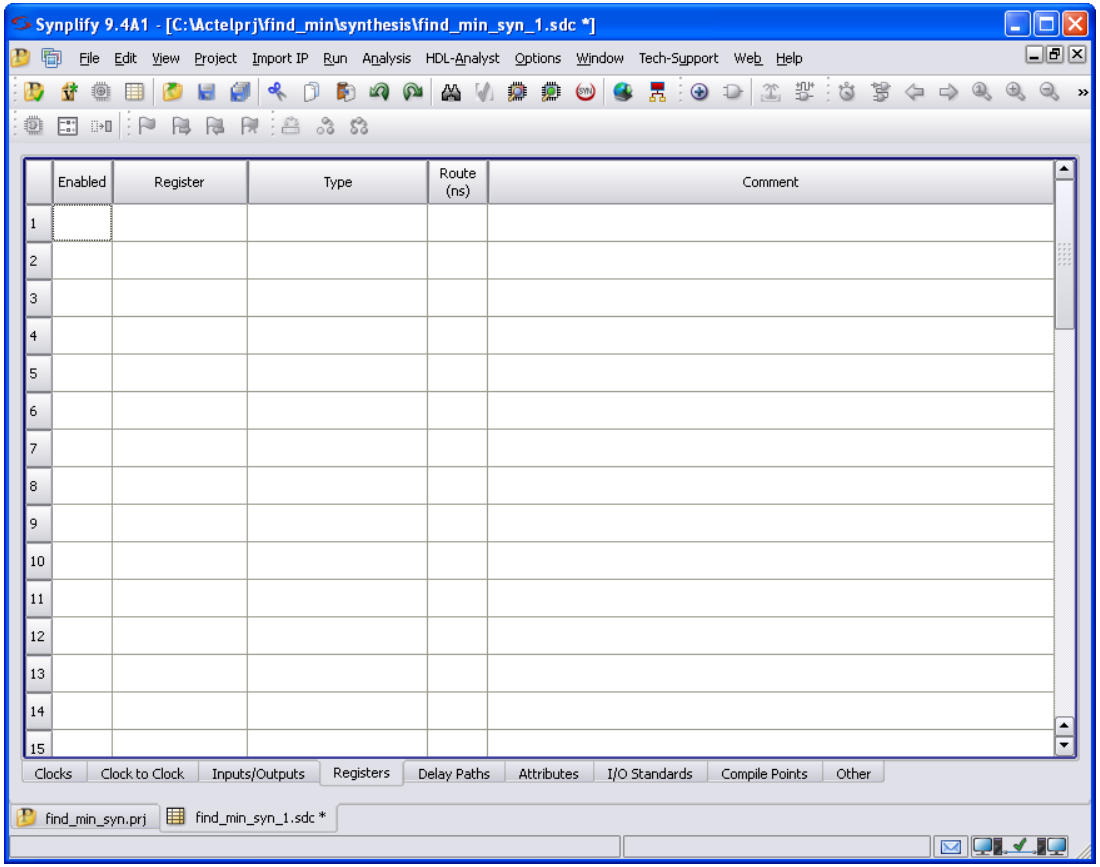
- Constraint Editor tabs Support:
  - Clock Frequency or Period
  - Clock to Clock timing

Constraint Editor (Cont.)



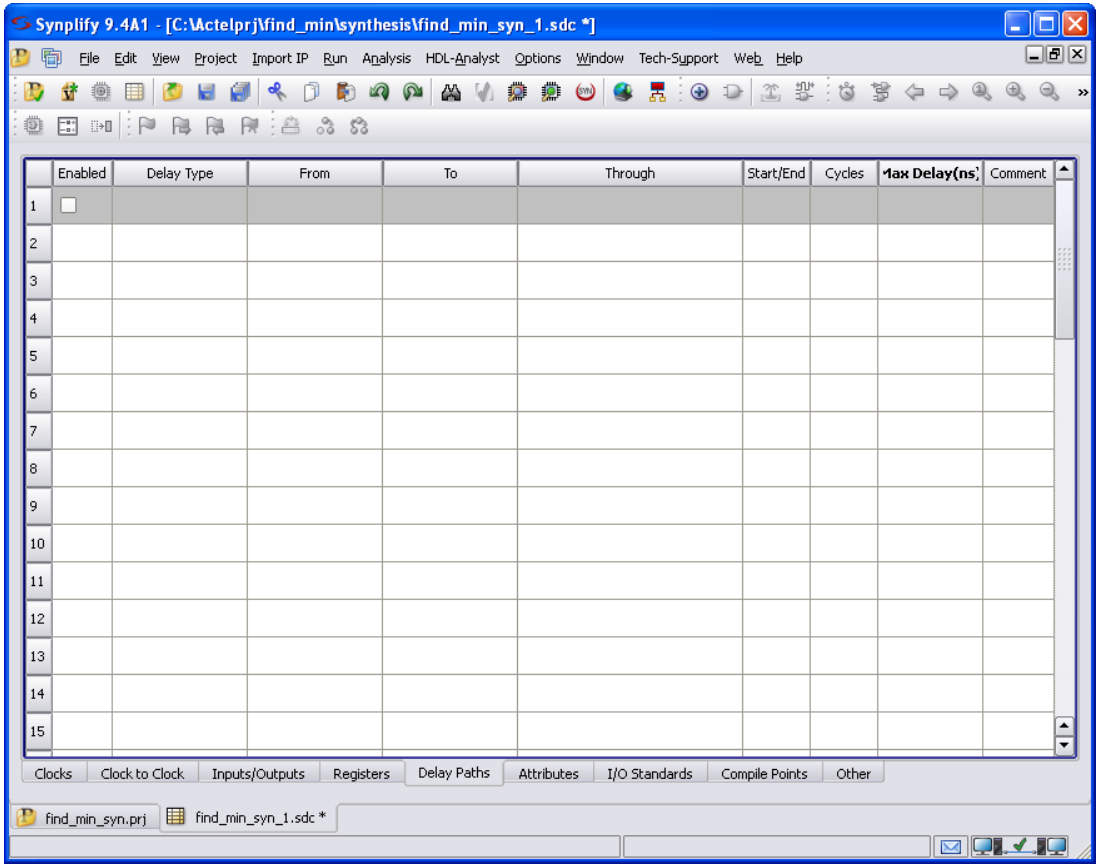
- Constraint Editor tabs Support:
  - Clock Frequency or Period
  - Clock to Clock timing
  - Input/Output Constraints

Constraint Editor (Cont.)



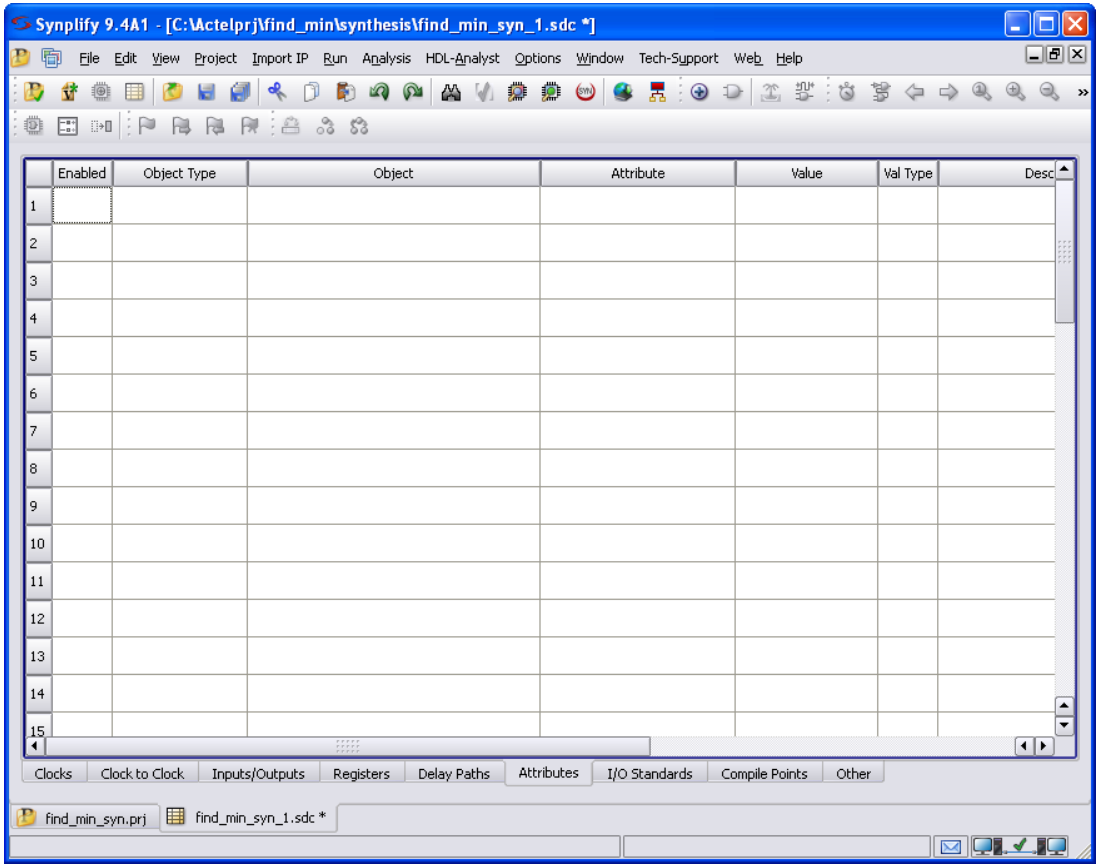
- Constraint Editor tabs Support:
  - Clock Frequency or Period
  - Clock to Clock timing
  - Input/Output Constraints
  - Register Constraints

Constraint Editor (Cont.)



- Constraint Editor tabs Support:
  - Clock Frequency or Period
  - Clock to Clock timing
  - Input/Output Constraints
  - Register Constraints
  - Timing Exceptions

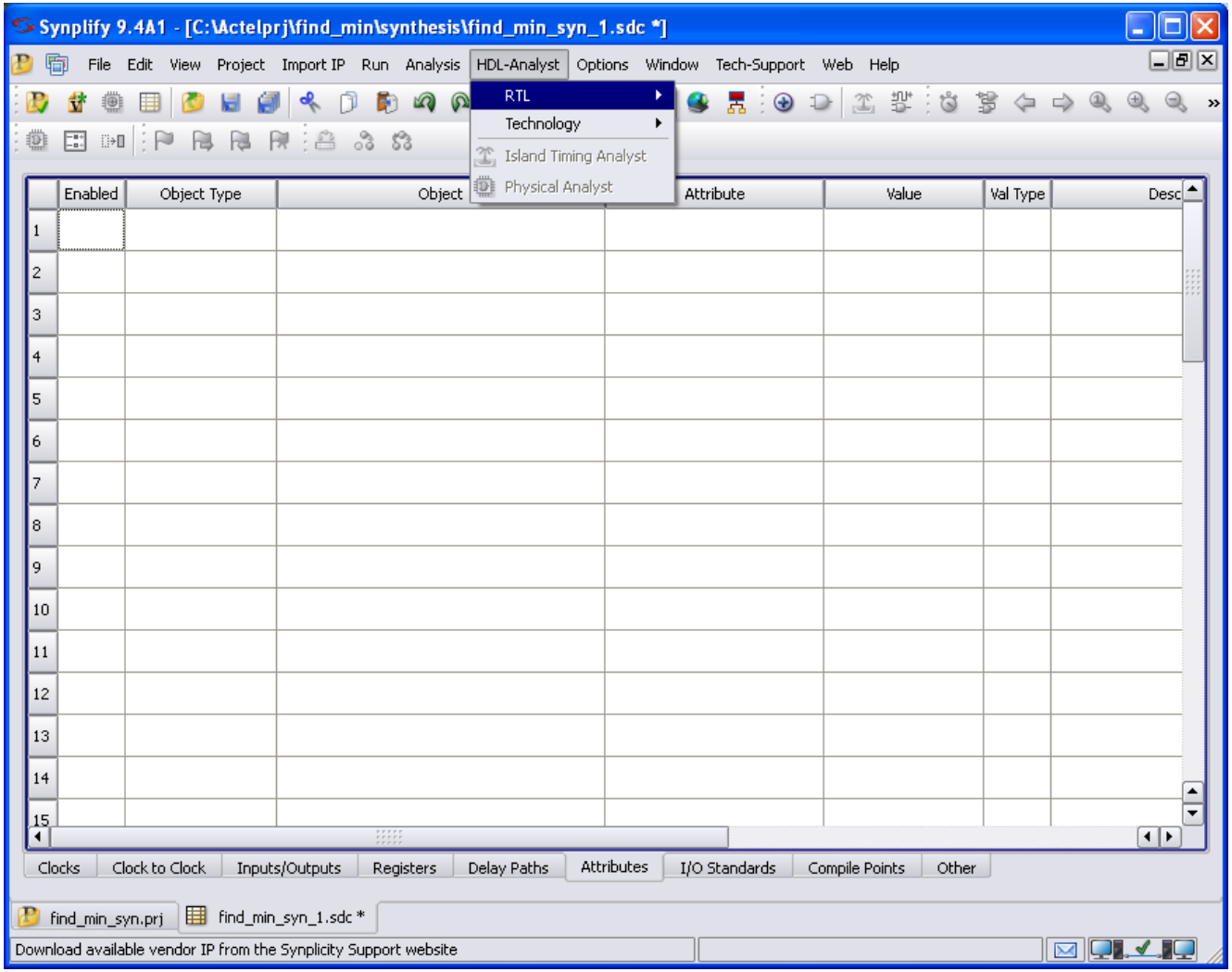
Constraint Editor (Cont.)



- Constraint Editor tabs Support:
  - Clock Frequency or Period
  - Clock to Clock timing
  - Input/Output Constraints
  - Register Constraints
  - Timing Exceptions
  - Synthesis Attributes

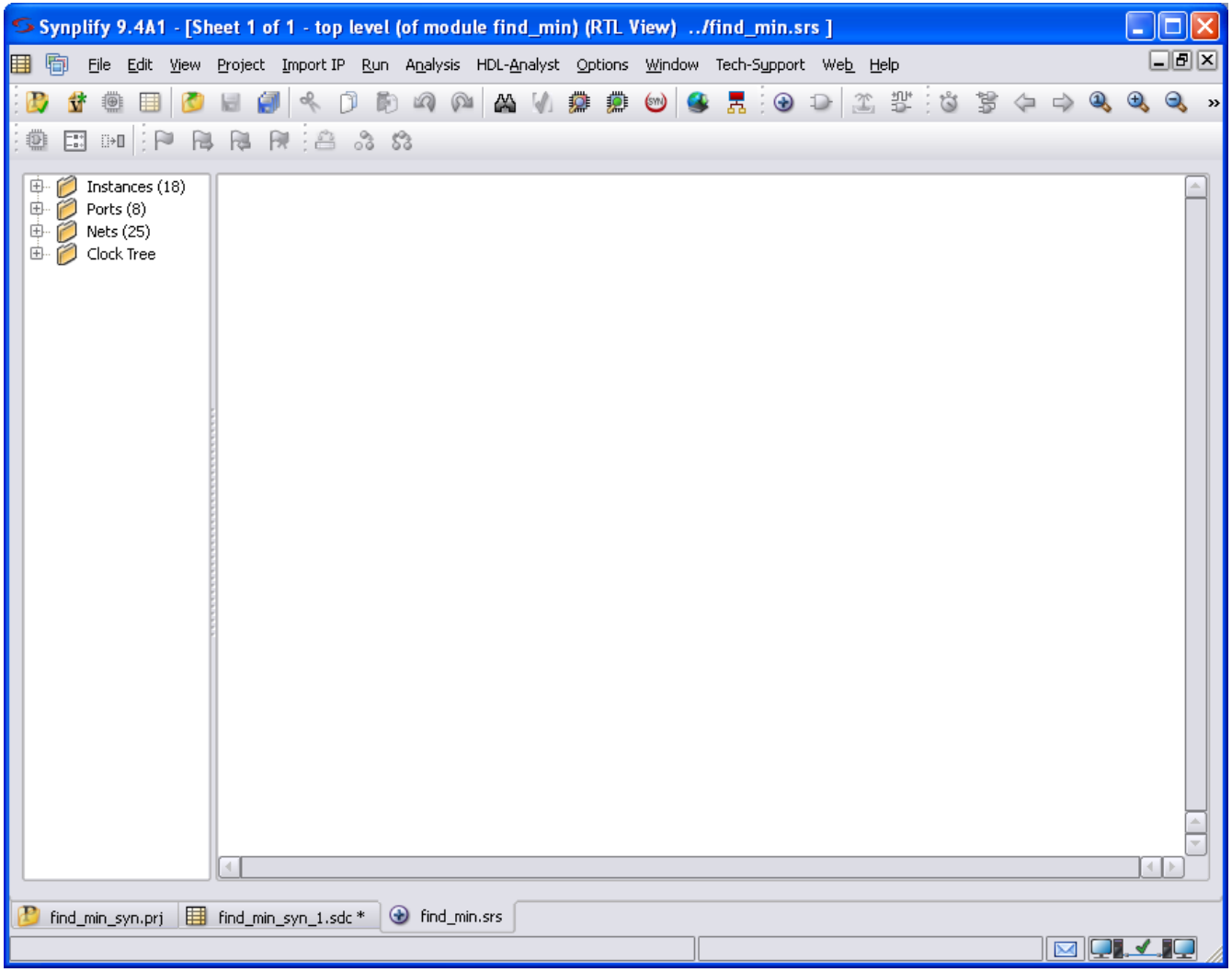
Constraint Editor

Drag and Drop Interface



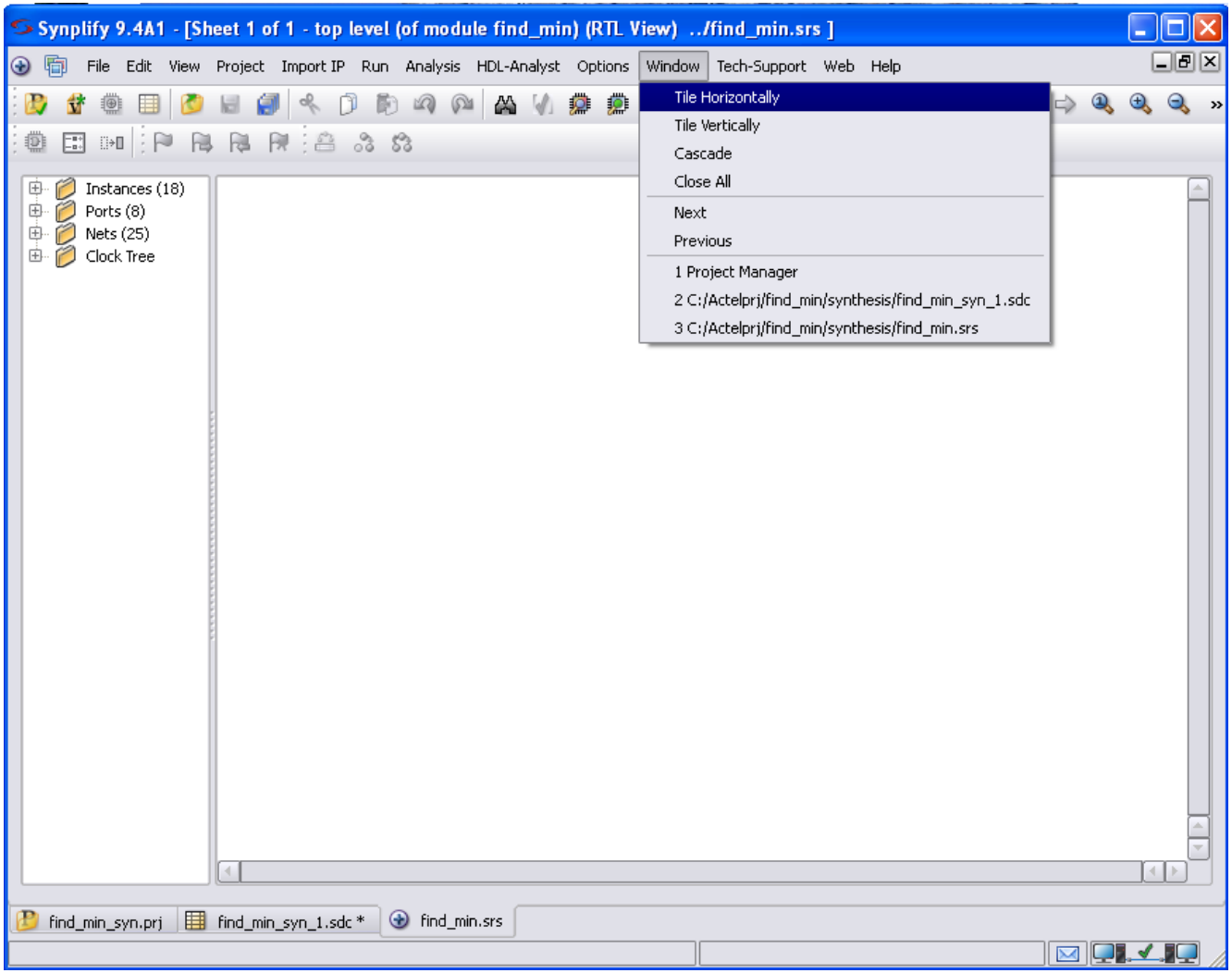
Constraint Editor

Drag and Drop Interface (Cont)



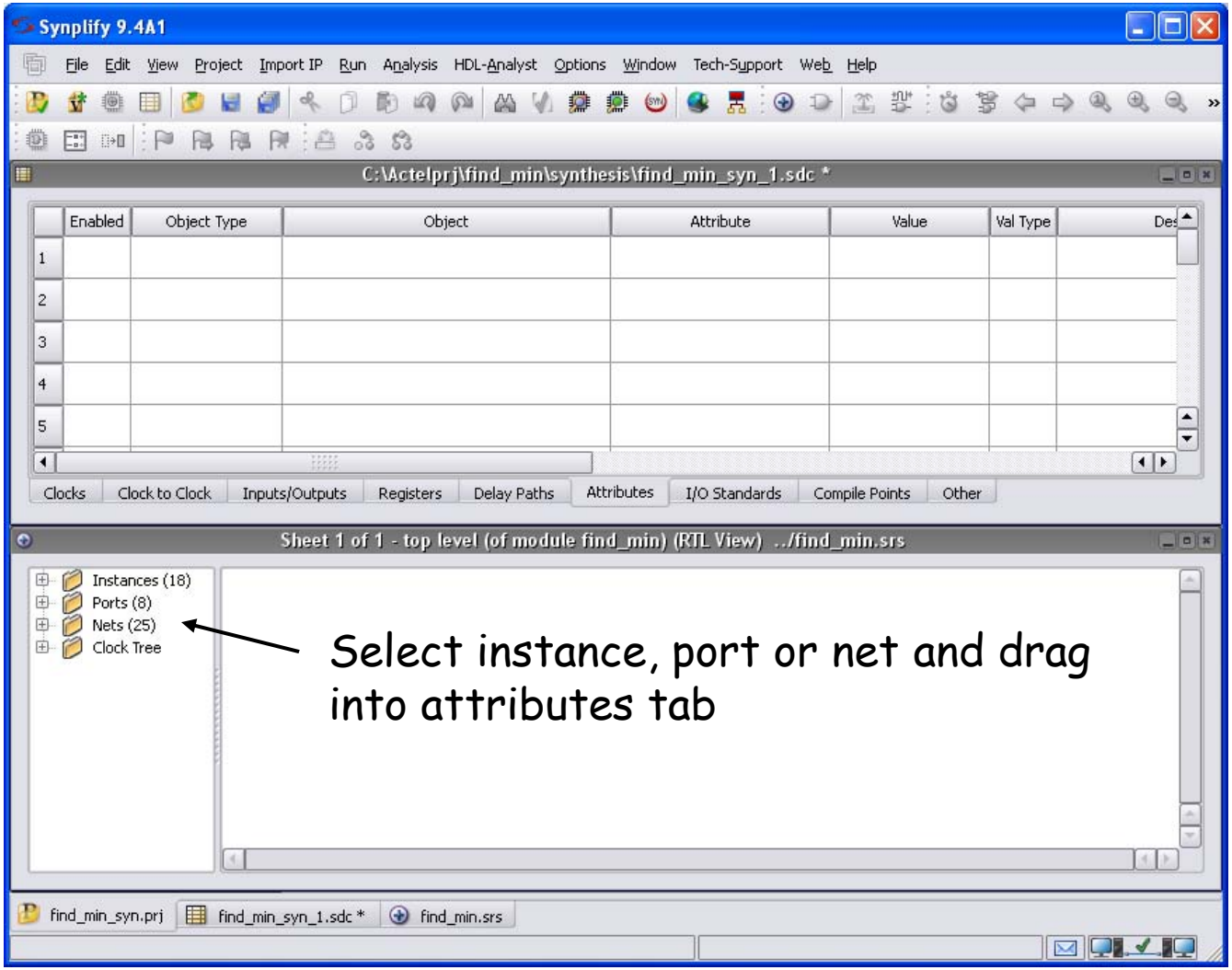
Constraint Editor

Drag and Drop Interface (Cont)



Constraint Editor

Drag and Drop Interface (Cont)



Synplify Directives and Attributes

Synplicity Directives and Attributes

- Let You Direct Analysis, Optimization, and Mapping of Design during Synthesis
- Attributes Control *Mapping* Optimizations
  - Attributes Can Be Entered in the Constraint Editor or HDL Source Code
    - Synplify Supports Limited Number of Attributes that Can Be Entered in Attribute Pane of the SCOPE Editor
    - *Most* Attributes Are Entered in your VHDL or Verilog Code
- Directives Control *Compiler* Optimizations
  - Directives *Must* Be Entered in HDL Source Code

Actel Attribute and Directive Summary

NAME	TYPE	VALUE	DESCRIPTION
syn_sharing	D	1/0 true / false	Enables / disables the resource sharing operators inside a module during synthesis
syn_black_box	D	1/0 true / false	Specifies that a module or component is a black box with only its interface defined for synthesis. The contents of a black box cannot be optimized during synthesis
syn_keep	D		Prevents an internal signal from being removed during synthesis and optimization
alspreserve	A	Boolean	Prevents a net from being removed during Place and Route
syn_encoding	A	sequential, onehot, gray, safe	Specifies encoding style for state machines
syn_maxfan	A	integer	Controls the maximum fanout or an instance, net or port
syn_noclockbuf	A	1/0 true / false	Disables automatic insertion of clock buffers
syn_radhardlevel	A	string	Specifies register design technique to apply to a module, architecture or instance
syn_replicate	A	1/0	Disables register replication
syn_insert_buffer	A	1/0 true / false	Directs Synplify to use specific I/O pads
syn_global_buffers	A	integer	Specifies the mumber of global buffers to be used in a ProASIC3/E, IGLOO/e or Fusion design
syn_ram_style	A		Specifies the implementation to use for an inferred RAM

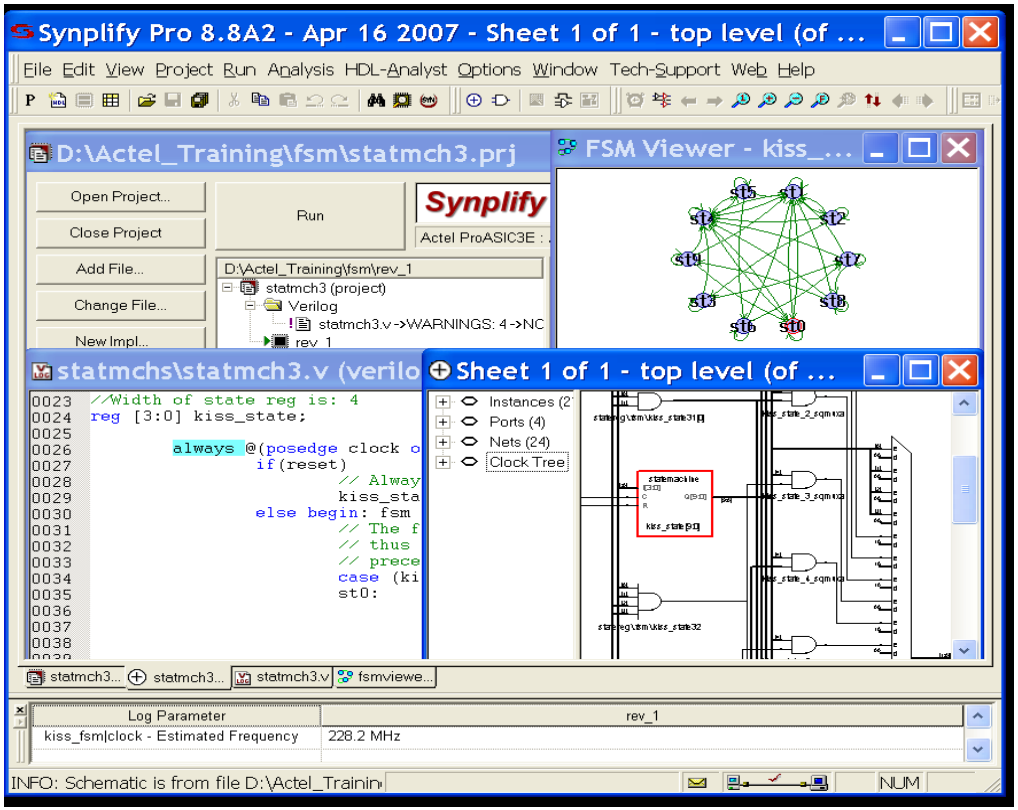
Synplify and Synplify PRO for Actel

- Synplify for Actel is Equivalent to Synplicity's Synplify Product
  - Included in Libero Gold
  - Limited to Actel Products Only
- Synplify Pro AE has Additional Features
  - Included in Libero Platinum
  - Limited to Actel products Only

Synplify Pro Logic Synthesis

- FSM Explorer
- Graphical State Machine Viewer
- Pipelining of Multipliers and ROMs
- Compile Point Creation
- Generic Cross-Probing of Critical Paths
- HDL Analyst®
- STAMP Support
- Multiple Implementations & Workspace Support
- Mixed Language design Entry
- Re-timing

Synplify Pro®

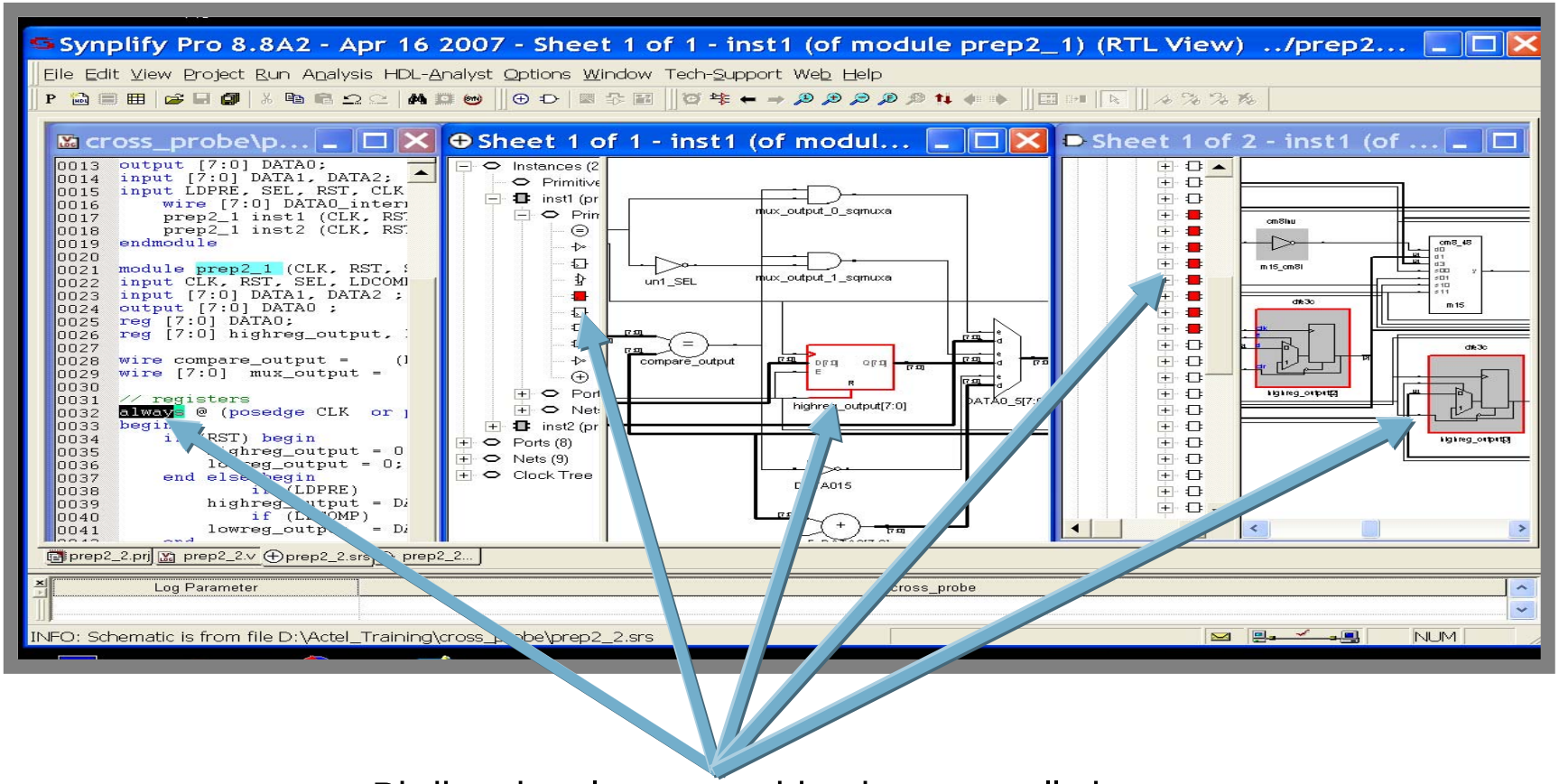


Synplify Pro
Powerful Graphical Debugging

Language Sensitive Editor
Describe the design functionality

Unique RTL View
Analyze a technology-independent block diagram

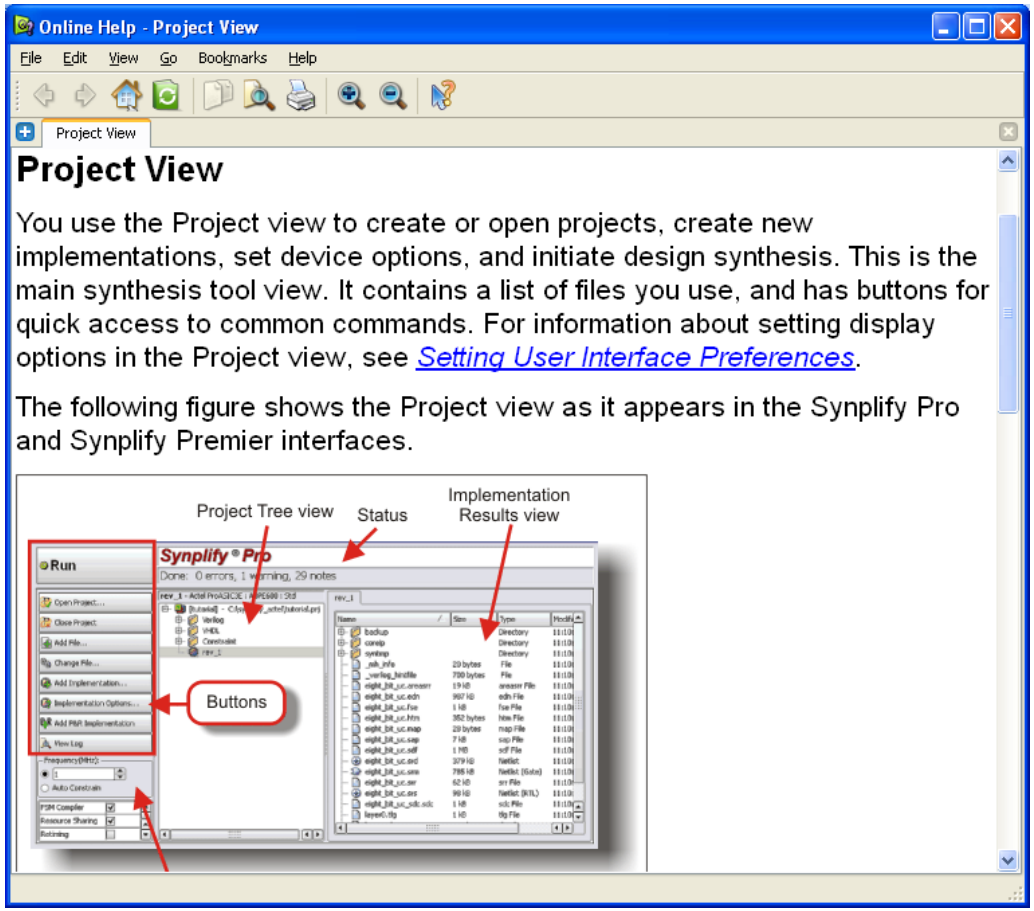
Technology View
View post-mapped schematic with annotated timing



Bi-directional cross-probing between all views

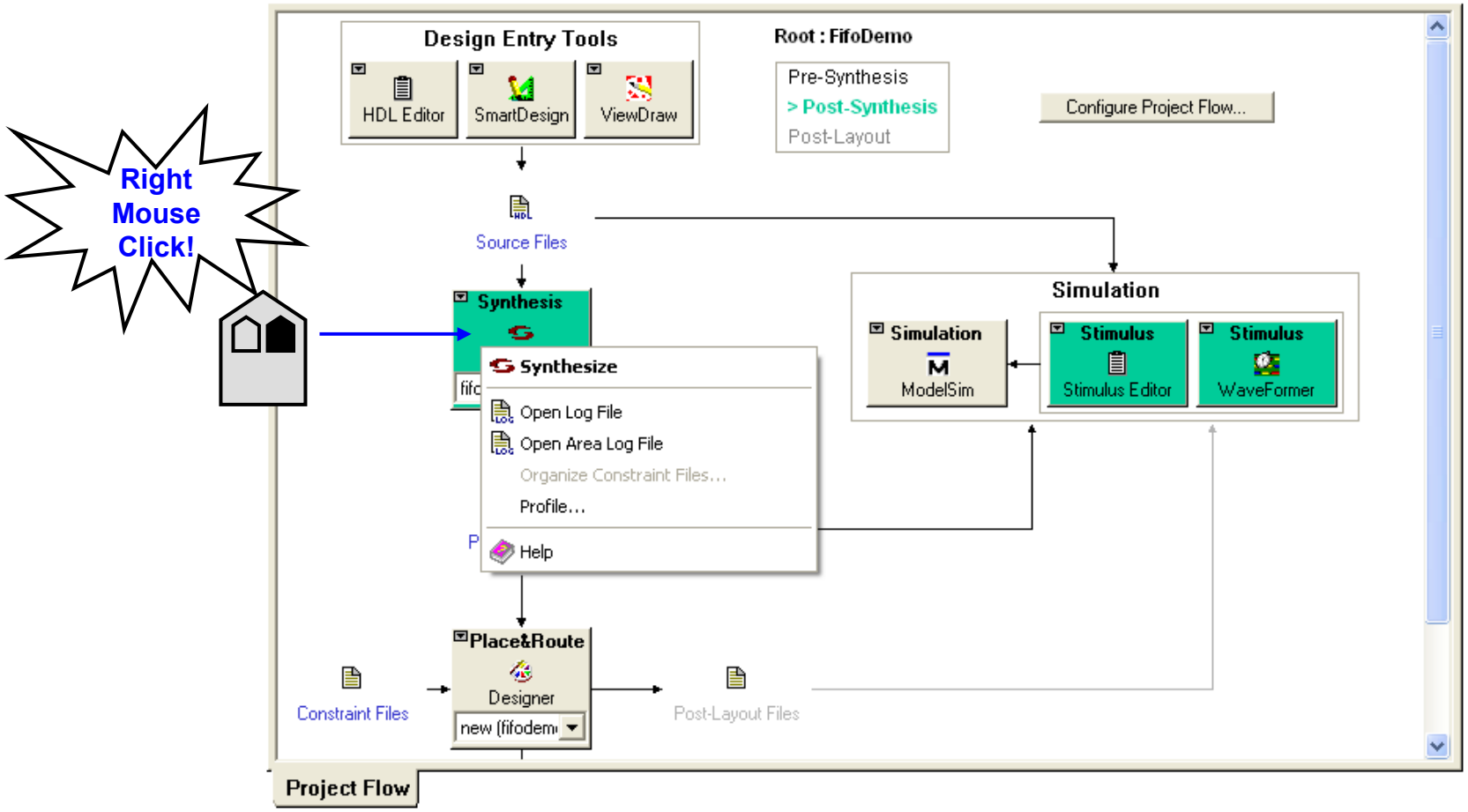
Synplicity Help

- Synplicity Has Complete On-line Manual
  - Invoked from Help Pulldown or by Pressing F1



Opening Synplify Log Files
Libero Project Flow Window

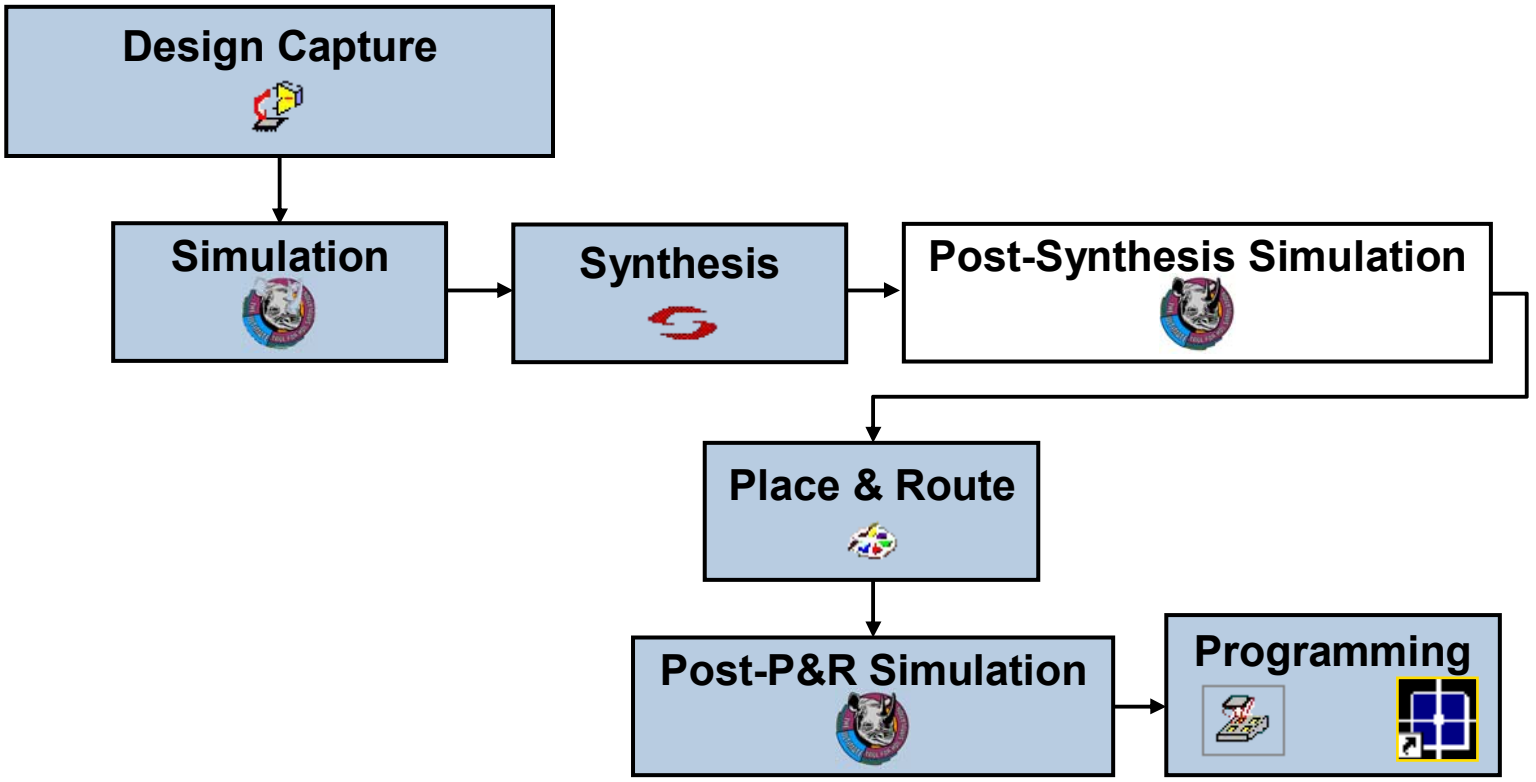
- Synplify Log Files can be Opened from Libero Project Flow Window



Hands-on Lab 4

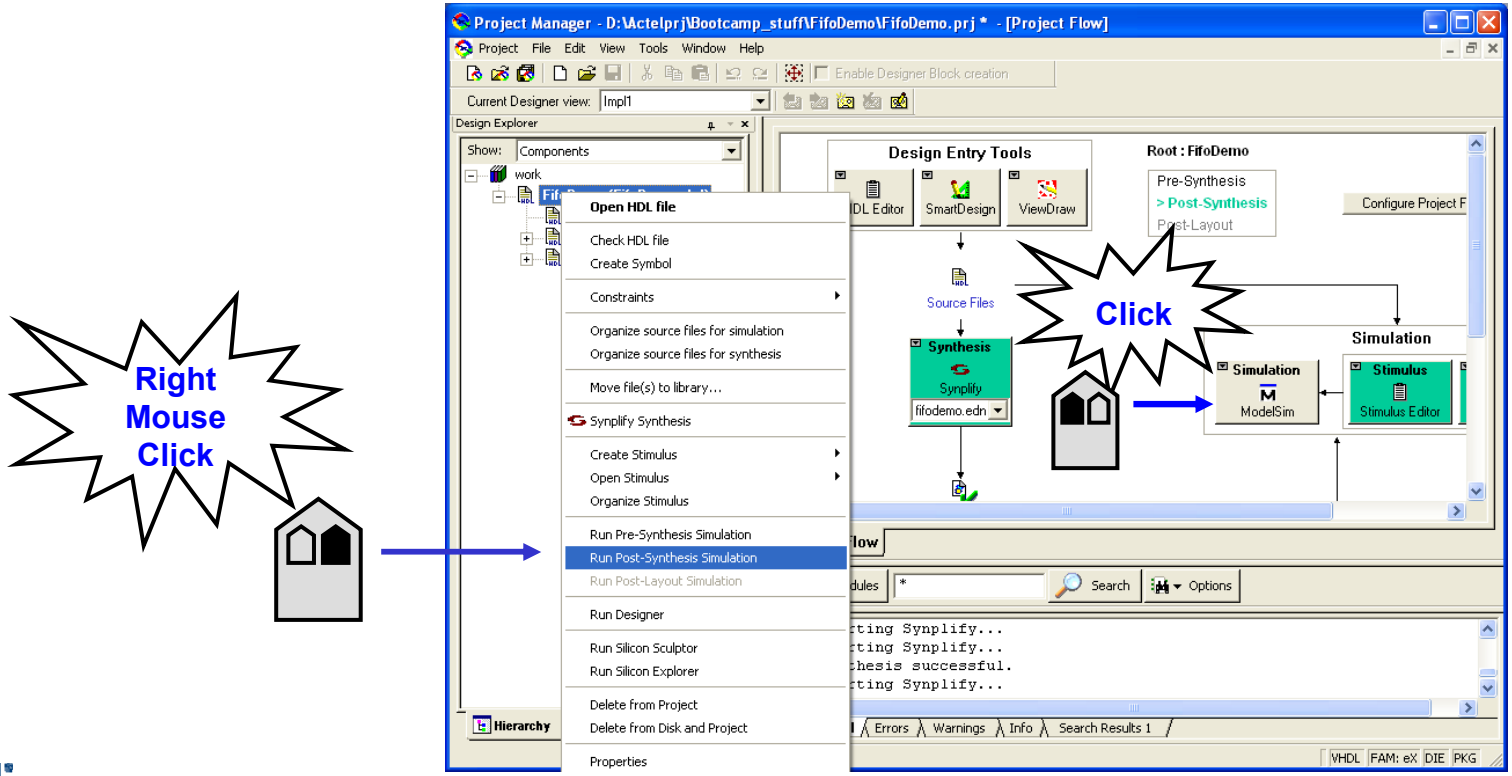
- Complete Lab 4 in the Lab Guide
  - Synthesize the Design with Synplicity
  - Save your Work

Post-Synthesis Simulation



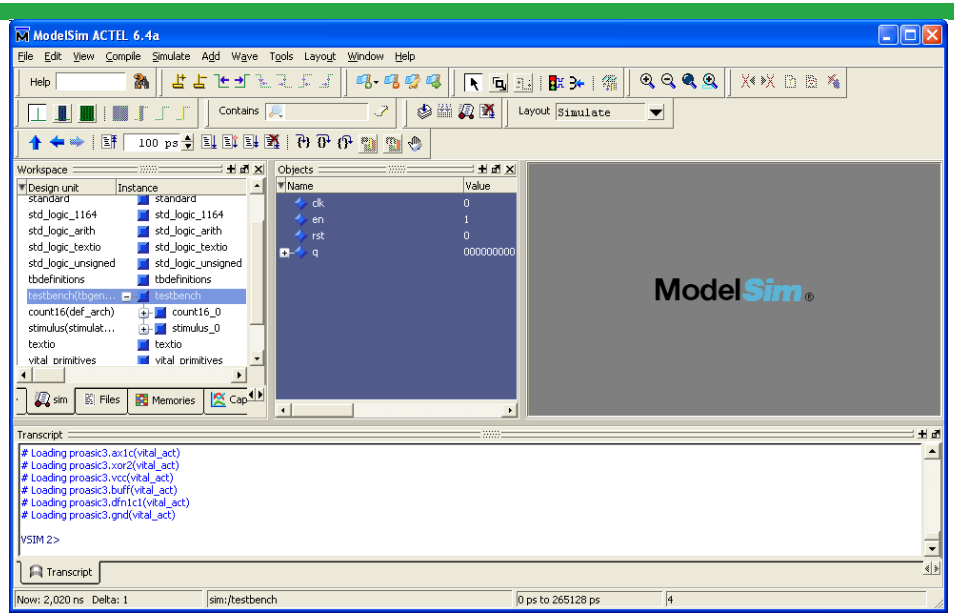
Post-Synthesis Simulation

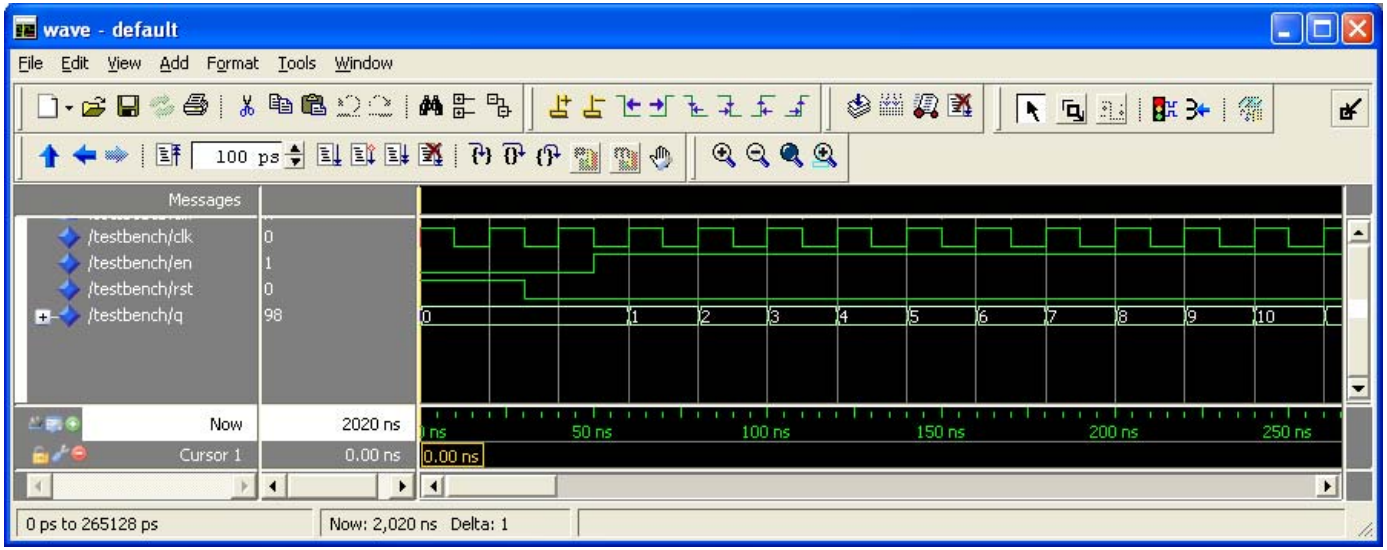
- Steps:
  - Synthesize Design with Synplicity
    - Generate EDIF Netlist from Synplicity
      - Libero Automatically Creates Structural VHDL or Verilog Netlist
  - Run Post-synthesis Simulation on Structural Netlist



Post-Synthesis Simulation

- ModelSim Automatically Compiles Structural Netlist Exported from Designer
  - Default simulation run time: 1 uS
  - Structural library mapping handled by Libero
  - Pre-compiled libraries do not require compiling prior to simulation

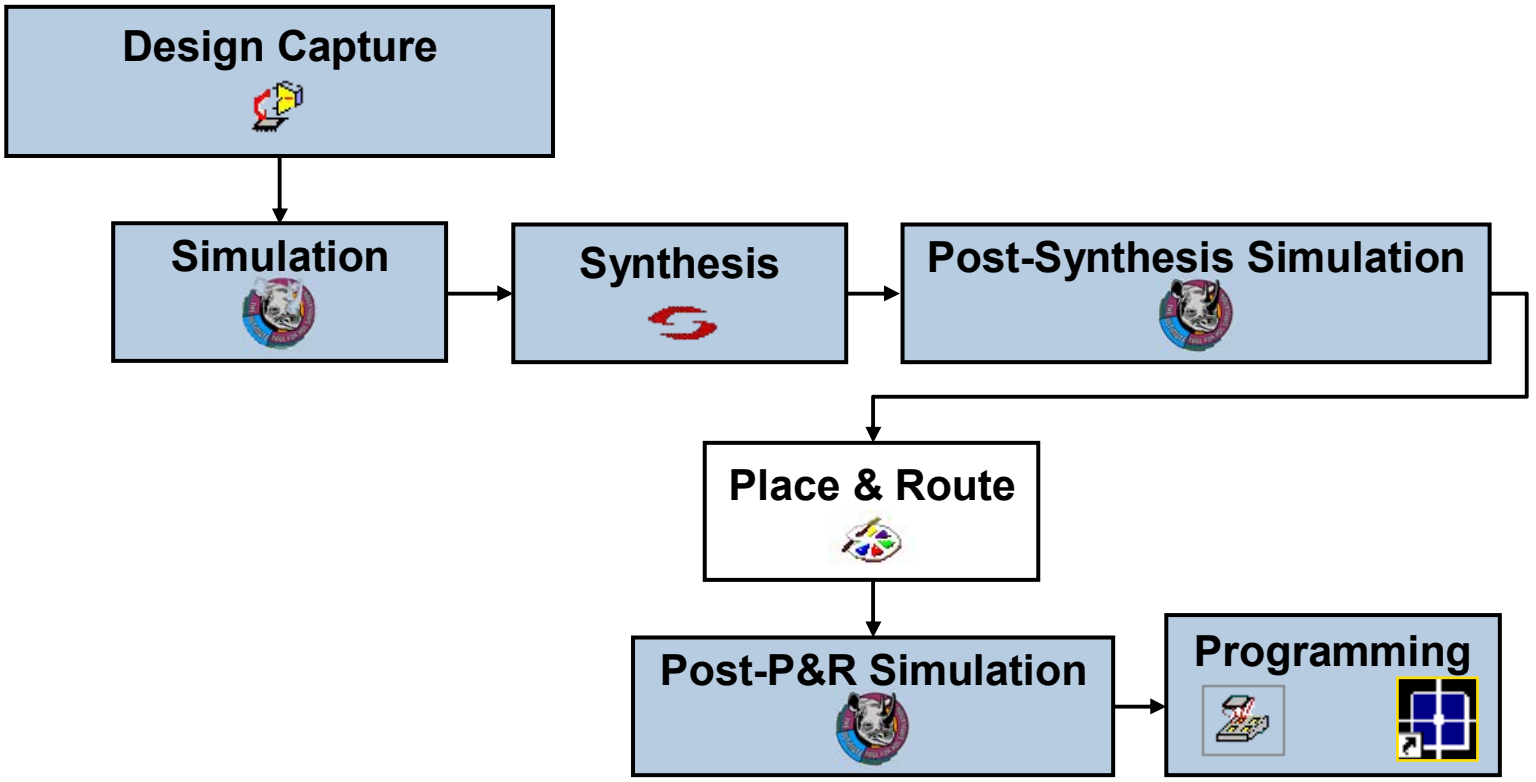




Agenda

- Libero Overview
  - Libero Editions and Content
  - Libero Installation and Licensing
  - Libero Design Flows
- 3rd Party Tools
- Designer Overview
- Programming
- Silicon Explorer
- Reference Material

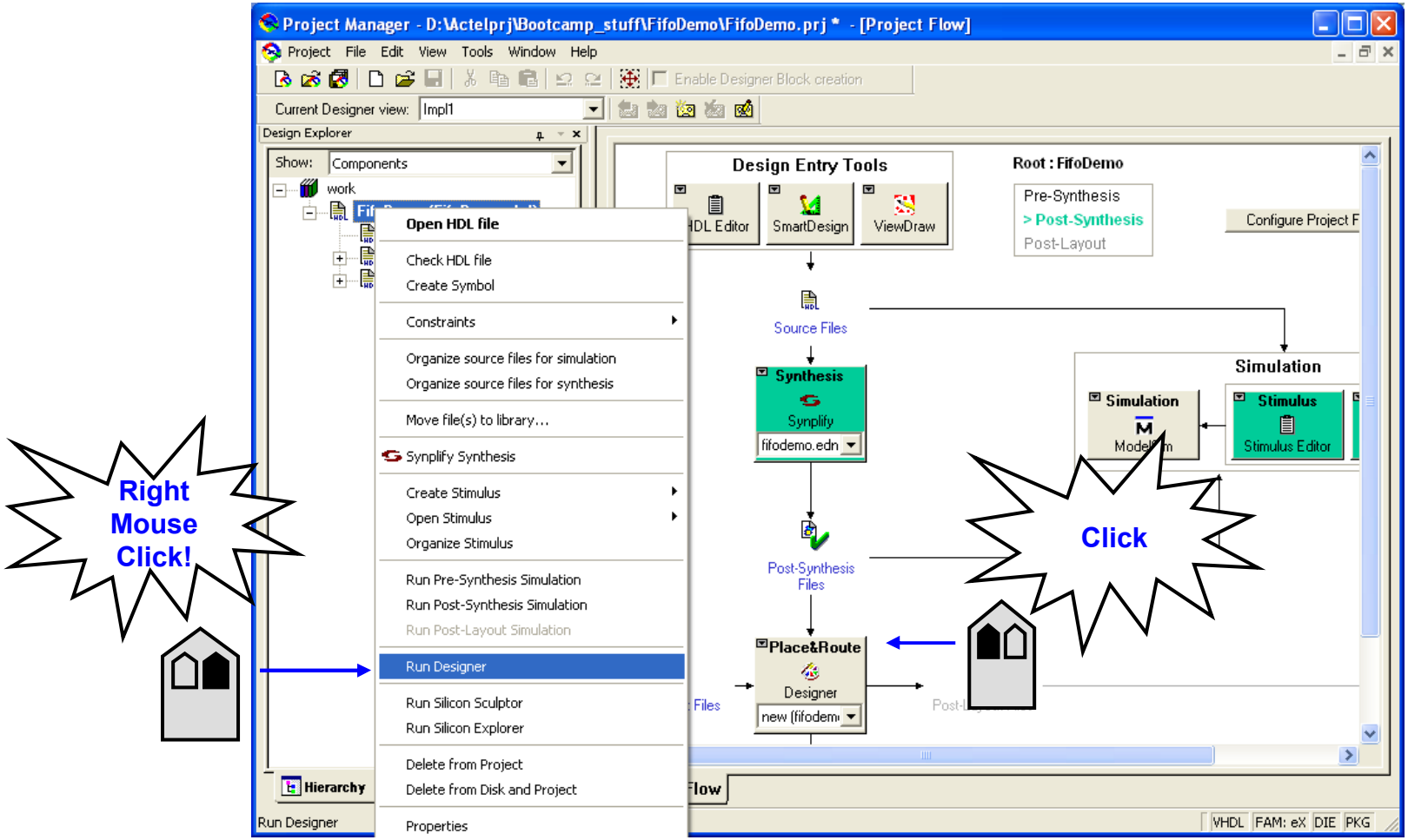
Place and Route



Designer

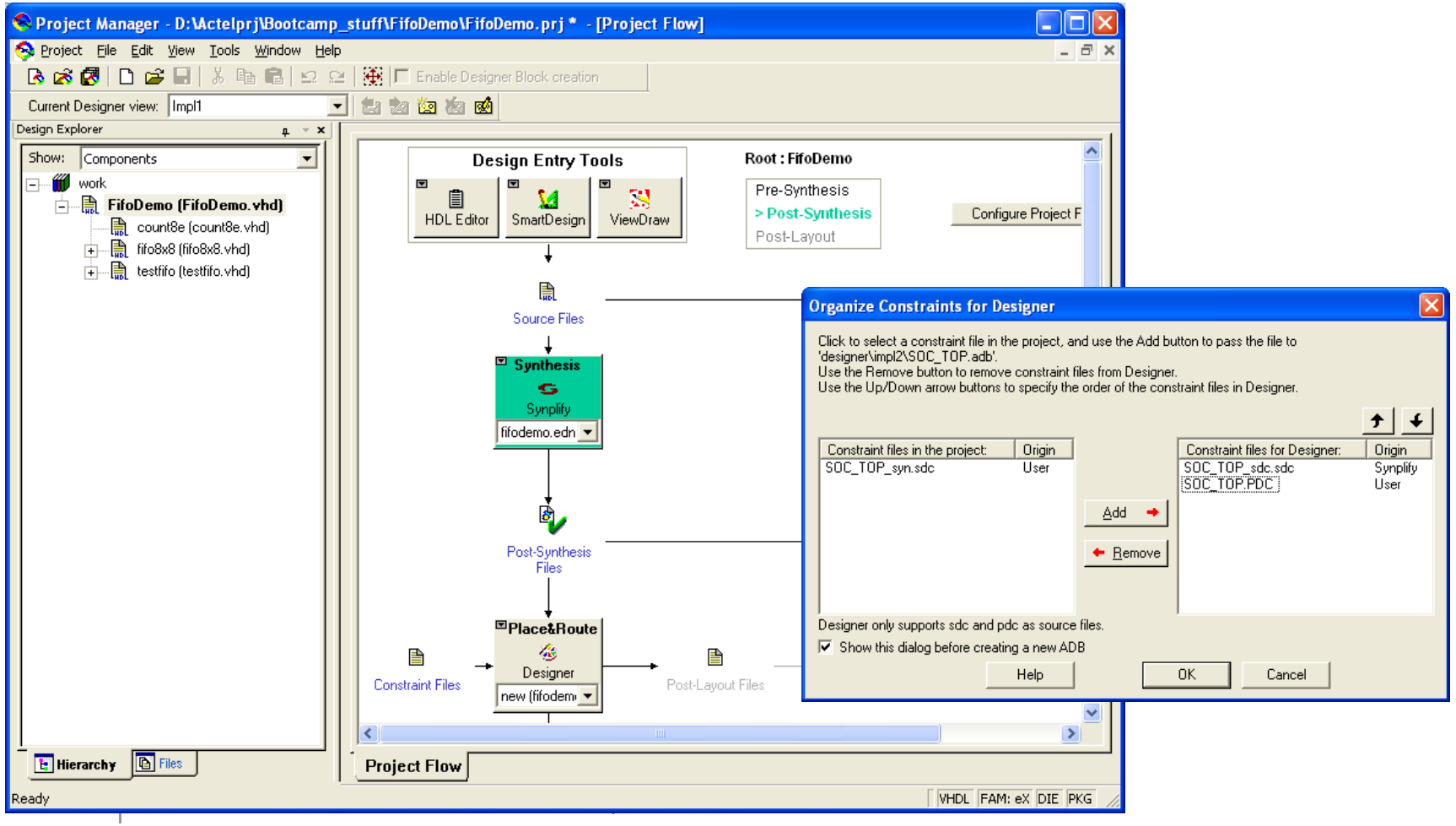
Open Design for Place & Route

- Click on “Place & Route” in Project Flow Window or…

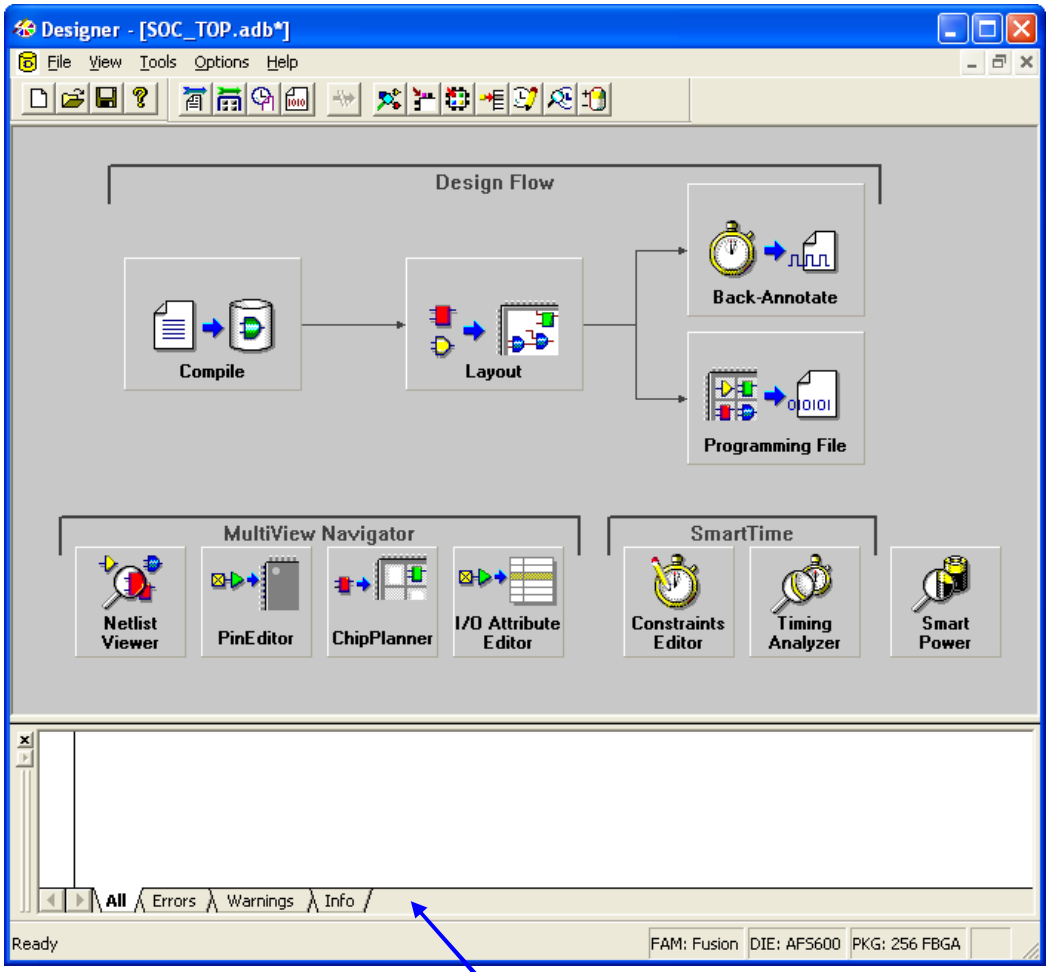


Designer Constraint Files

- Libero Automatically Prompts for Designer Constraint Files
  - Select Timing Constraints and Physical Constraints



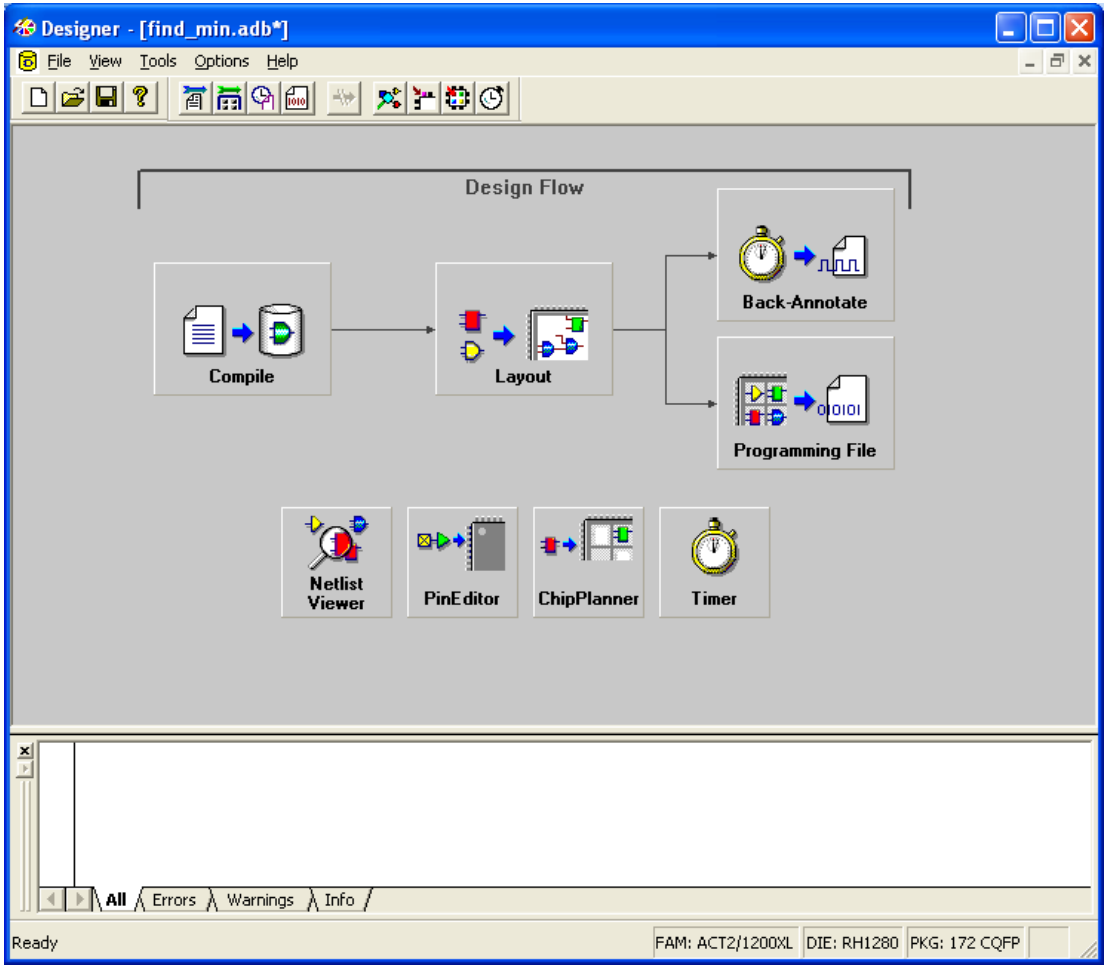
Designer Interface



- Designer Provides Graphical Flow Manager to Lead Designer through Design Flow
- Completed Tasks Highlighted
- Design Flow Steps Listed at Top
- User Tools Grouped Below

Designer Error Manager Tabs (same as Libero)

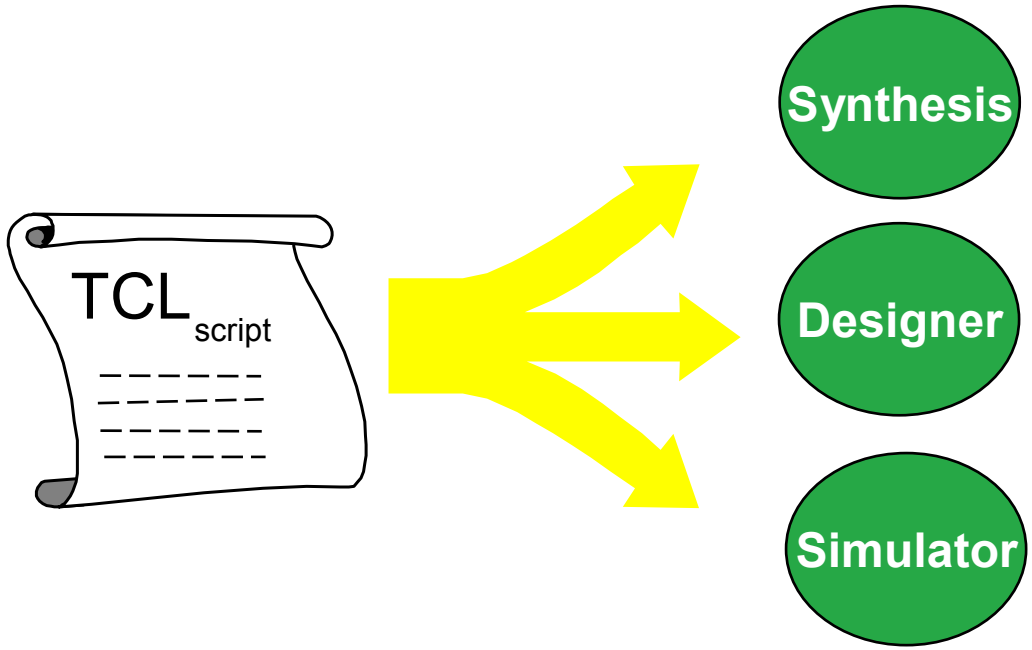
Designer Interface (Legacy Products)



- No MVN, SmartPower or SmartTime
- Timer is used for Timing analysis

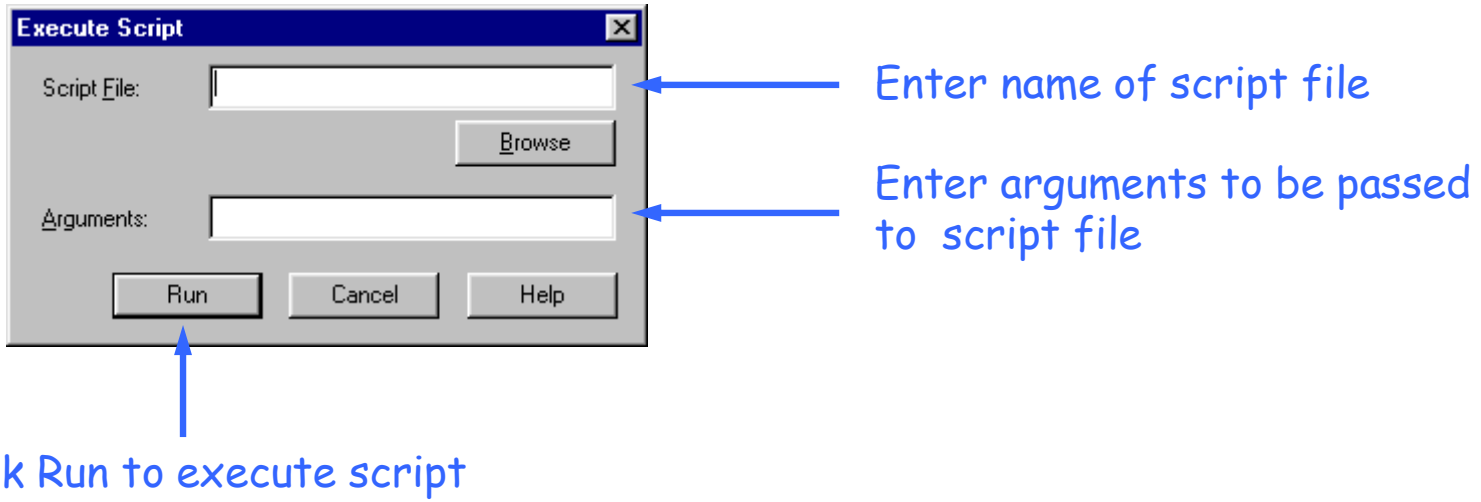
Designer TCL Script Support

- Industry-standard Language
  - Tool Command Language
- Launch Multiple Tools from Single Script
- Launch Multiple Design Runs



Running Scripts within Designer

- In File Menu, Click Execute Script File
  - Displays Execute Script Dialog Box



Tcl Scripts can be Executed from the Command Line:

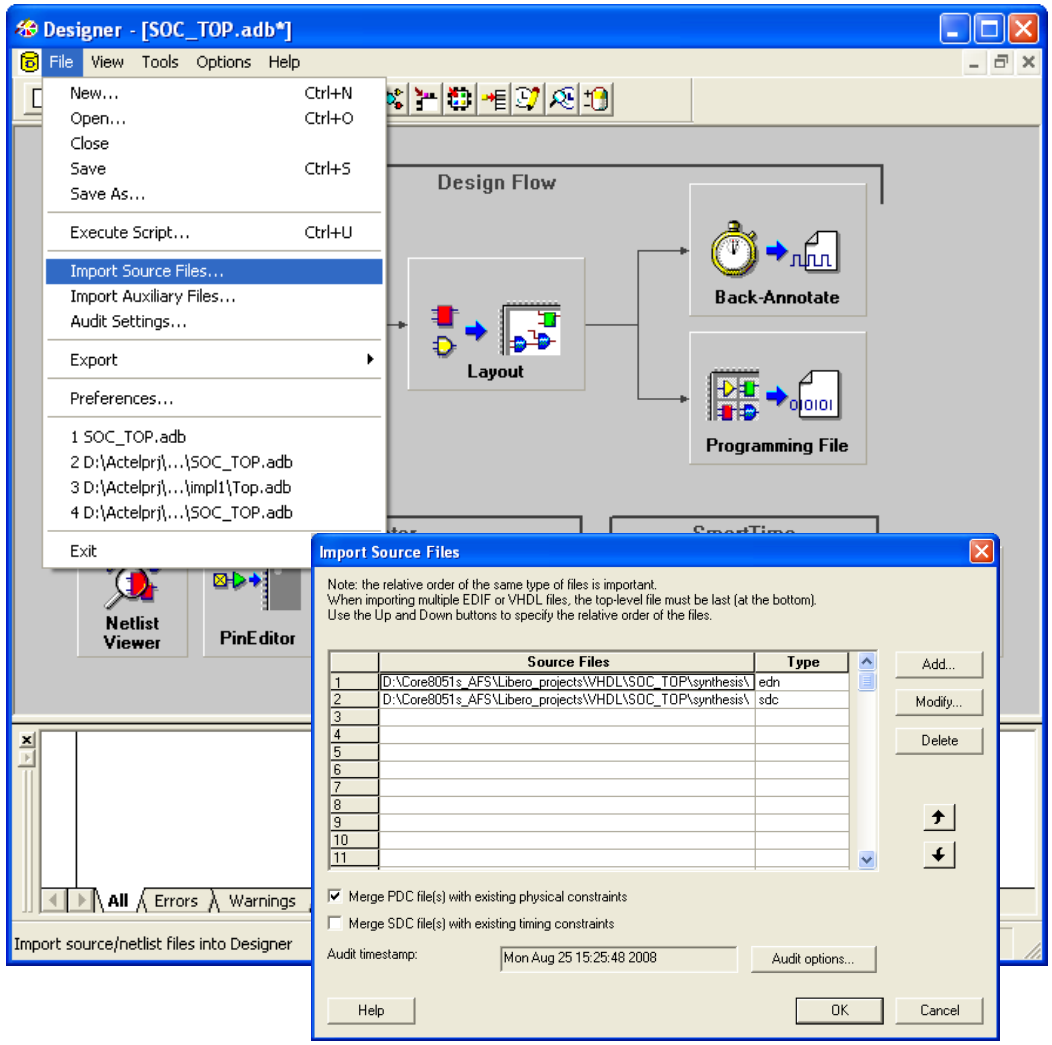
Example:

d:\Libero\Designer\bin\designer script:my_script

Recording Scripts

- Designer Can Export Tcl Script File that Contains Commands Executed in Current Session
- Exported Tcl Script can be used to …
  - … re-Execute Same Commands Interactively or in Batch
  - … Become More Familiar with Tcl Syntax

Importing Source Files



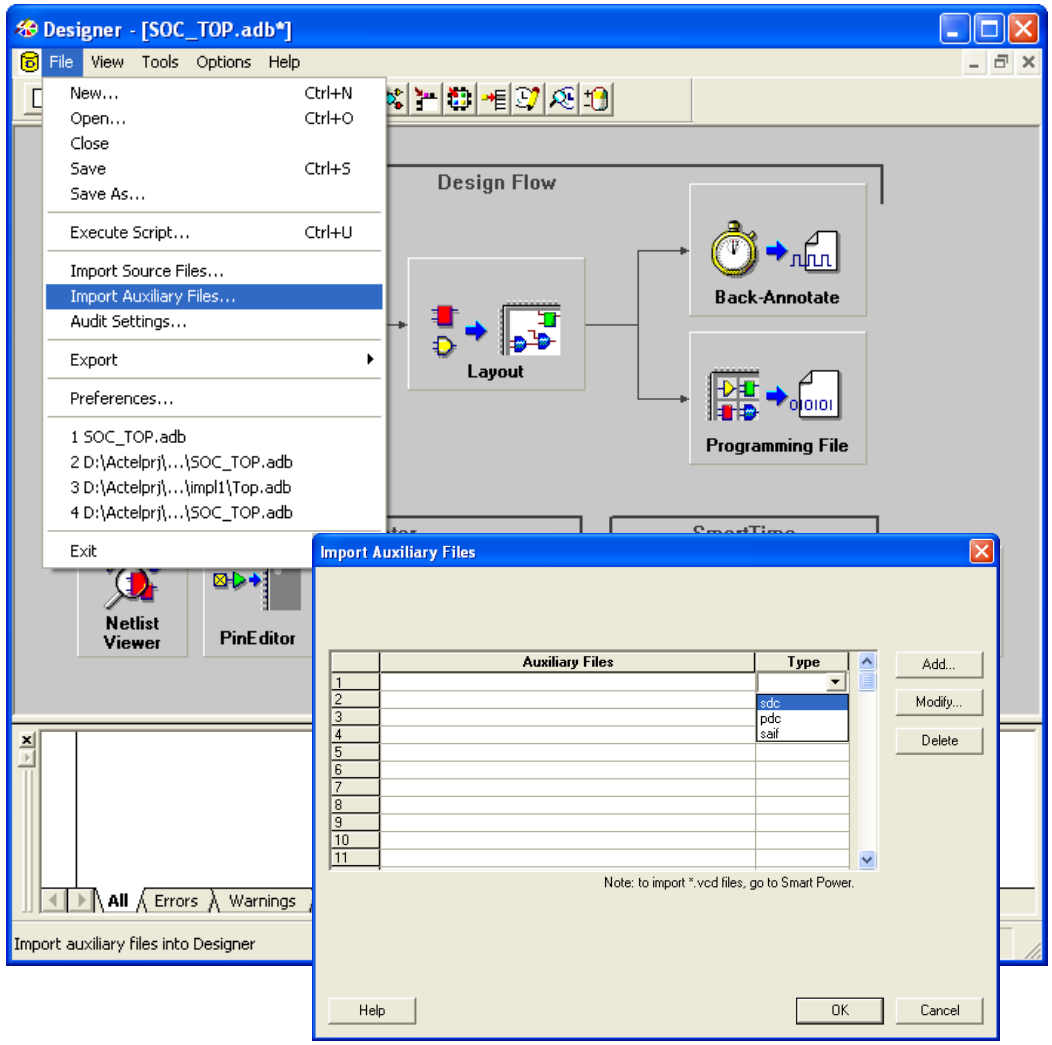
Multiple files can be imported at the same time

Libero sends Netlist and Constraint files to Designer Automatically

Import File types:

File Type	Extension
EDIF	*.ed*
VHDL	*.vhd
Verilog	*.v
Actel ADL Netlist	*.adl
Criticality	*.crt
Timing Constraint File	*.sdc
ProASIC Constraint File	*.gcf
Physical Design Constraint File	*.pdc

Importing Auxiliary Files



Optional step to import pin files, timing constraints, etc.

Import Auxiliary files after compile completes

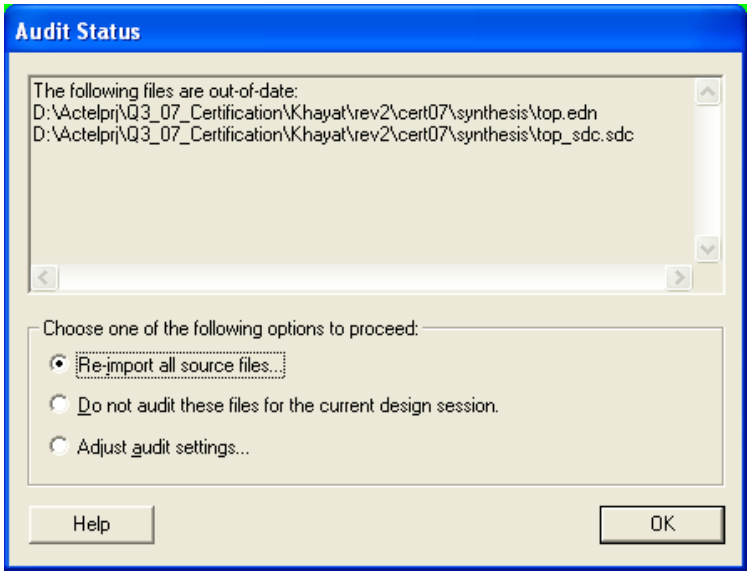
Import File types:

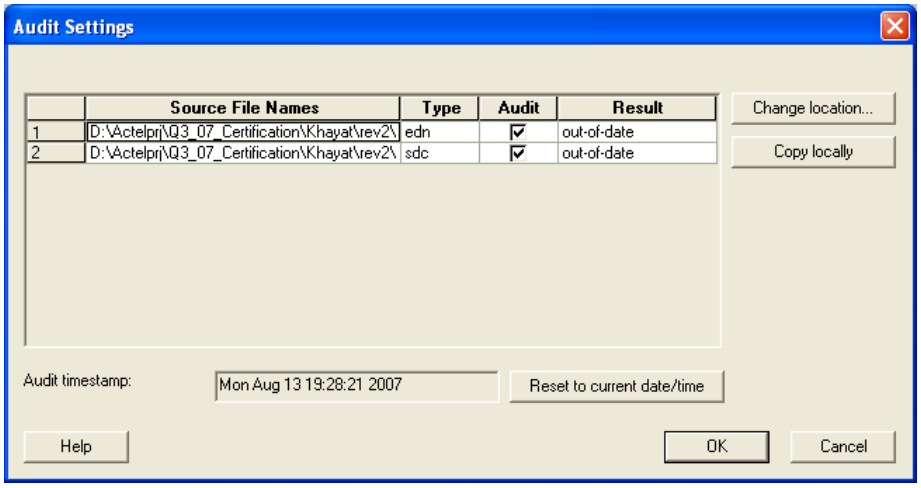
File Type	Extension
Criticality	*.crt
PIN	*.pin
Timing Constraint File	*.sdc
ProASIC Constraint File	*.gcf
Physical Design Constraint File	*.pdc
Value Change Dump	*.vcd
Switching Activity Intermediate File	*.saif
Design Constraint File	*.dcf

Entering Constraints in Designer

- Option 1 - Import Files in Designer
  - Source Files are Passed by Libero Automatically
- Option 2 - Import Files in TCL Script
- Option 3 - Set All Constraints Directly in Designer
  - Physical – PinEdit or IO Attribute Editor
  - Timing – SmartTime or Timer

Designer File Auditing



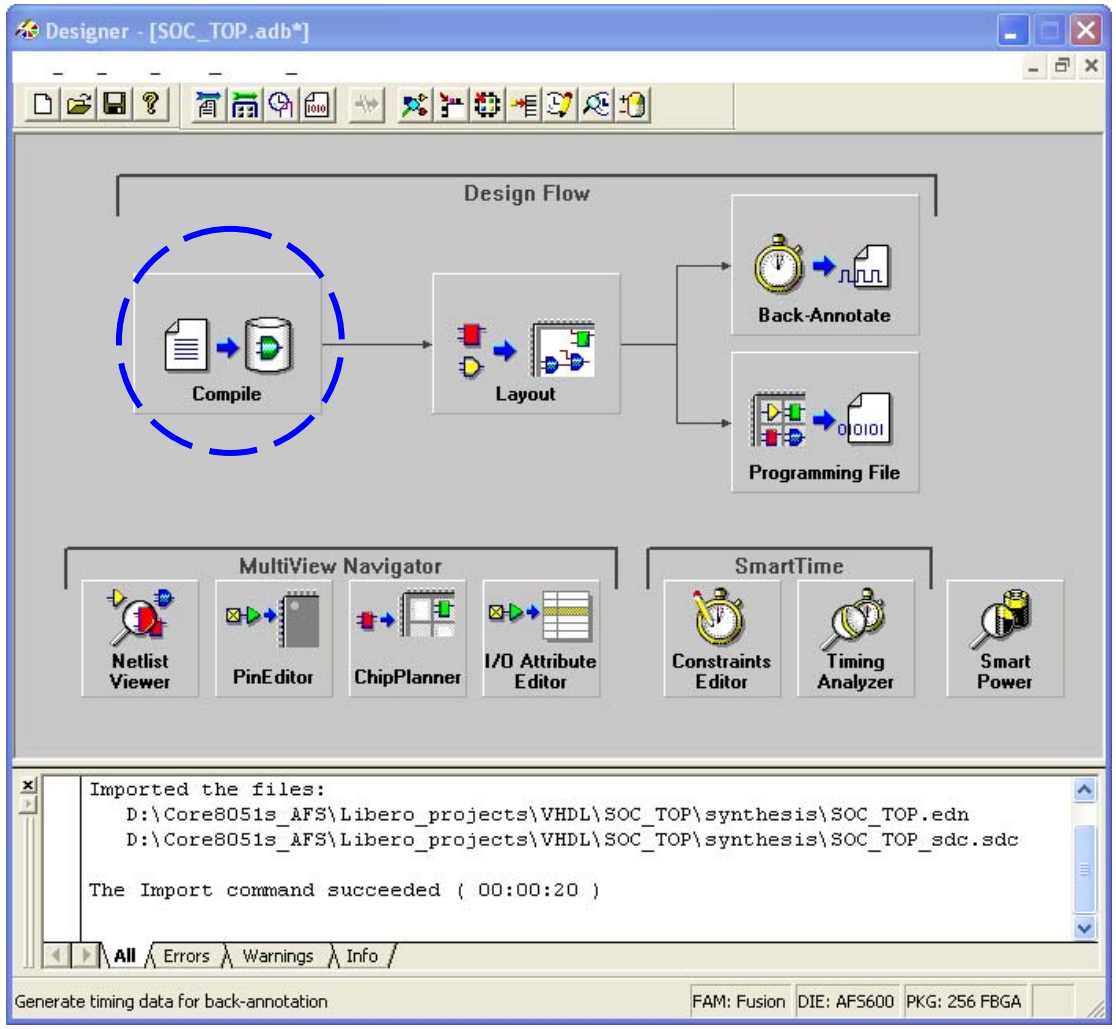


- Designer Source File Auditing Ensures Imported Files Are Current
  - All Imported Source Files Are Date- and Time-stamped
  - Designer Notifies You if File Is Changed
- Audit Settings Can Be Changed (File > Audit Settings)
  - Enable / Disable Auditing
  - Move File to New Location
  - Associate File with Current Date and Time

Importing Files into Designer - Summary

- Import the Following Source Files
  - EDIF, VHDL, Verilog Netlists
  - PDC, GCF and SDC Files
  - Source Files Are Audited per User Settings
- Import the Following Auxiliary Files
  - DCF, SDC, PDC, GCF, VCD, and SAIF Files

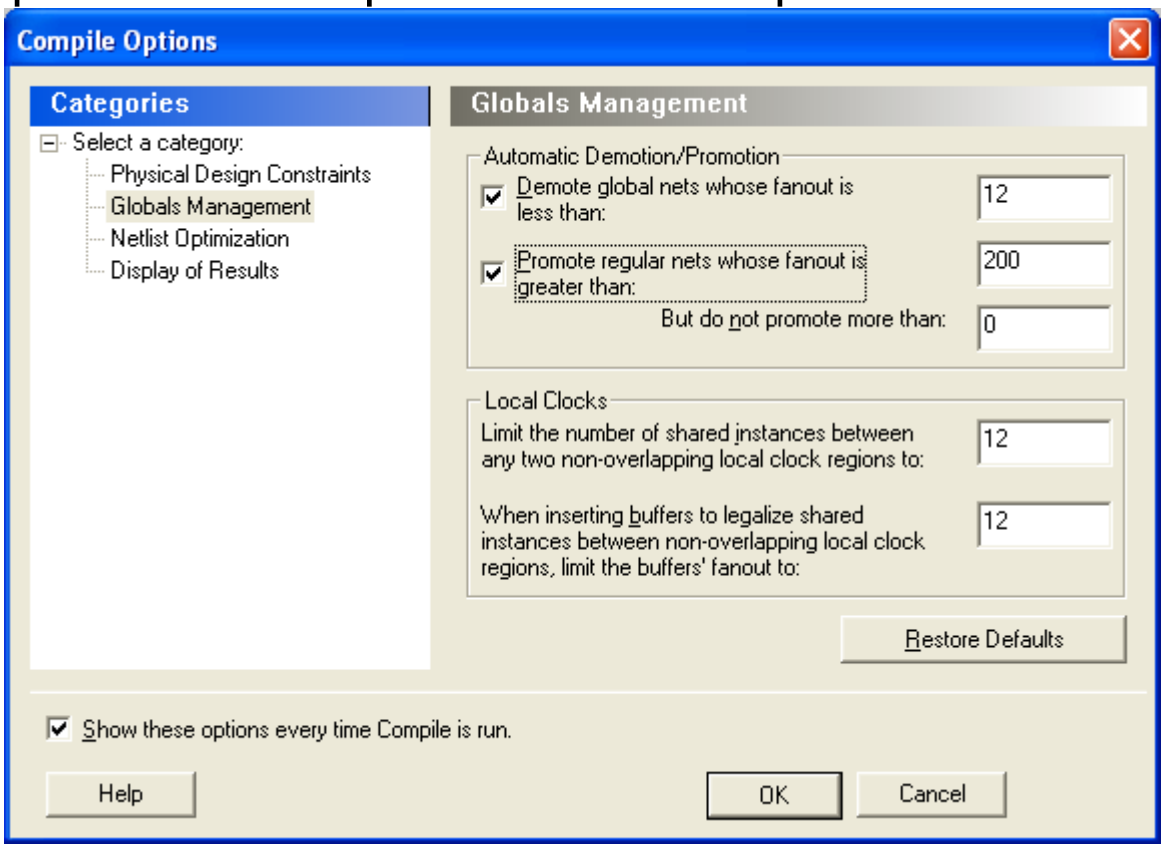
Designer Compile



- Reads Netlist
- Compiles Design into Actel Database (ADB) File
- Runs Combiner
- Performs Design Rule Checking
- Checks for Netlist Errors (Bad Connections and Fanout Problems)
- Removes Unused Logic (gobble)
- Verifies that Design fits into Selected Device

ProASIC3 Compile Options
Globals Management

- Automatically Promote Nets to Globals
- Automatically Demote Global Nets to Regular Routing Resources
  - Each option has an equivalent Tcl Compile command

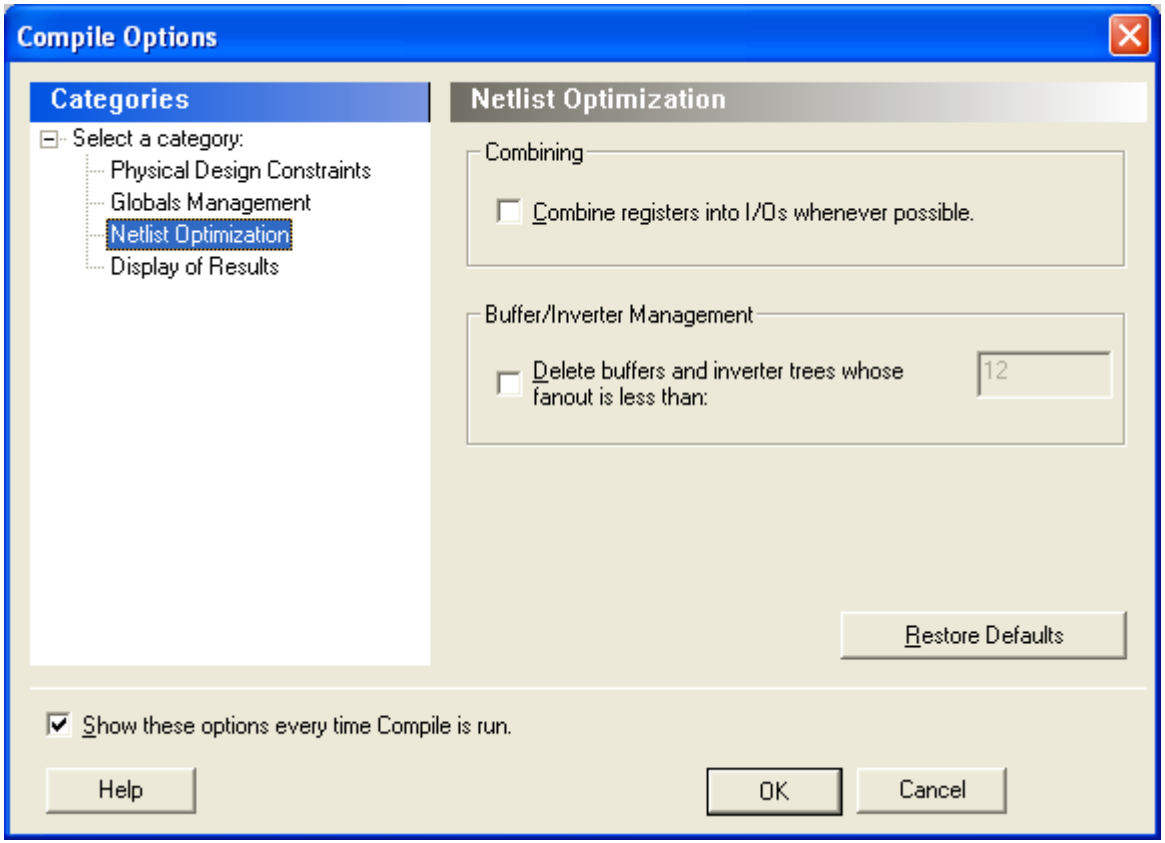


Constraint File Types by Family

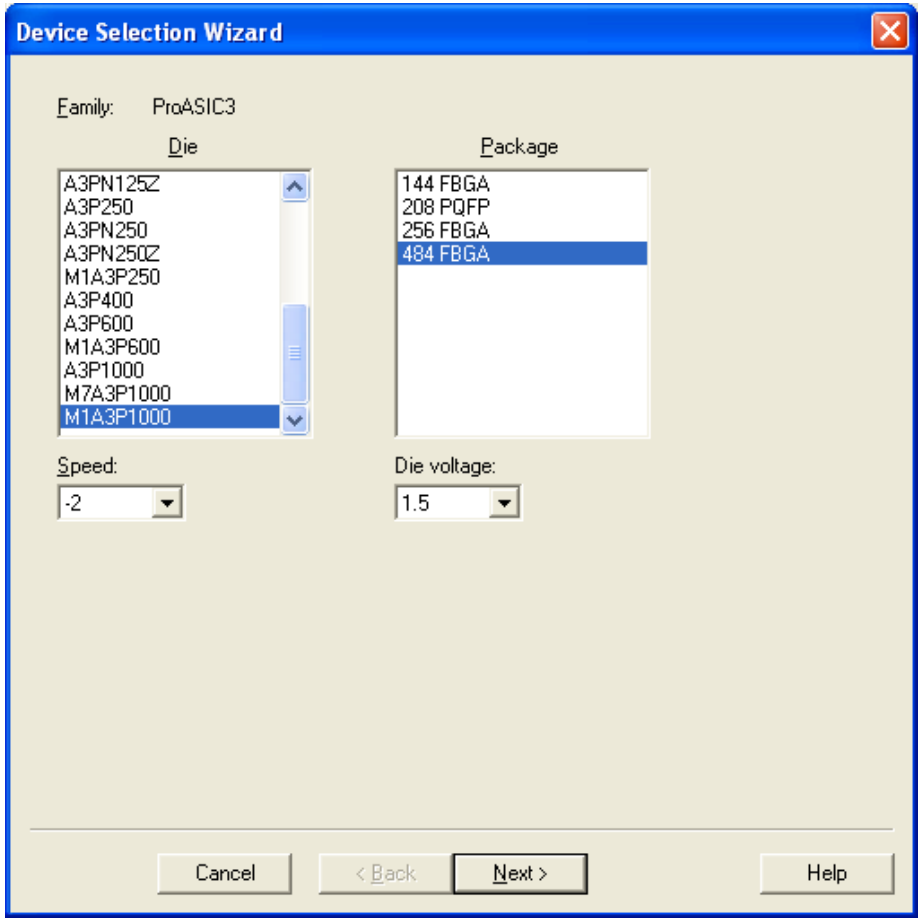
Family	Timing		Physical Placement			Netlist Optimiization	
	SDC	DCF	PDC	PIN	GCF	PDC	GCF
IGLOO	X		X				
Fusion	X		X			X	
ProASIC3	X		X				
ProASIC PLUS	X				X		X
ProASIC	X				X		X
Axcelerator	X		X			X	
eX	X	X		X			
SX-A	X	X		X			
SX		X		X			
MX		X		X			
DX		X		X			
ACT3		X		X			
ACT2/1200XL		X		X			
ACT1		X		X			

ProASIC3 Compile Options
Netlist Optimization

- Perform Netlist Optimization
  - Use I/O Registers
  - Delete Buffers and Inverters

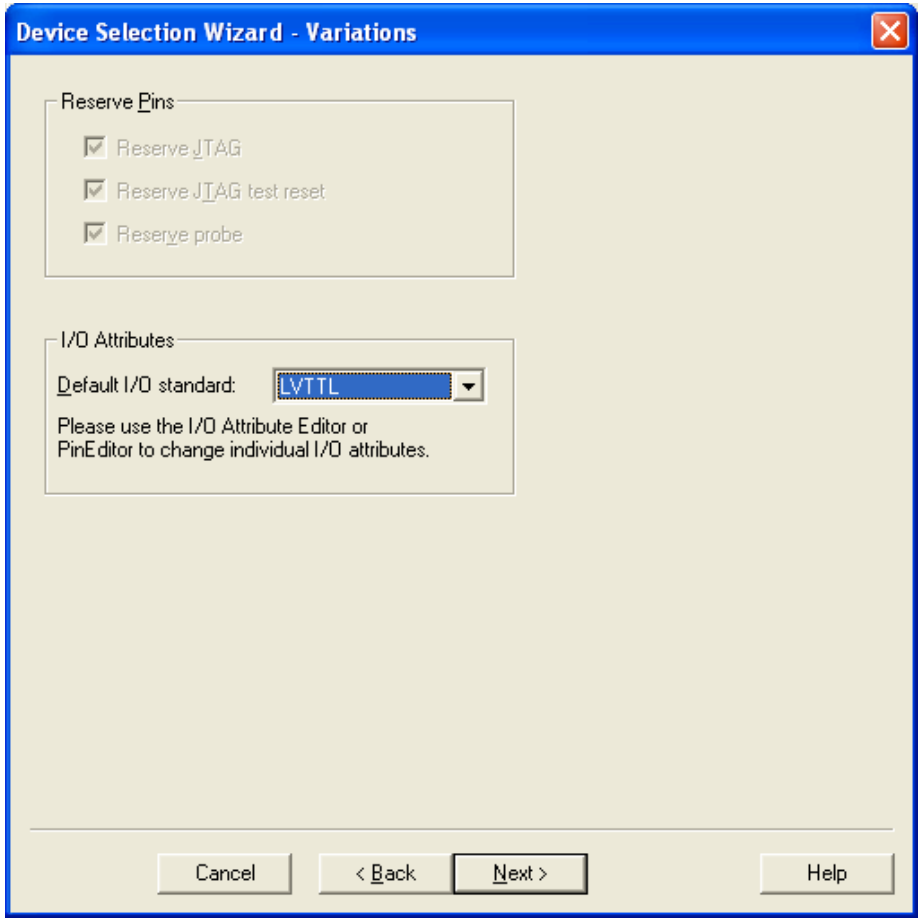


Compile Wizard - Die and Package



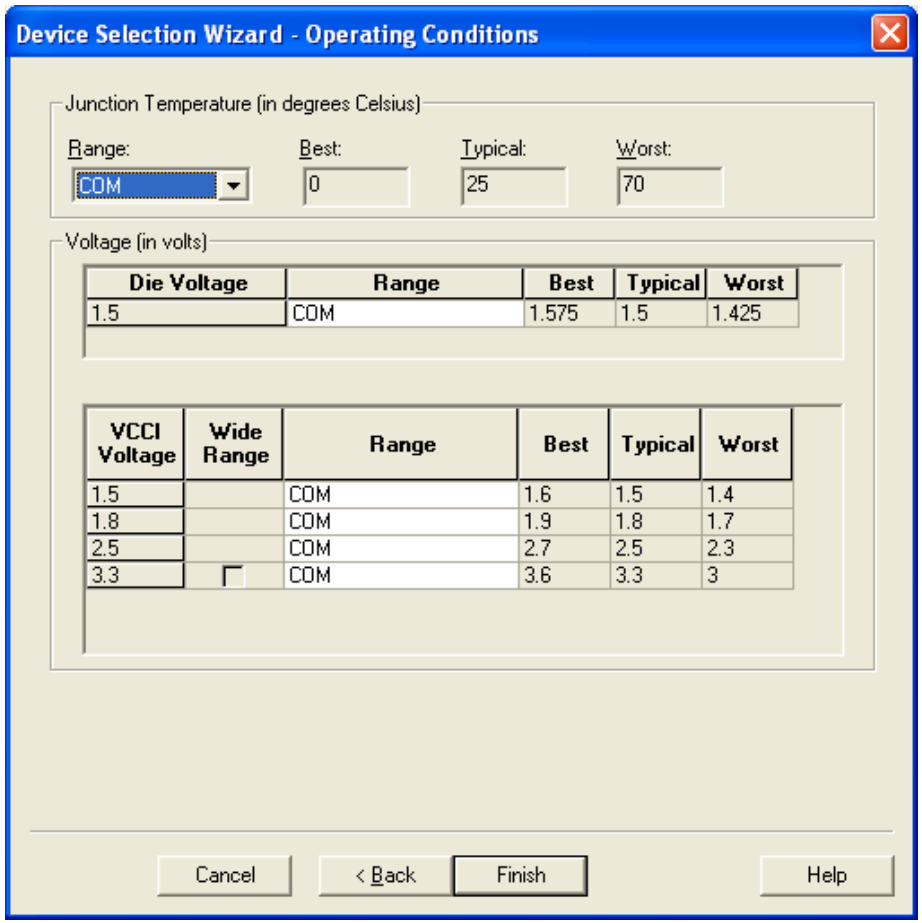
- Select:
  - Die
  - Package
  - Speed Grade
  - Die Voltage

Compile Wizard - I/O Attributes



- Restrict Pin Usage
  - Reserve JTAG Pins
  - Reserve ActionProbe Pins
- Select Default I/O Standard

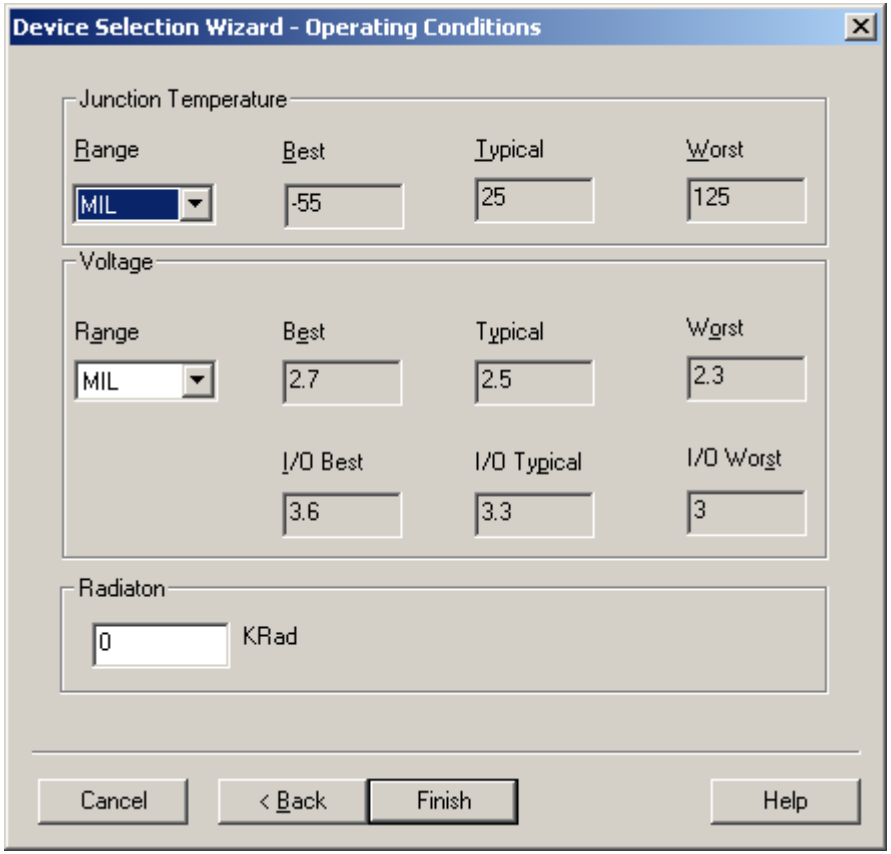
Compile Wizard - Operating Conditions



- Select Junction Temperature
  - Commercial (0 - 70ºC)
  - Industrial (-40 - 85ºC)
  - Custom
  - Military (-55 - 125ºC)
    - Mil Devices only
- Select Voltage Range
  - Commercial
  - Industrial
  - Custom
  - Military
    - Mil Devices only
  - Die and I/O Voltage Selections

Operating Conditions
Radiation Derating

- Radiation Derating for AX-S and SX-S
  - Add Radiation Exposure Level for Radiation Derated Timing



Designer Netlist Optimization

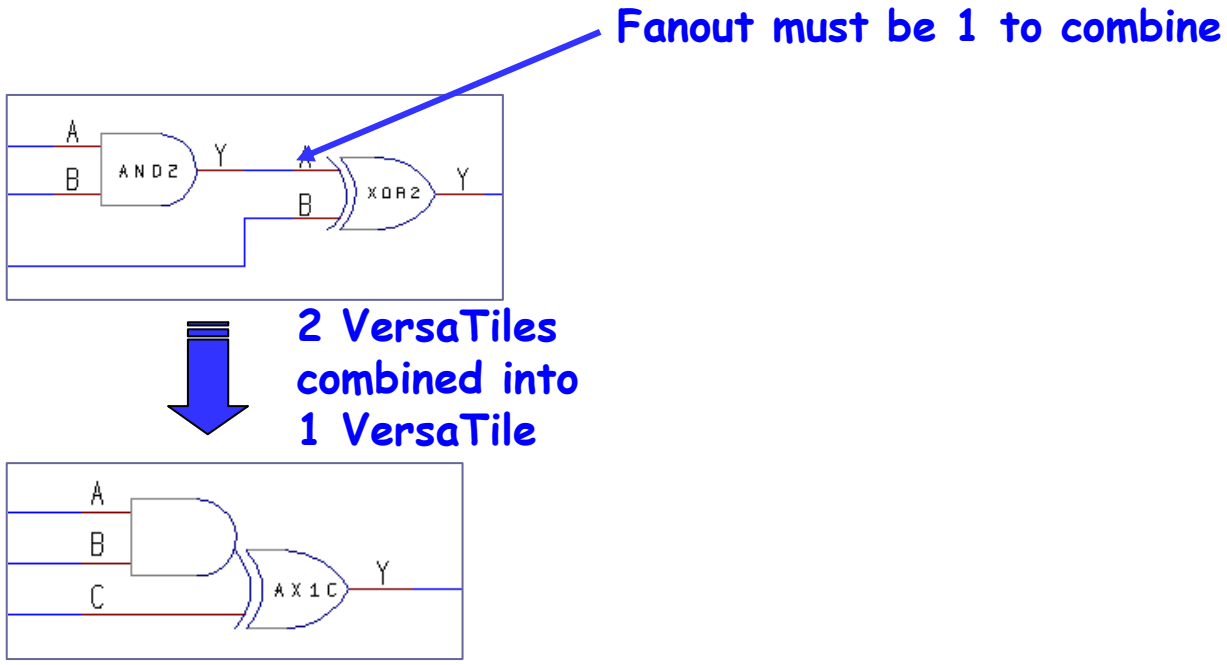
Netlist Optimization

ProASIC3, Fusion, IGLOO and ProASIC^PLUS

- Combiner can Remove all Buffers and Inverters from Netlist that Have No Effect on Circuit's Functional Behavior
  - Reduces Overall Size of Design
  - Produces Faster Place and Route Times
  - Takes Advantage of Inverted Inputs of Logic Tiles
- Original Netlist Preserved
  - Removed Cells Back-annotated with 0ns Delay in SmartTime

ProASIC3, Fusion, IGLOO - Logic Combining

- Combiner Can Combine VersaTiles Configured as Combinatorial Cells
  - Not Supported for ProASICPLUS



ProASIC3/E, Fusion, IGLOO/e

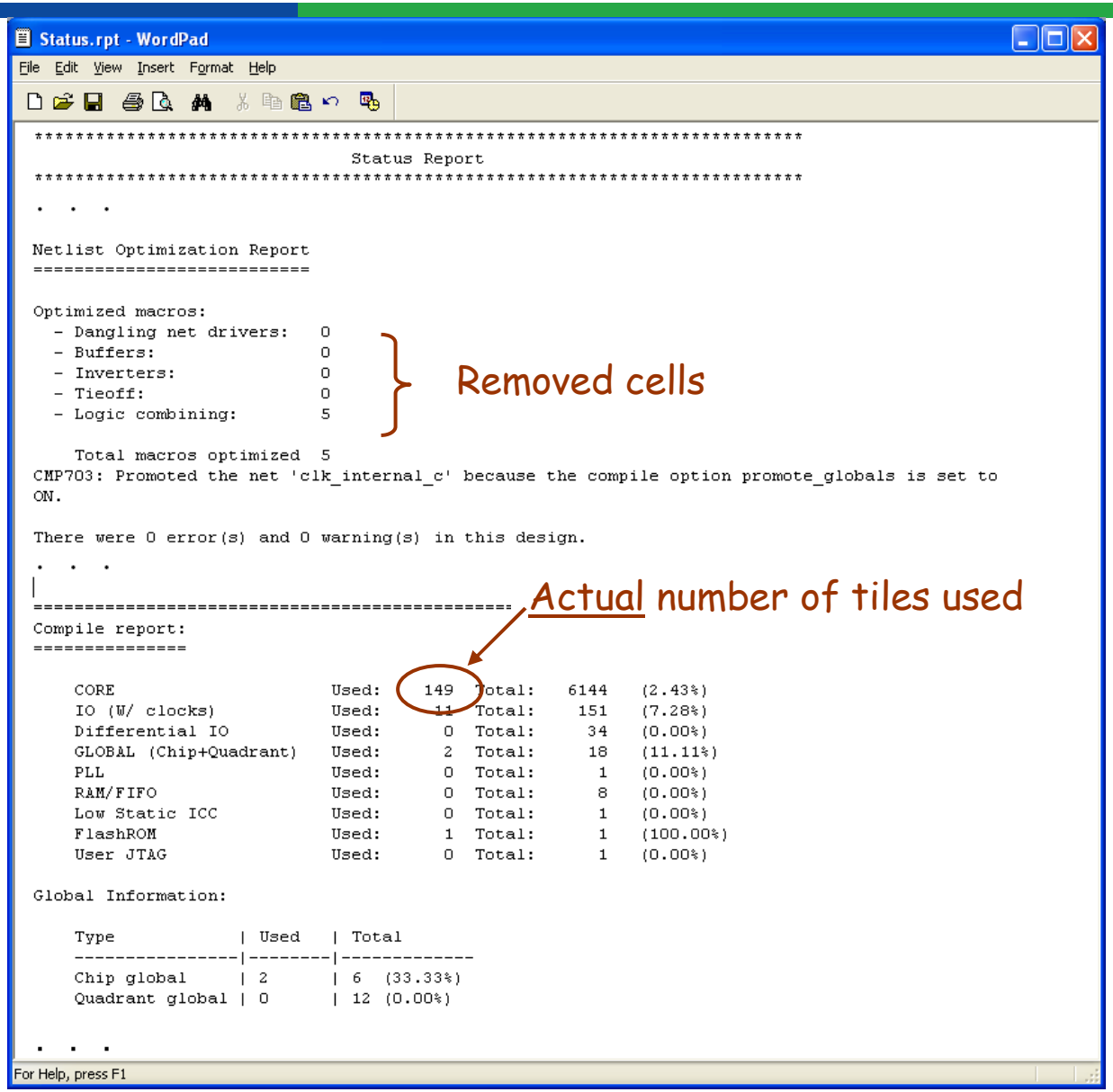
Netlist Optimization Constraint Options

- Delete Buffer Tree
  - Instructs the Compile command to remove all buffers and inverters from a given net
- Syntax:

  `delete_buffer_tree [`*`netname`*`]`

Designer Status Report

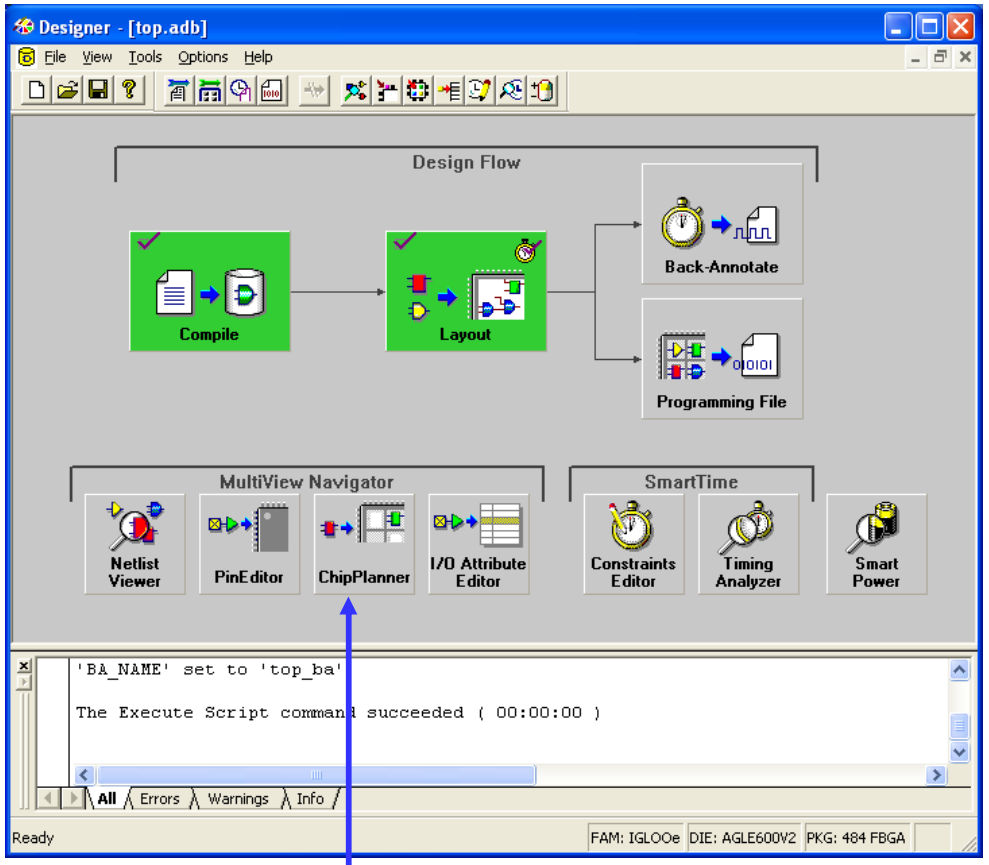
ProASIC3/E, Fusion, IGLOO/e Designs



MultiView Navigator

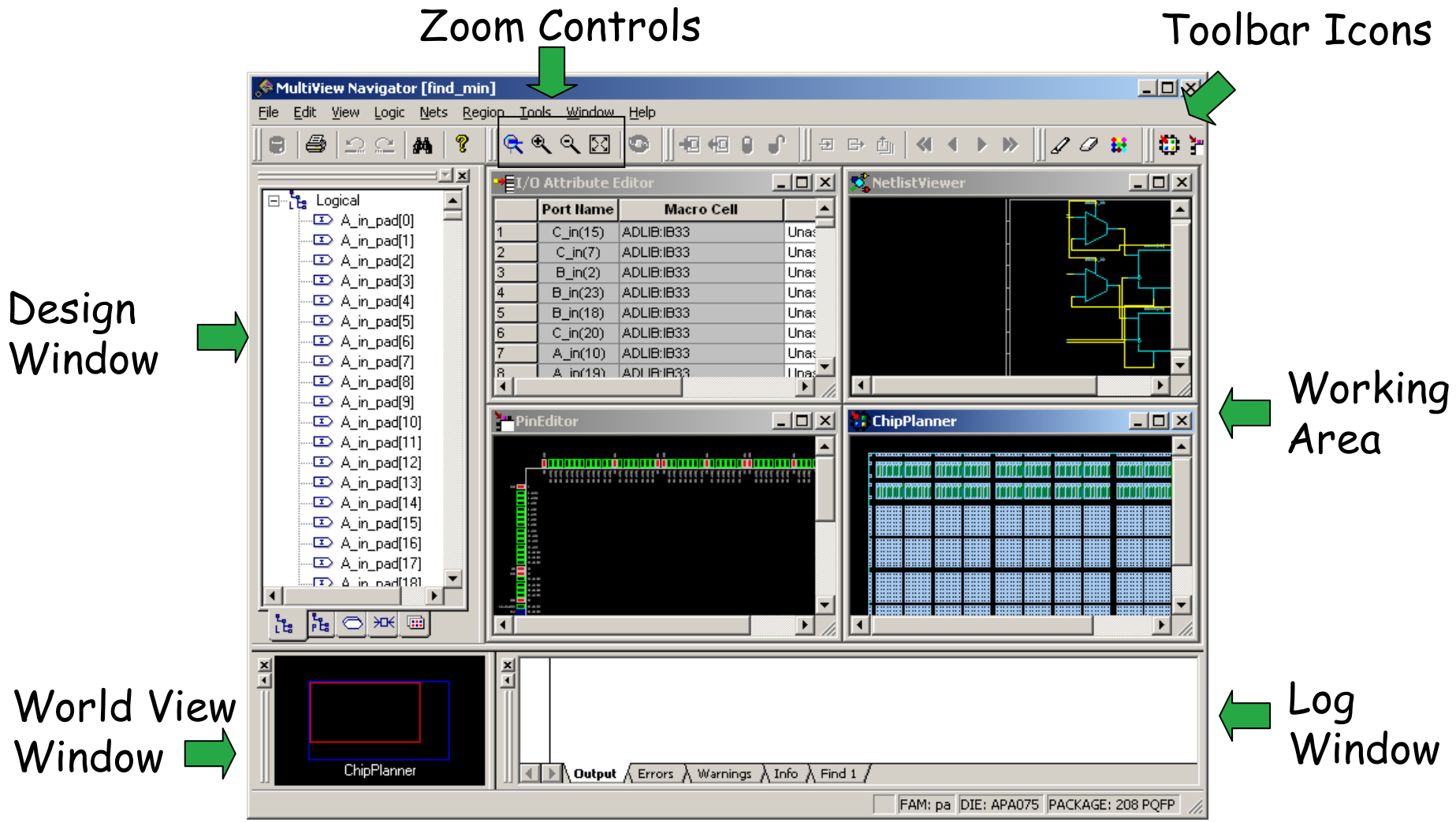
MultiView Navigator

- MultiView Navigator Includes the Following Tools:
  - PinEditor, I/O Attribute Editor, NetlistViewer, and ChipPlanner
- Supported for the following families:
  - IGLOO/e, Fusion, ProASIC3/E, ProASIC3L
- Supports Cross-probing Among Different Designer Tools



Click one of these buttons to open MVN

MultiView Navigator

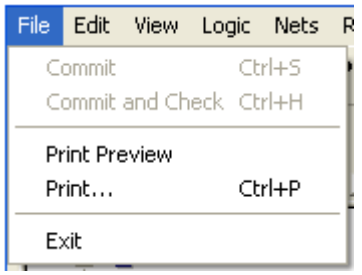


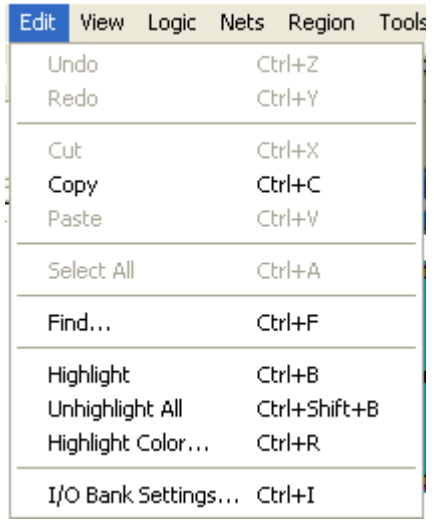
MultiView Navigator
Windows

- Design Window
  - View Design as Logical Blocks, Physical Elements, Ports, Nets, and Regions
- World View Window
  - Shows Position of Current Viewing Window Relative to Chip
- Working Area Shows Current Active Tools
  - Tile or Cascade Active Tools
- Log Window Keeps Running Log of Activity
  - Output – Shows All Messages
  - Errors – Shows Error Messages
  - Warnings – Shows Warning Messages
  - Info – Shows Informational Messages
  - Find Window – Keeps Result of Find Function for Later Usage

MultiView Navigator Toolbar

- File ->
  - Commit – Writes any Changes Made in Editors to Design
  - Prelayout Check – Verifies Placement Changes in Editors Are Legal
- Edit ->
  - Undo/Redo – Allows User to Undo a Mistake or Redo an Accidental Undo
  - Find – Enables Find Interface
- Zoom Controls
  - Zoom Region, Zoom In, Zoom Out, Zoom to Fit
- Assignment Options
  - Place, Unplace, Lock (Fix), Unlock (Unfix)
- Tools Menu
  - ChipPlanner, PinEditor, NetlistViewer, I/O Attribute Editor
- “1 click" commands
  - Commit and Check
  - Lock All
  - Unlock All
  - Un-assign All from Location
  - Un-assign All from Region





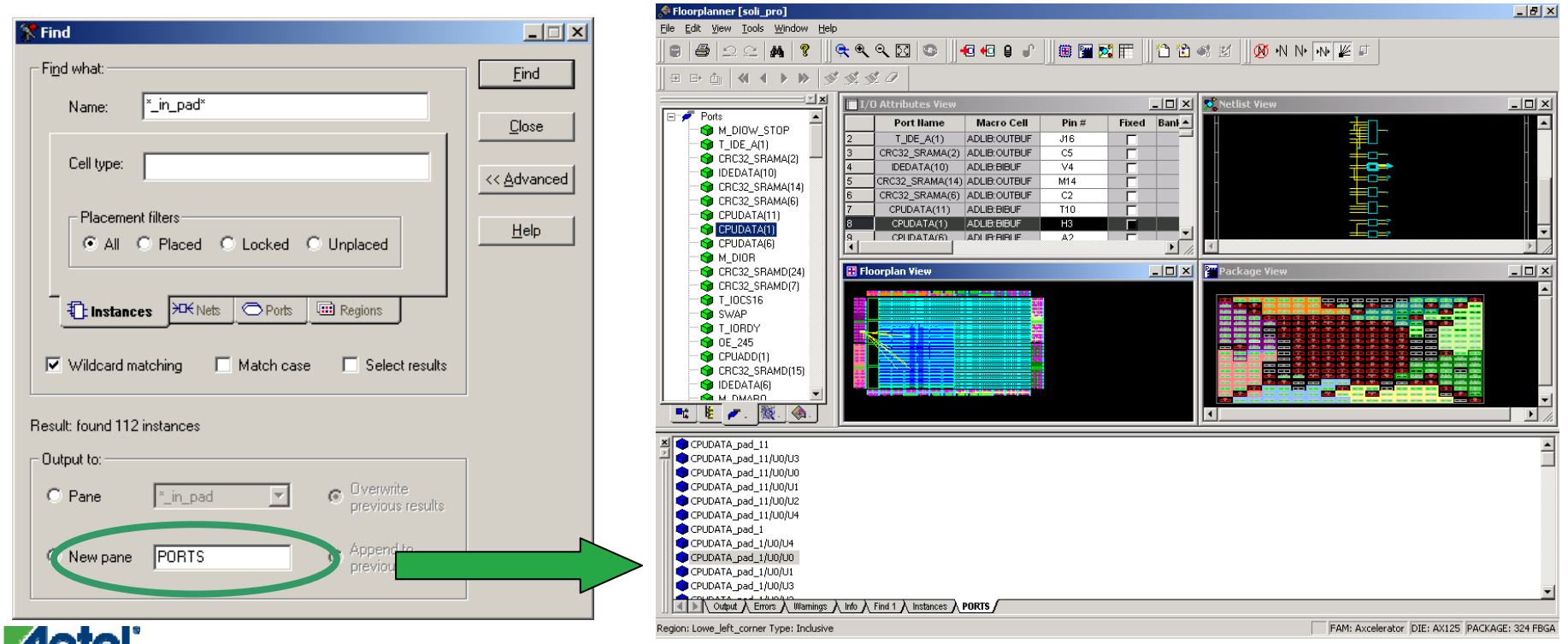
MultiView Navigator
Prelayout Checker

- Infeasible Constraints Identified pre-Layout
  - Overlapping Region Checks
  - Resource Overbooking
  - I/O Technology Checks

- Constraint Checker Runs Automatically when You Commit from MVN
  - Users Can Run DRC Checker from MVN Menu (Tools->DRC)

MultiView Navigator
Find / Search

- Search for Instances, Nets or Ports
  - Wildcard (*) Matching
  - Advanced Options - Choice of Log Window Pane for Search Results
- Cross-probing from Find Tab to the other Four Windows

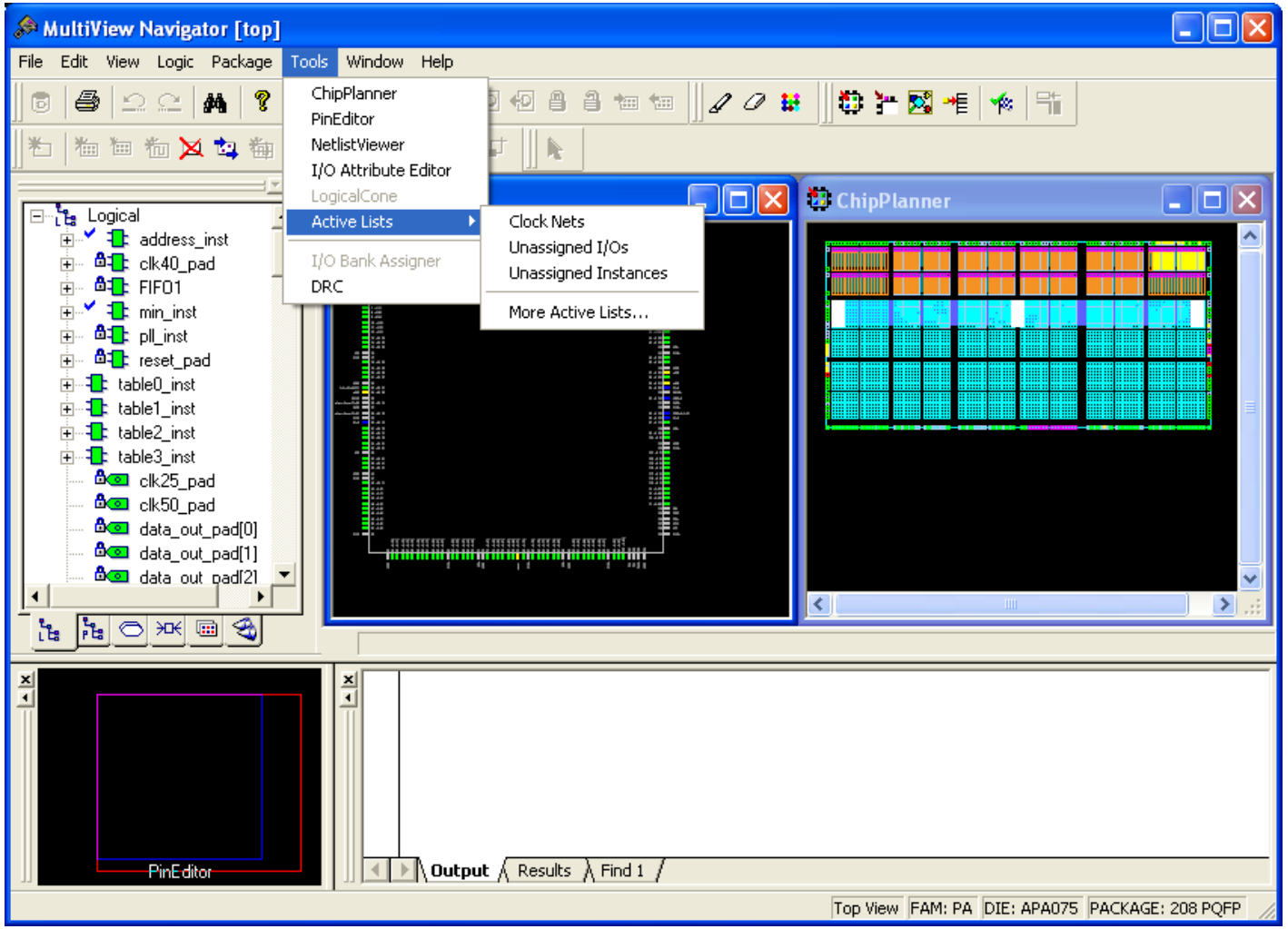


MultiView Navigator
Active Lists

- “Enables User To Compile Lists Of Important Resources
  - Instances
  - Nets
  - Macros
  - Regions
- Default Lists
  - Unassigned Instances
  - Unassigned Pins
  - Clock Nets
- Navigate Thru Design To Find Status Of Important Design Resources

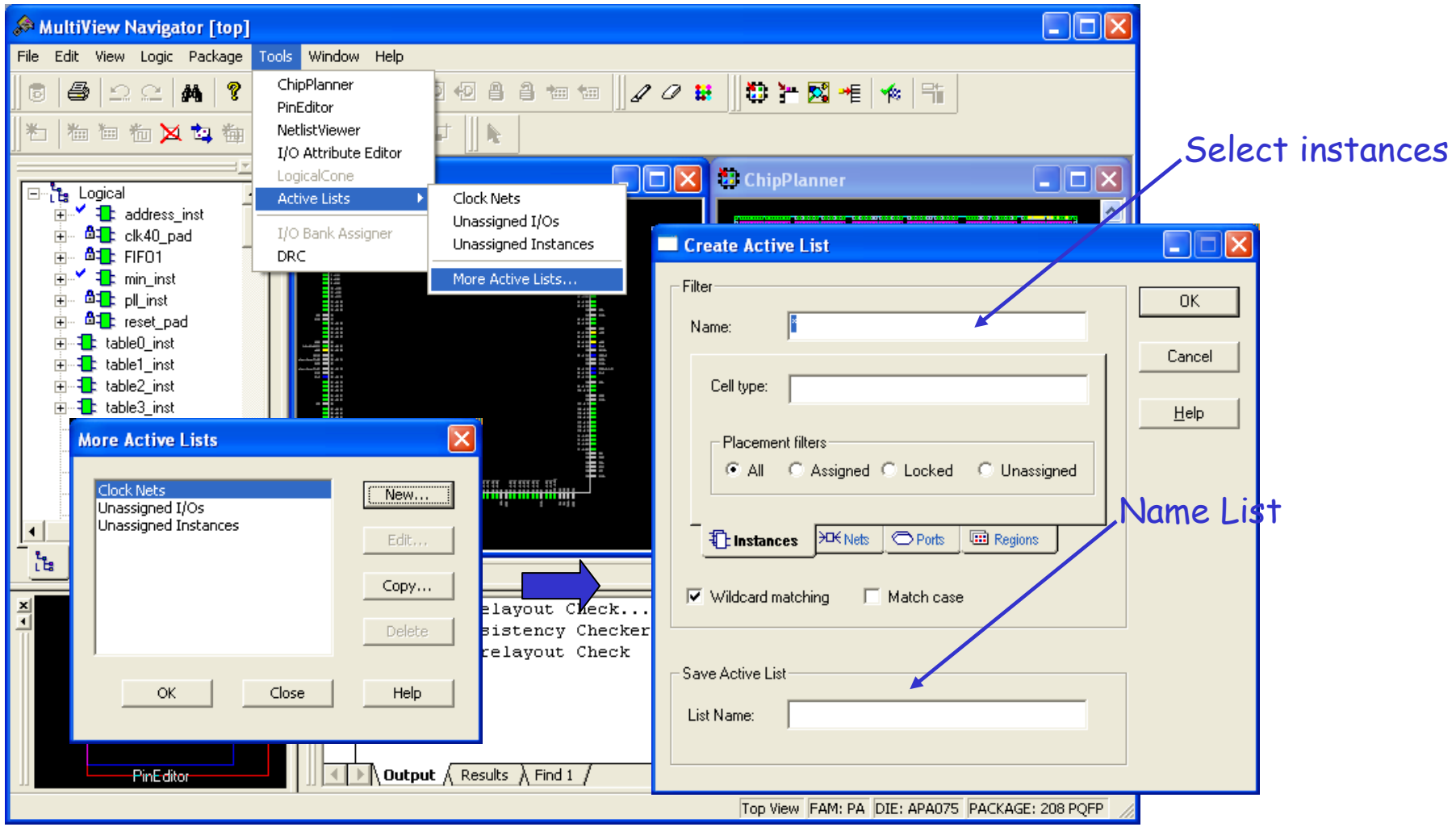
MultiView Navigator
Viewing Active Lists

- Open Default Active Lists from MVN Tools Menu



MultiView Navigator
Creating Active Lists

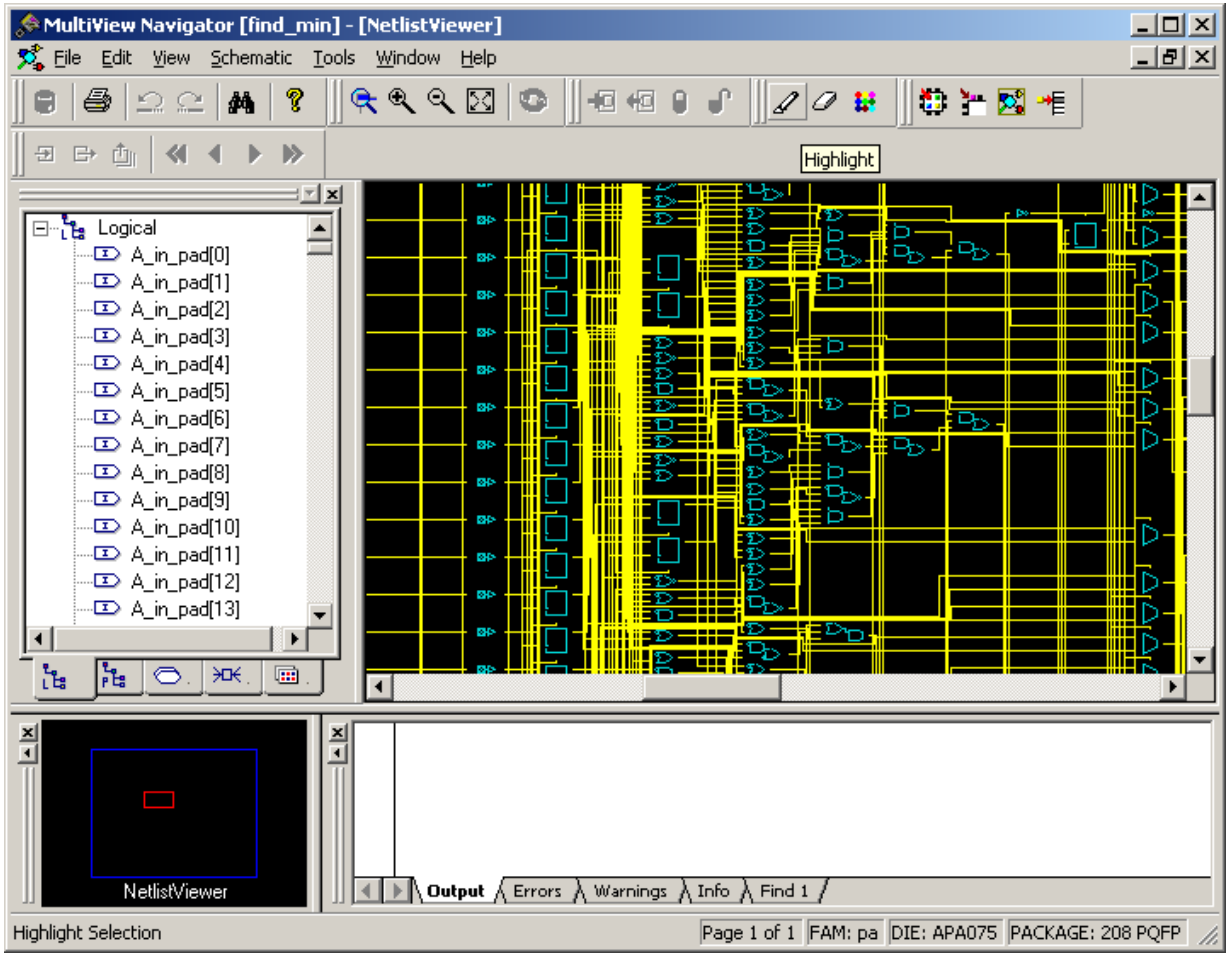
- Users Can Create New Active Lists



Netlist Viewer

MultiView Navigator
Netlist Viewer

- Displays Netlist in Hierarchical Manner
  - Explore each Level of Hierarchy and Trace Signals

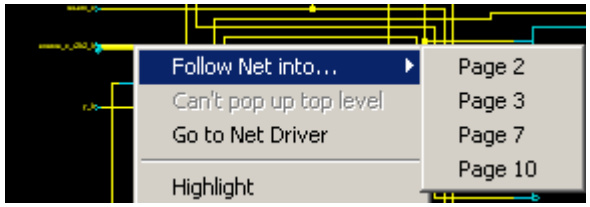


Schematic View Window – displays a graphical representation of the netlist

Search Window – shows results of searches

MultiView Navigator
Netlist Viewer (cont.)

- Viewing Options
  - Push, Pop, Jump to Top 
  - Go to First Page, Go to Last Page, Go to Next Page, Go to Last Page 
  - Right-click on Net to Follow Net to Other Pages or Net Driver

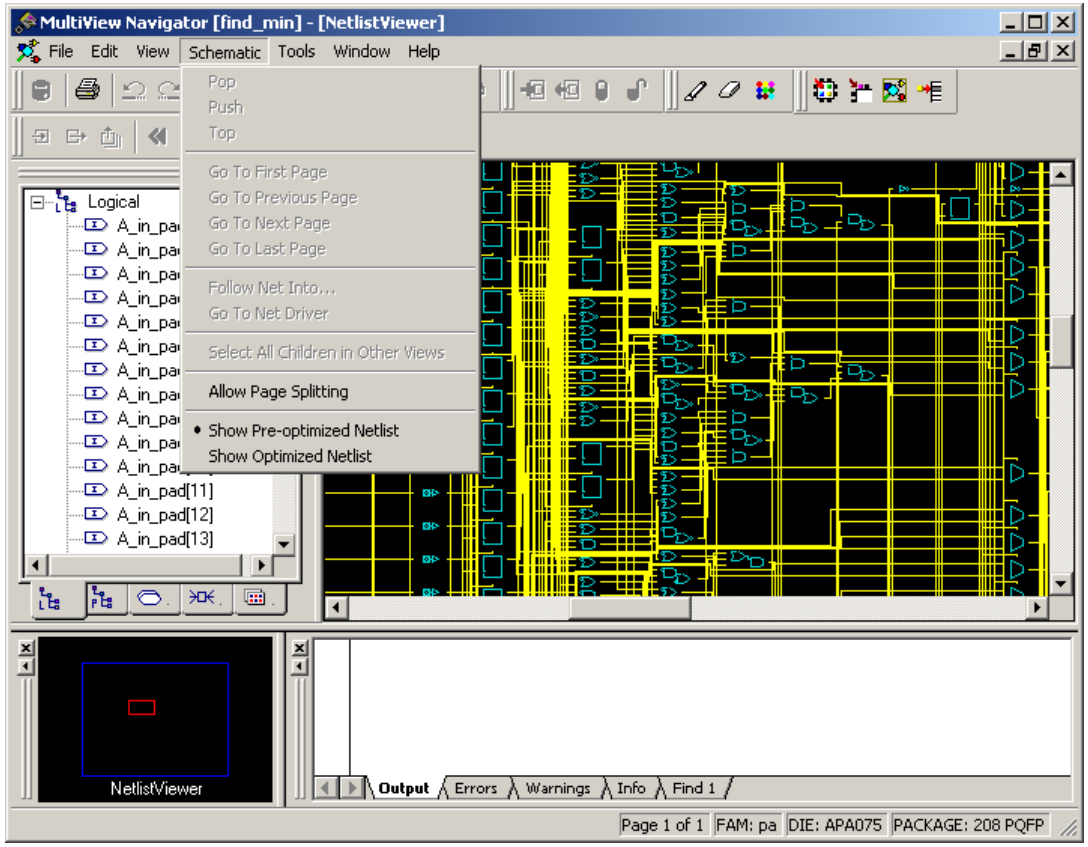


  - Highlight, Highlight Append, un-Highlight, un-Highlight All 
  - Allows Page Splitting 

    - Allows User to Decide if All Elements on that Level Are Shown as Single Page

MultiView Navigator
Netlist Viewer (cont.)

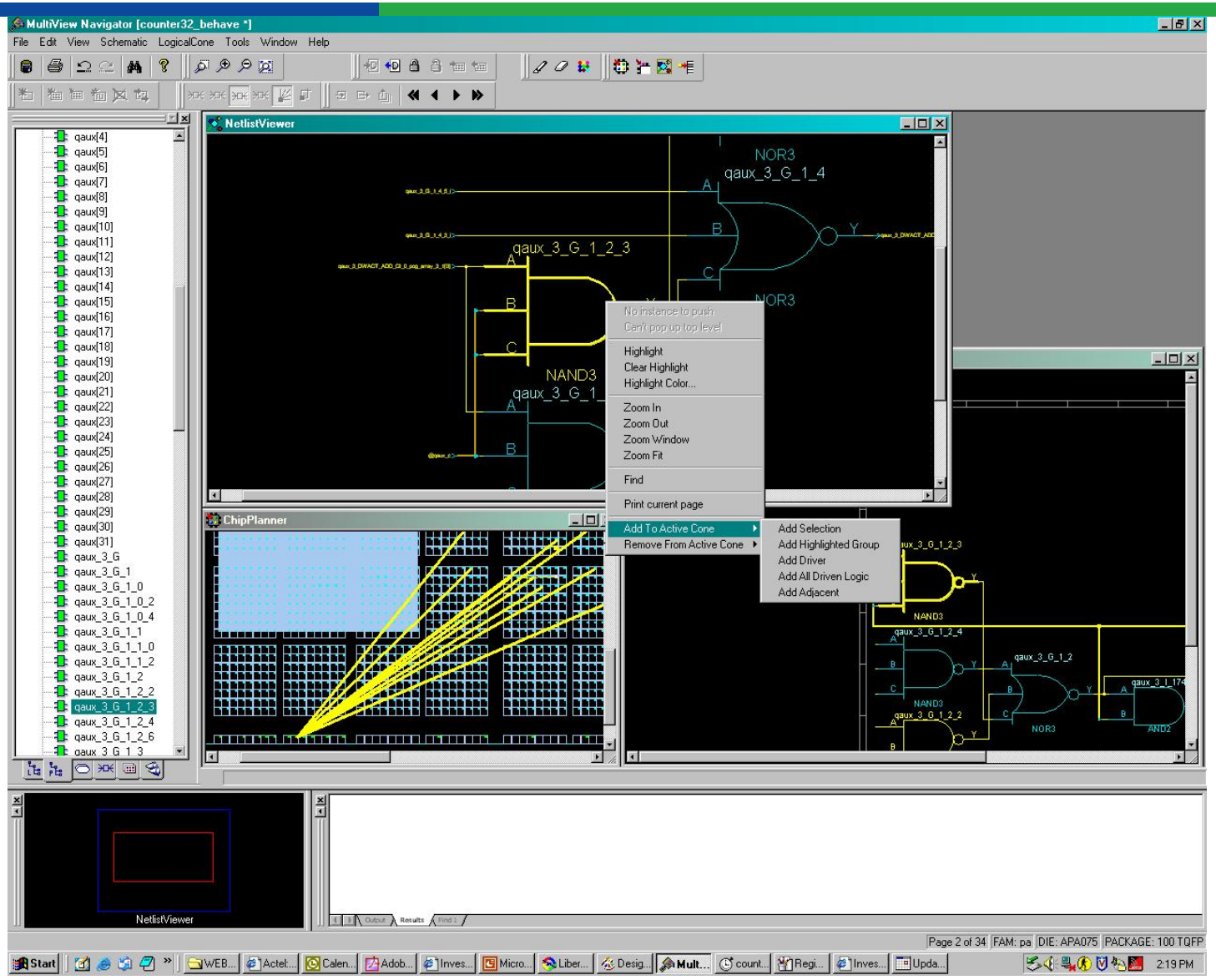
- View Pre- and Optimized Netlists for ProASIC3/E, ProASIC^{PLUS} and Axcelerator
  - Pre-optimized Netlist Is Original Hierarchical Netlist
  - Optimized Netlist is Flattened
    - Reflects what other Tools Use (ChipEdit, PinEdit)



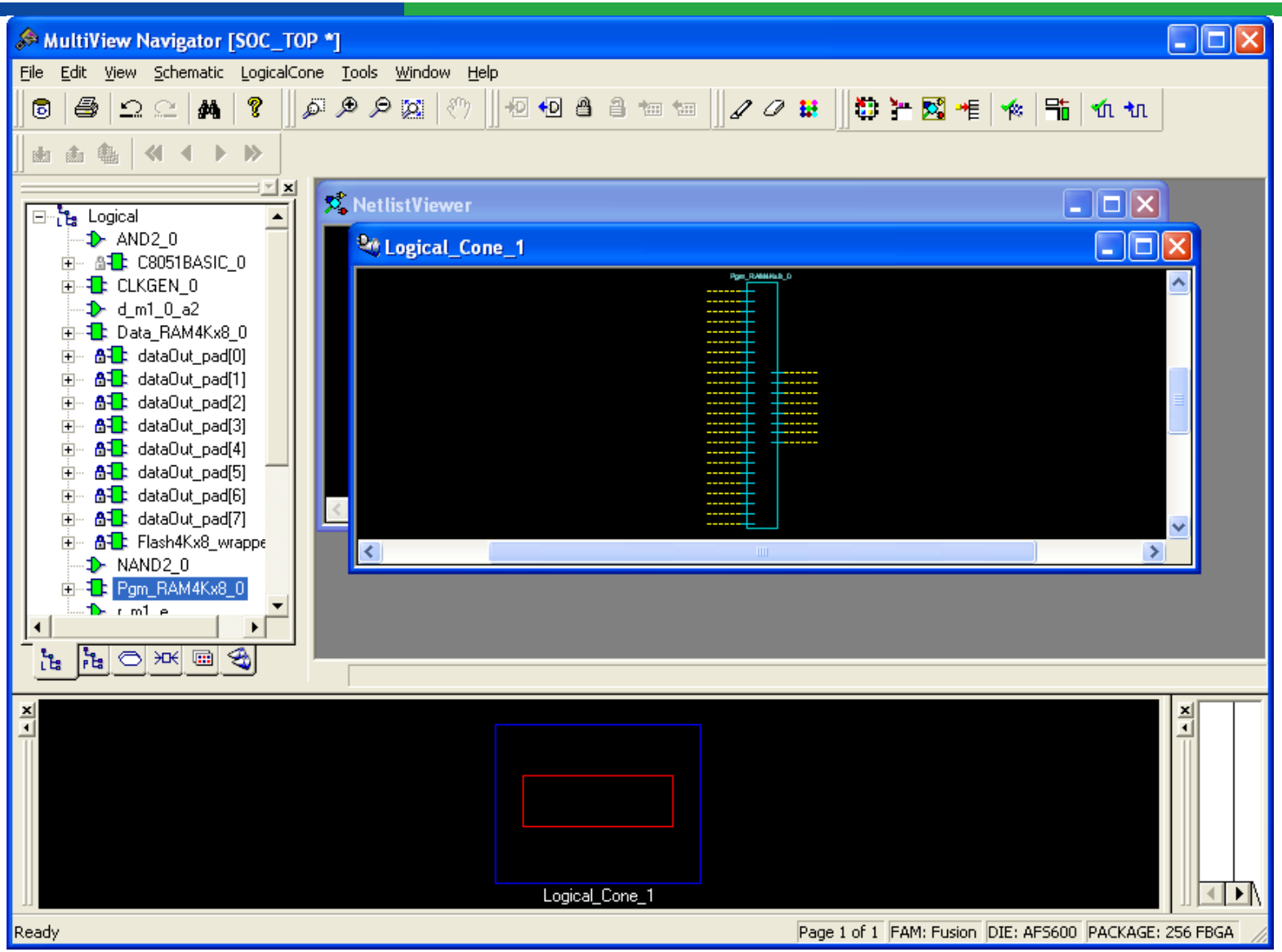
MultiView Navigator
LogicalCone

- Helps View Critical Portions of Netlist
  - Identify Critical Paths using SmartTime
  - Add this Logic to LogicalCone
  - Incrementally Add / Remove Logic from Cone
  - CrossProbe from / to LogicalCone
  - All NetlistView Features available in LogicalCone
  - LogicalCone Tab in Hierarchy Window
  - LogicalCone Menu Accessible from NetlistViewer
  - Can Simultaneously Create Multiple Cones
  - Set of Macros Can Be Highlighted, then Added to Cone
  - LogicalCone Data No Longer Valid after Recompile

LogicalCone
User Interface



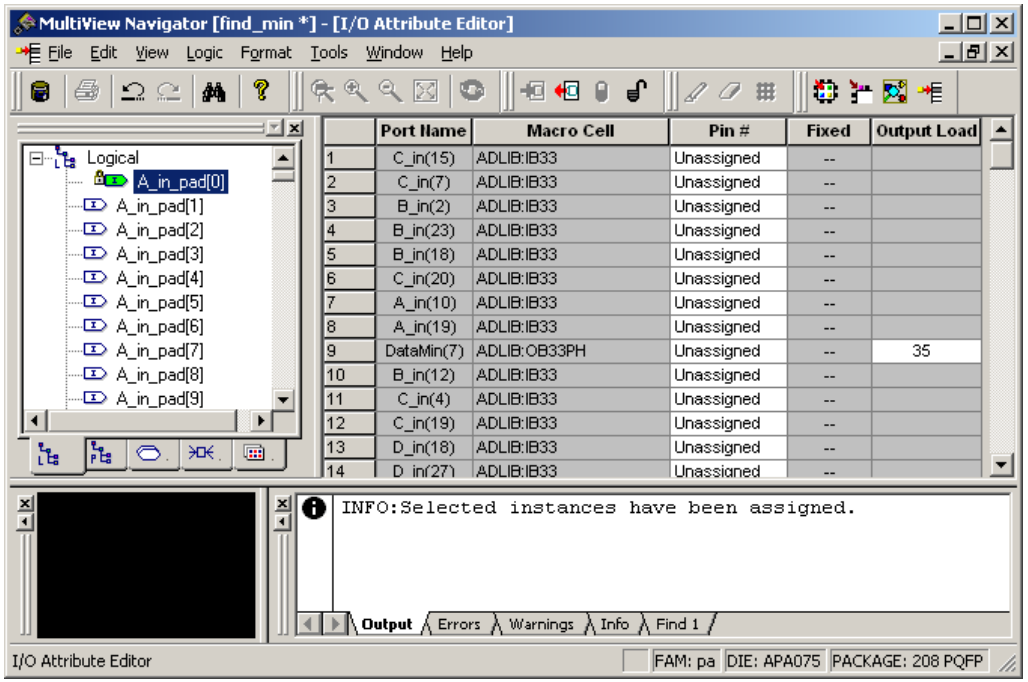
LogicalCone



I/O Attribute Editor

MultiView Navigator
I/O Attribute Editor

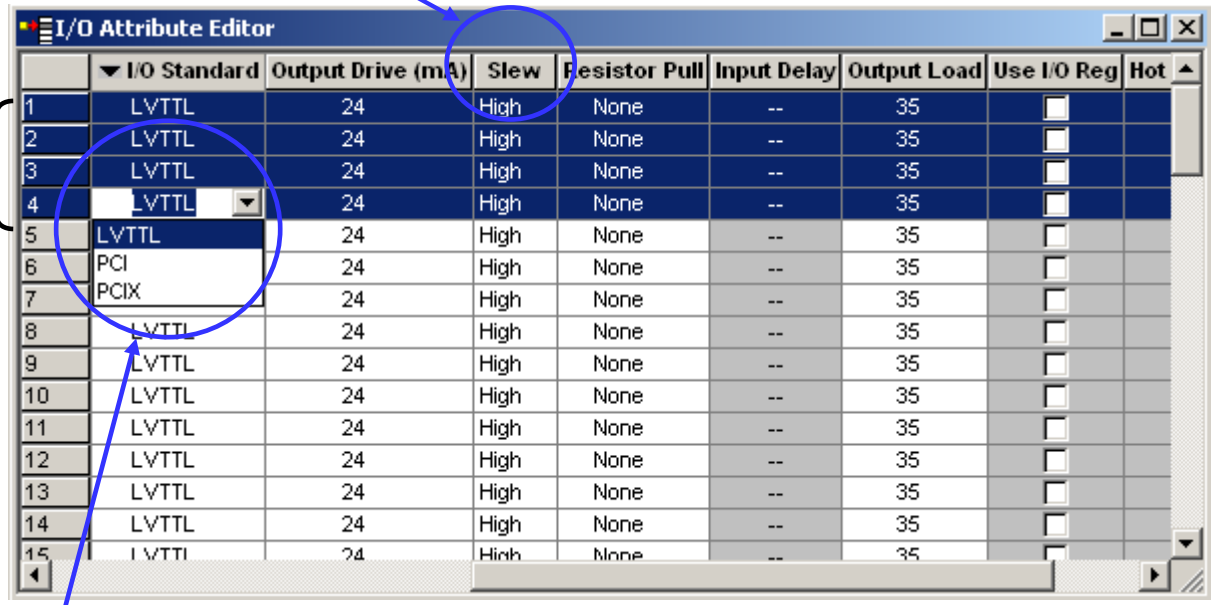
- Input/Output Attribute Editor
  - Select (Varies by Family):
    - I/O Standard
    - I/O Threshold
    - Slew Rate
    - I/O Power-up State
  - Enter Load Capacitance
    - Does Not Change SDF File Generation
- Spreadsheet-like Sort, Copy, Paste



MultiView Navigator
I/O Attribute Editor (cont.)

Double click on column to sort display by that column

Hold down CTRL key to select multiple rows

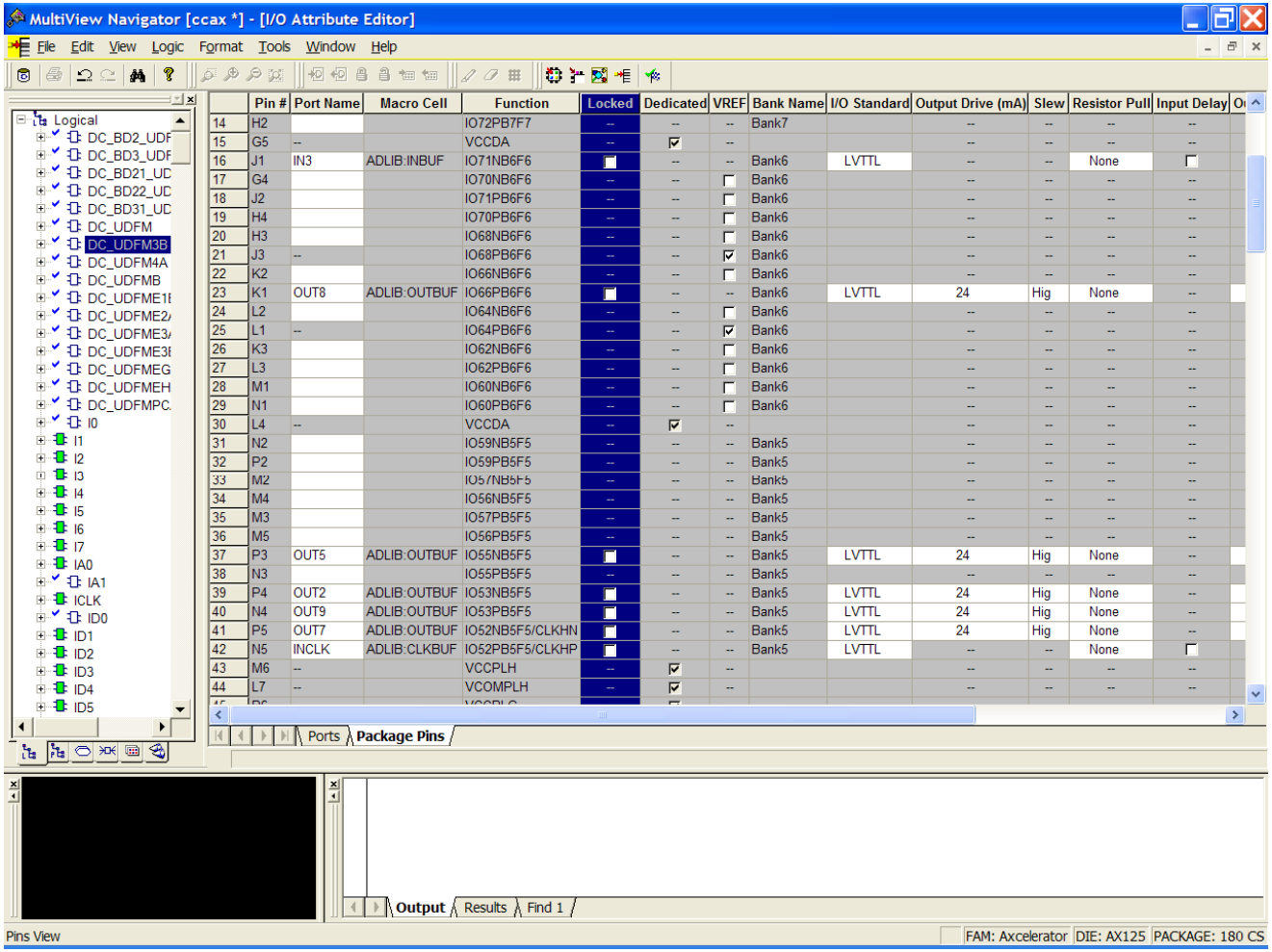


With multiple rows selected, changing the value of a drop-down item with CTRL key pressed will change the value for all rows

MultiView Navigator

Package Pins View

- Package Pins View Shows All Pins on Package



Pin Editor

Pin Assignment Options

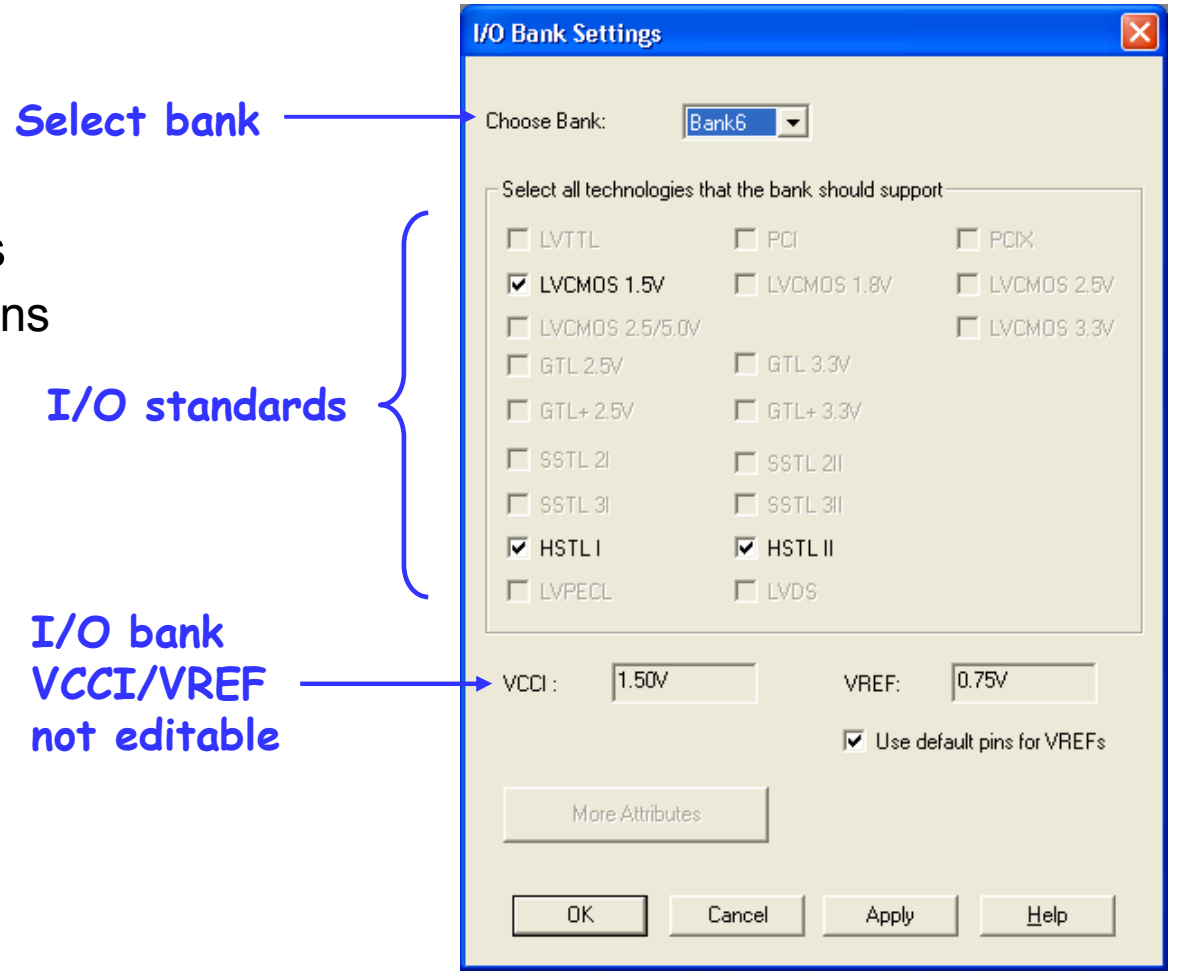
- I/O Locations Can Be Assigned as Follows:
  - Automatically by Designer Software during Layout
  - By Importing a Physical Design Constraint (.pdc) File
  - Manually using PinEdit and I/O Attribute Tools in Designer
    - Pin Assignments May Be Exported for Later Use

MultiView Navigator
PinEditor

- Graphical Pin Location Editor
  - Drag and Drop Placement of Pins
  - Fix Pin Locations for Subsequent Place and Route Runs
- Flip Display
  - Enables Assignments as if Looking from Top or Bottom of Chip
- Assign I/O Bank Properties for ProASIC3E, Fusion and IGLOOe
- Pinout Can Be Printed for Documentation

I/O Bank Configuration
ProASIC3E, Fusion and IGLOOe

- I/O Bank Configuration
  - Select Bank
  - Select I/O Standards
    - Incompatible Options Disappear when Selecting I/O Standards
  - Select Low-power Mode
  - Select Input Delay



VREF Pin Assignment

Axcelerator, ProASIC3E, Fusion and IGLOOe

- Select Package Pin And Click Right Mouse Button
  - Select “Use Pin for VREF”

Designer
Automatic I/O Bank Assignment

- I/O Bank Assigner for Axcelerator, ProASIC3/E, Fusion and IGLOO/e
  - Runs automatically during layout
  - Is available when at least one I/O bank is unassigned
  - Automatically assigns voltages to I/O banks that do not have I/Os assigned
    - Fills them with compatible I/Os
  - Assigns V_{REF} pins if required
  - Respects manual assignment already in place
  - Maintains placement quality and performance
    - No impact on device performance
  - Can be controlled manually from within the MultiView Navigator

Pin Assignment Recommendations

- Enter Design as Completely as Possible
  - Don’t Worry about Functionality
- Compile (Ignore Warnings) and Layout
- “Fix”All Pin Assignments

  Edit > Select All then Edit > Fix
- Send Pin Report to PCB Layout
- Continue Working Out Bugs
  - Future Layouts Will Honor “Fixed” Assignments

Chip Planner

ChipPlanner

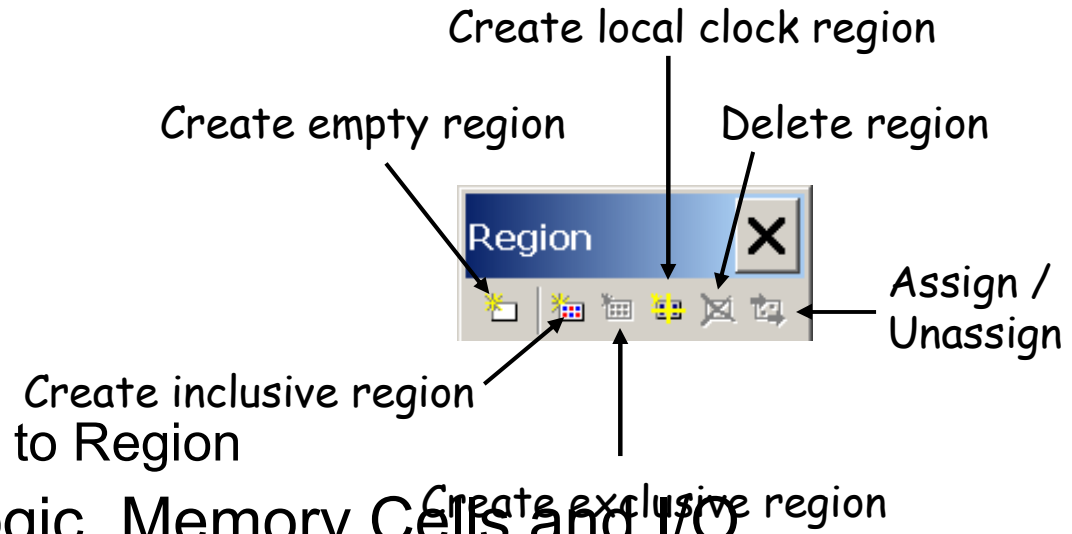
- Editing and Floorplanning Support for:
  - Fusion, ProASIC3/E, IGLOO/e, ProASIC3L, ProASIC^{PLUS}, ProASIC, Axcelerator, eX, SX-A and SX-S Families
  - Use ChipEditor for All other Actel FPGA Families
- ChipPlanner Capabilities:
  - Editing
    - Place, Unplace, or Move Logic and I/O
    - View Macro Placements Made during Layout
    - View Net Connections with Ratsnest or Route View
    - View Architectural Boundaries
    - View and Edit Silicon Features, such as I/O Banks
    - View Placement and Routing of Paths when Used with SmartTime
  - Floorplanning
    - Create and Assign Logic or Nets to Regions
  - Cross-probe with Silicon Explorer to Select Probes

ChipPlanner Terminology

- Region
  - Defined sub-Portion of Die
  - Shapes - Rectangular or Rectilinear (Union of Rectangles)
  - Types:
    - Empty - No Logic Can Be Put into this Region
    - Inclusive - Assigned Logic *Must* Be Put into this Region
      - Other Unassigned Logic Can Be Added to this Region by Layout
    - Exclusive - Only Assigned Logic Can Be Put into this Region
      - Not Supported for APA or A500K
- Assign
  - Place Logic into Particular Region or Location
    - Similar to “Place” in ChipEditor
- Lock
  - Finalizes Allocation of Logic in Particular Location
    - Similar to “Fix” in ChipEditor

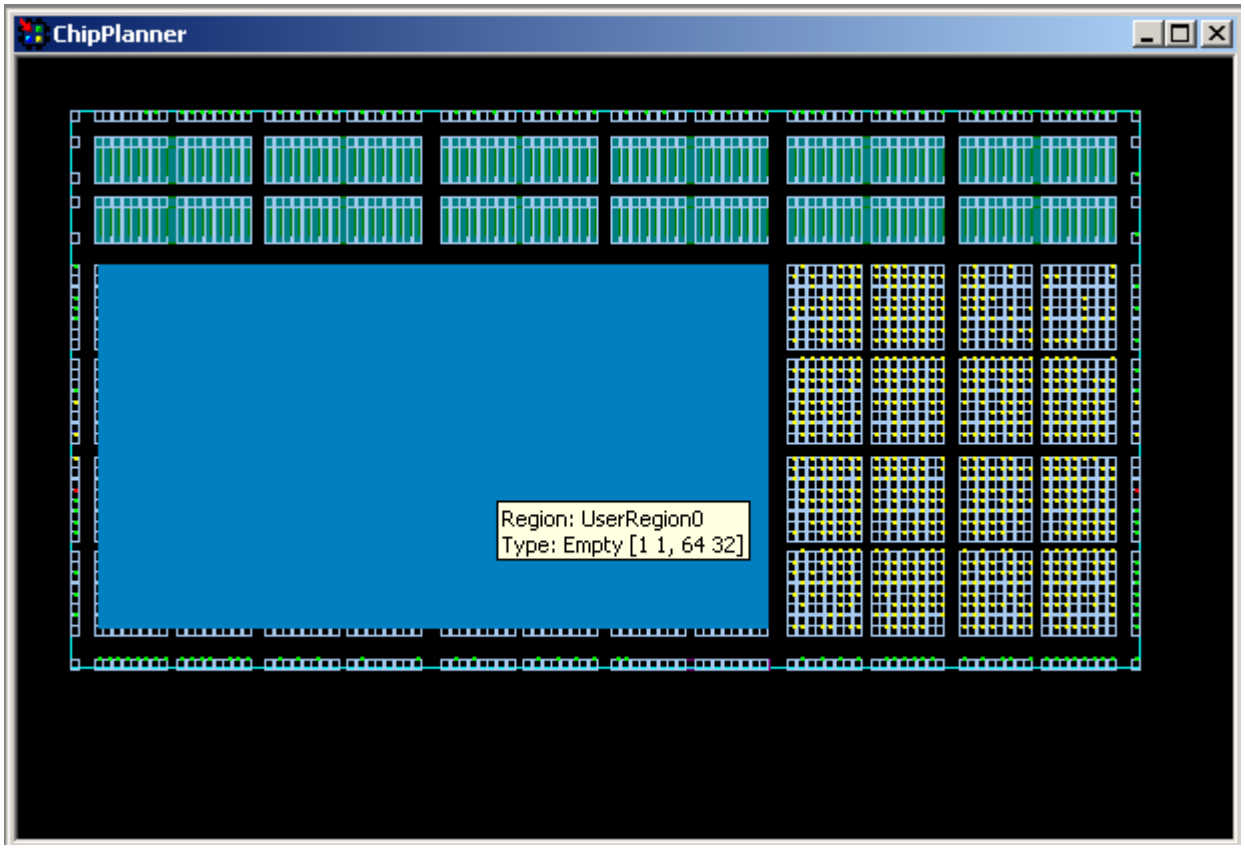
ChipPlanner
Editing and Floorplanning

- Drag Logic or I/O to Desired Location
- ChipPlanner Floorplanning Functions
  - Create Logic Region
  - Create Empty Region
  - Select Region
  - Move Region
  - Delete Region
  - Resize Region
  - Assign/Unassign Logic to Region
- Regions Can Span Logic, Memory Cells and I/O



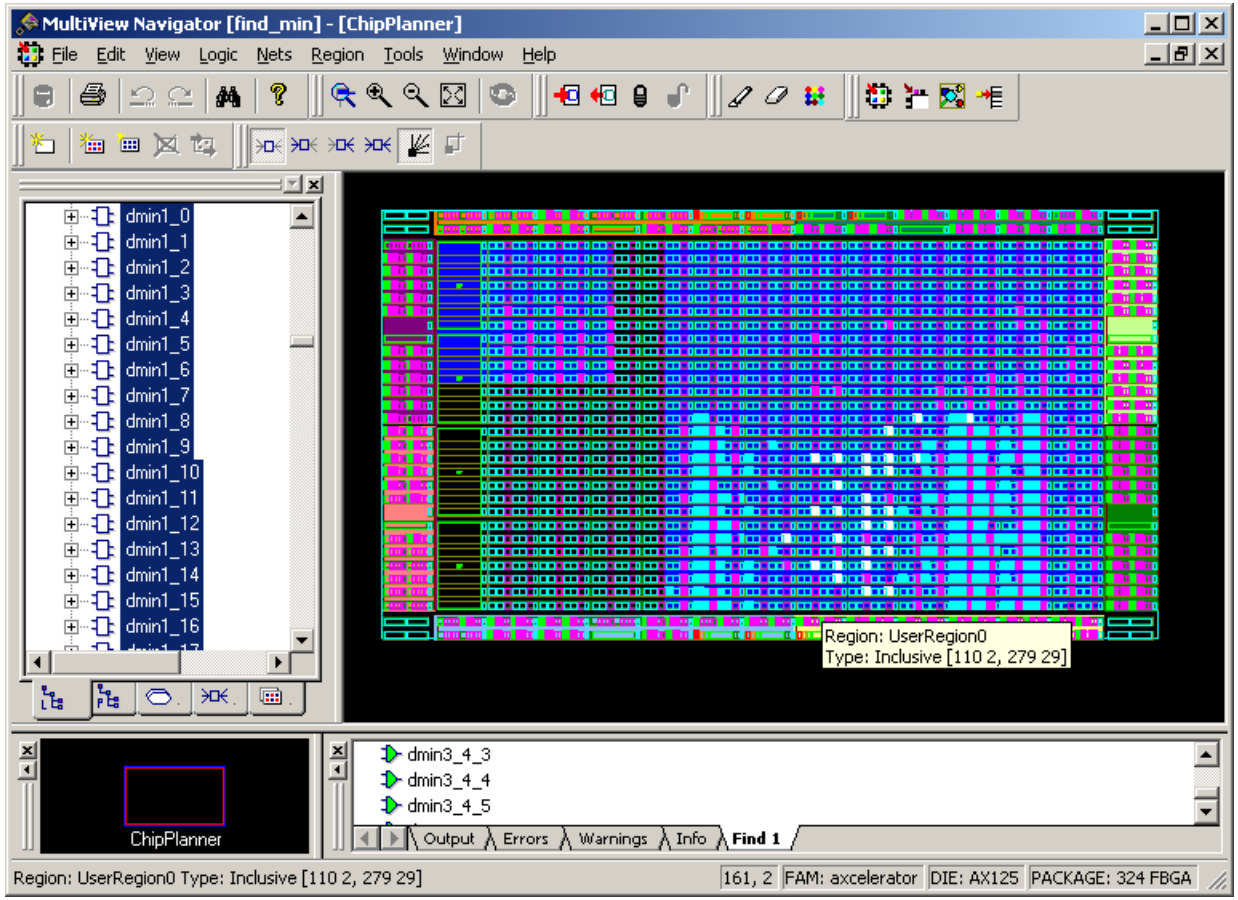
ChipPlanner
Empty Region

- Region > Create Empty
  - No Logic Assigned to Empty Regions



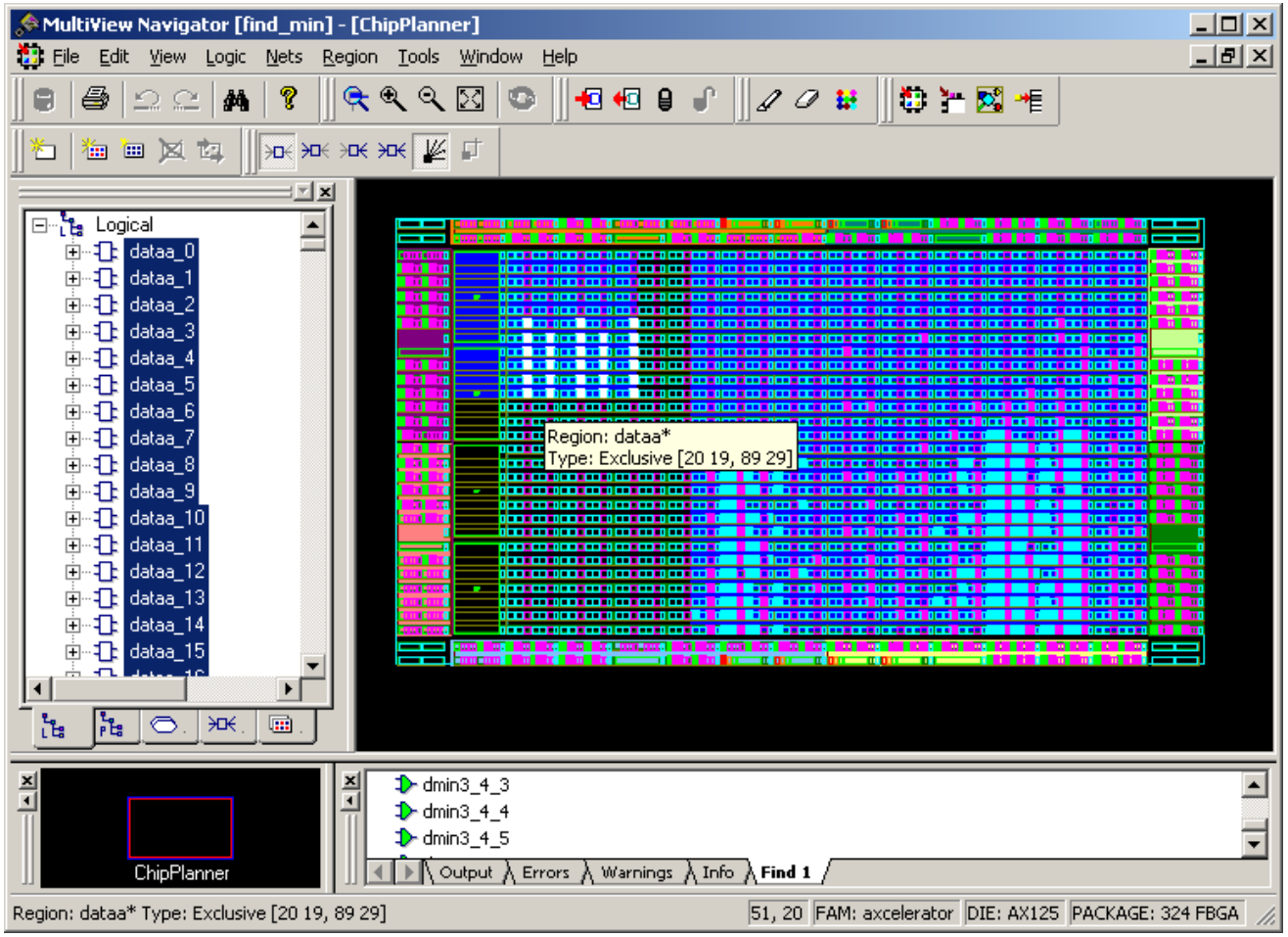
ChipPlanner
Inclusive Region

- Region > Create Inclusive
  - Assigned Logic Put in Inclusive Region
    - Other Logic May also Be Put in this Region



ChipPlanner Exclusive Region
ProASIC3/E, ProASIC3L, Fusion, IGLOO/e

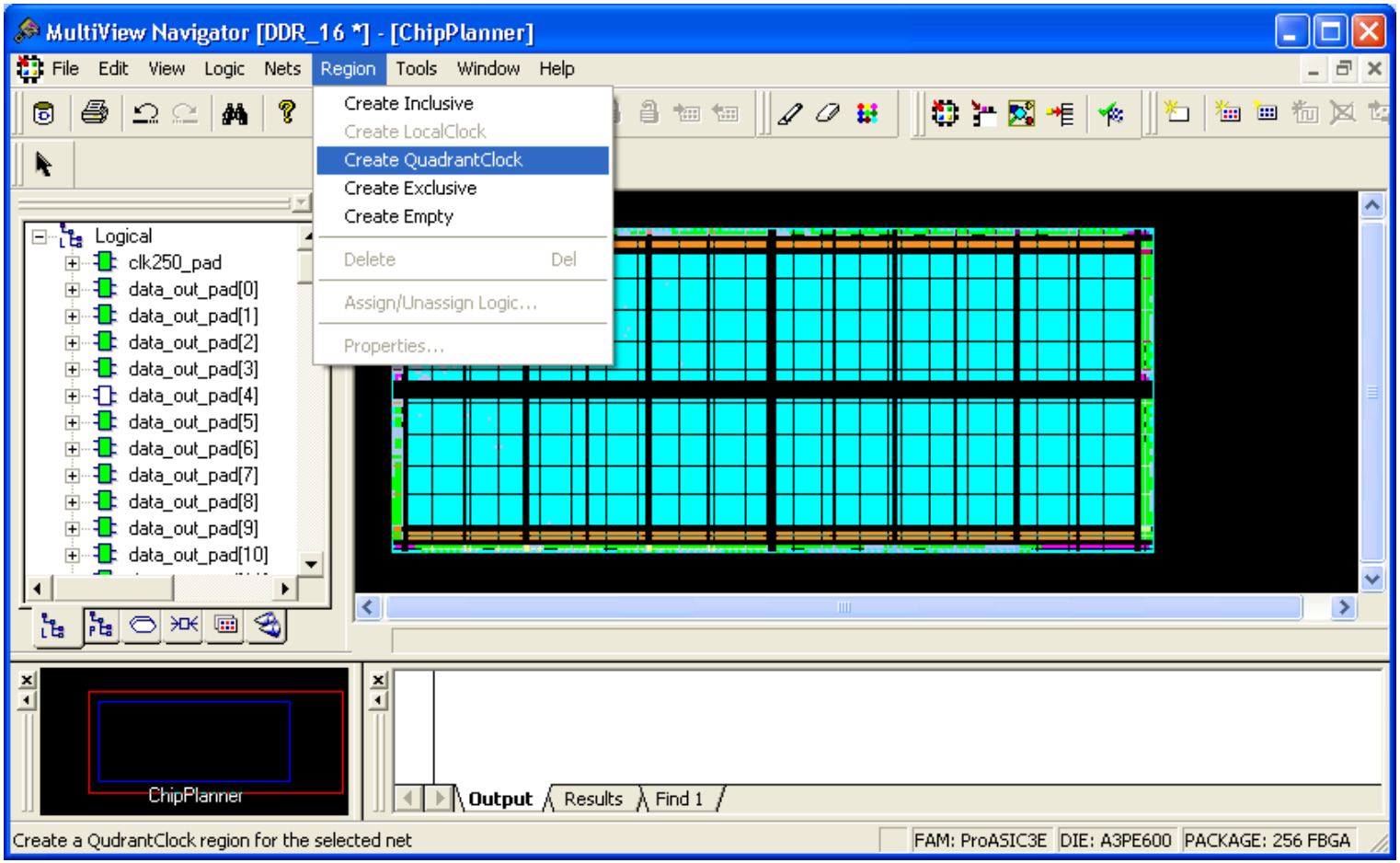
- Region > Create Exclusive
  - Only assigned logic placed in exclusive region



ChipPlanner Quadrant Clock Region

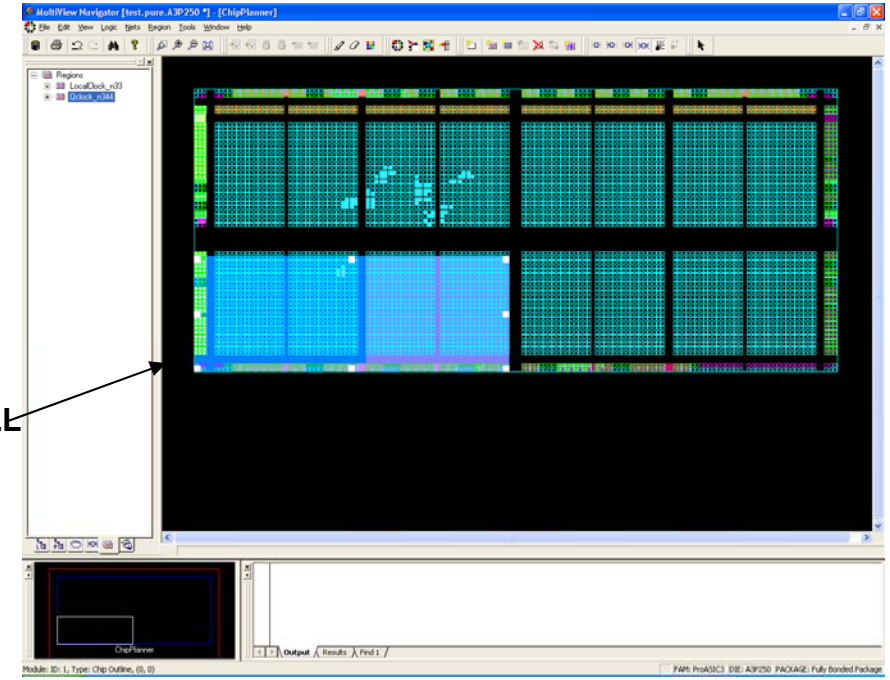
ProASIC3/E, ProASIC3L, Fusion, IGLOO/e

- Region > Create QuadrantClock
  - Graphically Assign net to a Quadrant Clock



Quadrant Clock Region PCD Constraint
ProASIC3, ProASIC3L, Fusion, IGLOO

- Create Quadrant Clock Regions with PDC Constraint
  - PDC Syntax:
    - assign_quadrant_clock -net *netname* -quadrant *quadrant_clock_region [-fixed value]*
    - *quadrant_clock_region is UL, UR, LL, LR*

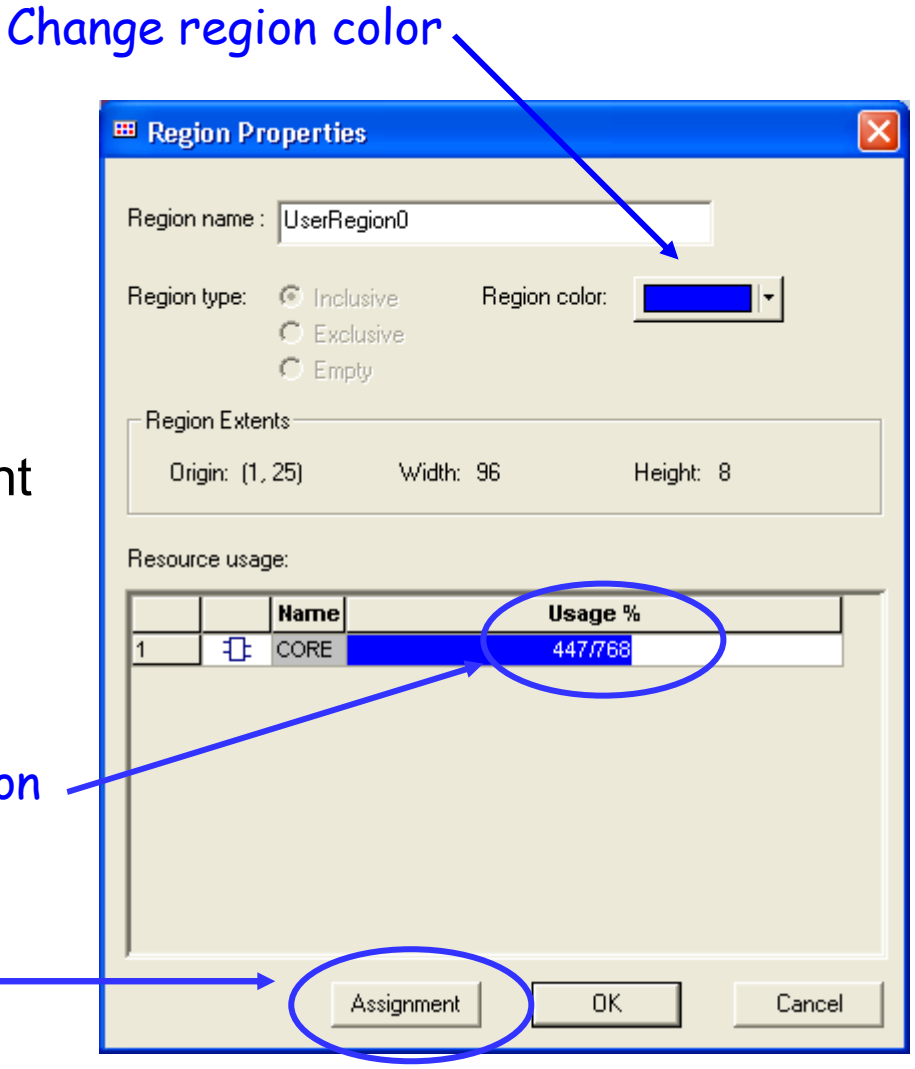


ChipPlanner
Region Color Control

- Individual Region Color Control
  - Regions Have Different Default Colors Based on Types
  - Can Change Each Region's Default Color
  - Region Colors Saved in .adb File
  - Region Colors Reset to Defaults upon Recompile

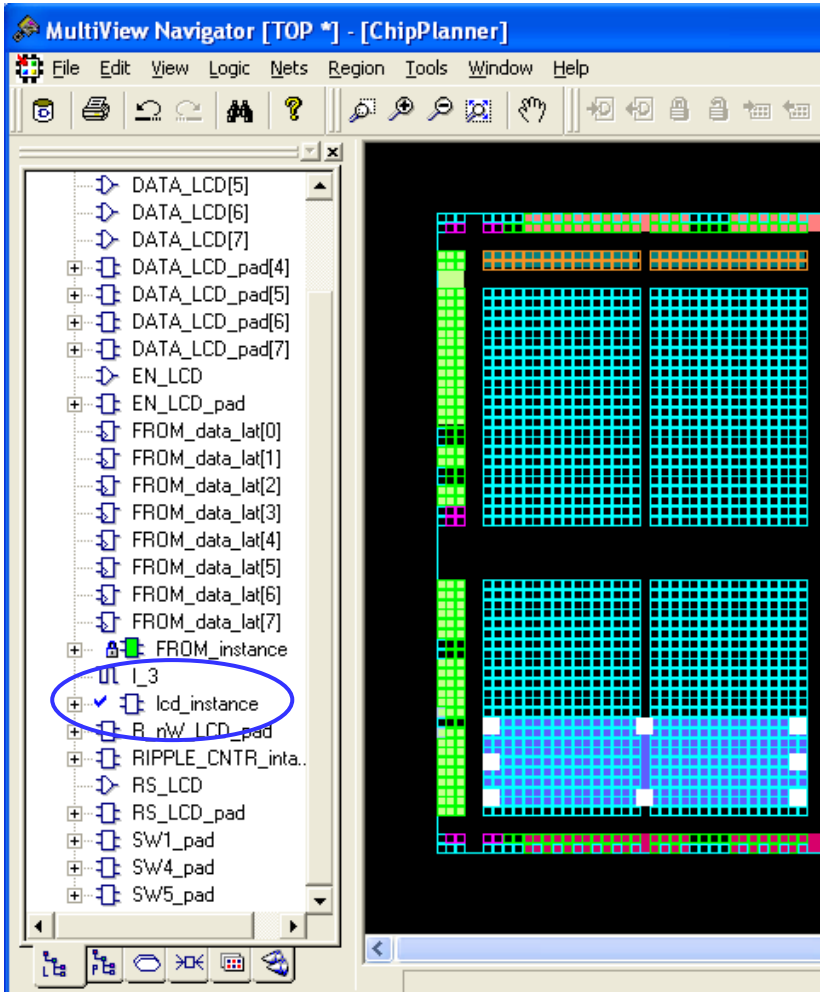
ChipPlanner
Region Properties

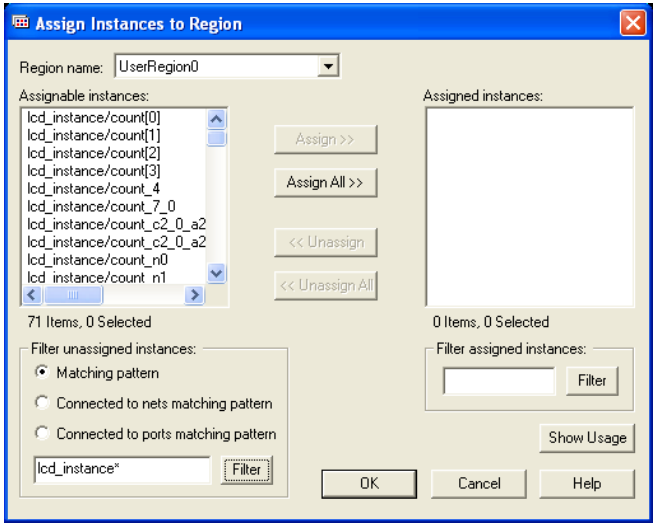
- Region > Properties
- Indicates:
  - Region Type
  - Region Width, Height and Origin
  - Region Usage
- Shows Region Default Color
  - Default Color Can be Changed
- Also Provides Access to Assignment Window



ChipPlanner
Logic Assignment

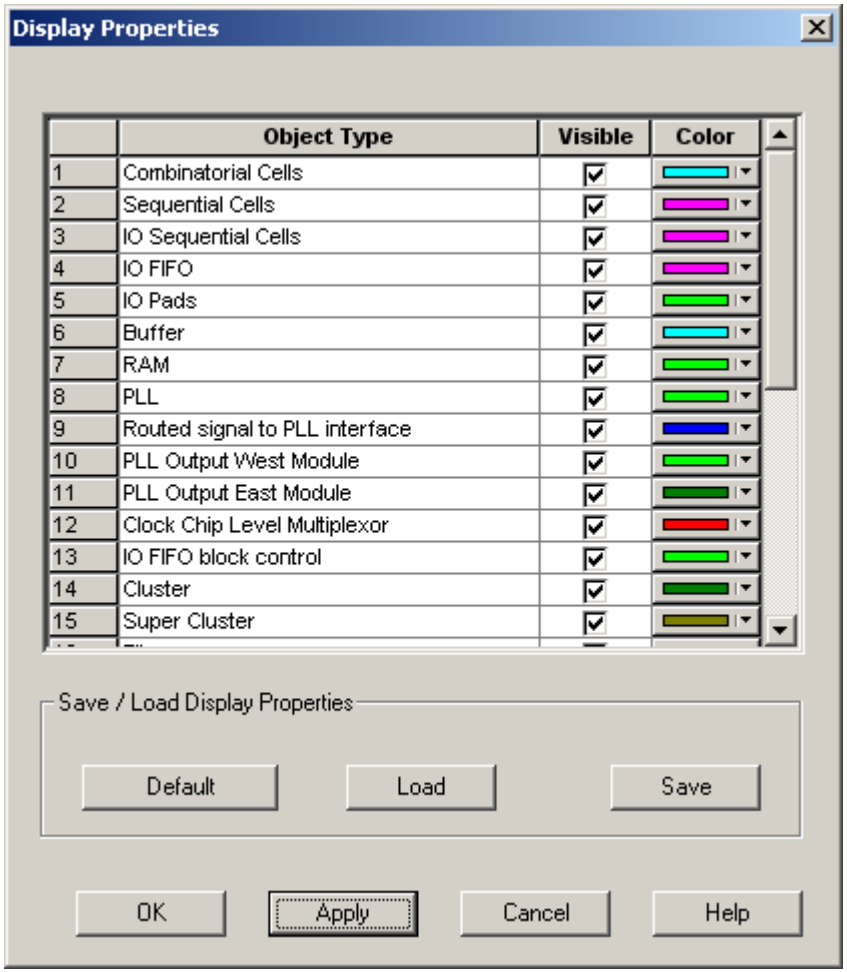
- Two Logic Assignment Methods
  - Assignment Window (Region > Assign/Unassign Logic…)
    - Provides Search and Selection Capability
    - Also Available from Region Properties Dialog Box
  - Drag and Drop Logic into Region
- Checkmark Indicates Assigned Logic





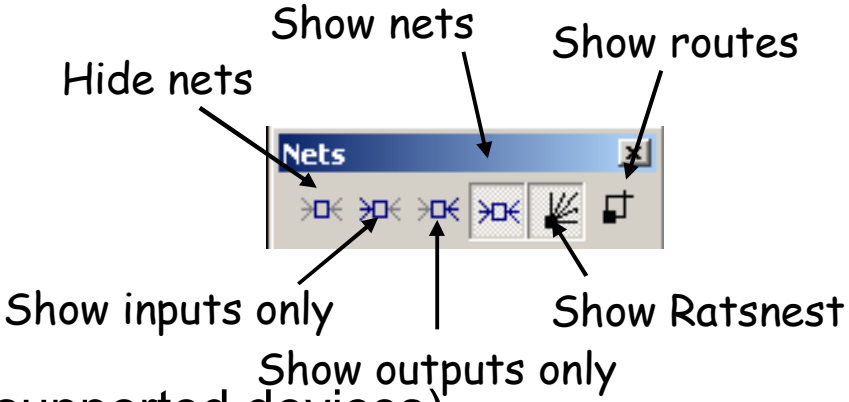
ChipPlanner
Display Settings

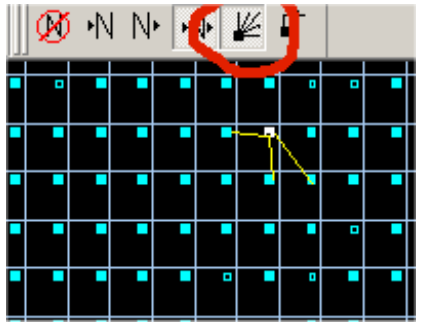
- Users Can Show/Hide Object and Assign Color to Resource
  - View > Display Settings



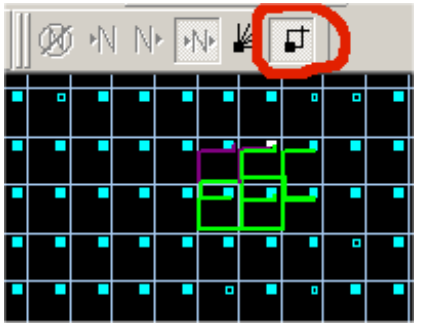
ChipPlanner
Viewing Nets

- Select Net View Options from Nets Toolbar:
  - Display No Nets
  - Display Input Nets
  - Display Output Nets
  - Display Input and Output Nets
  - Display Ratsnest or Routes (All supported devices)





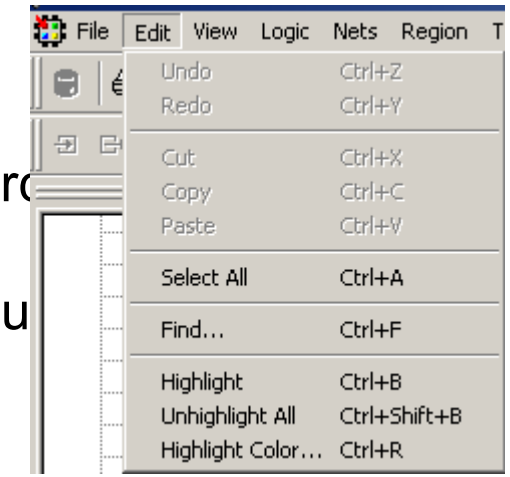
View Ratsnest

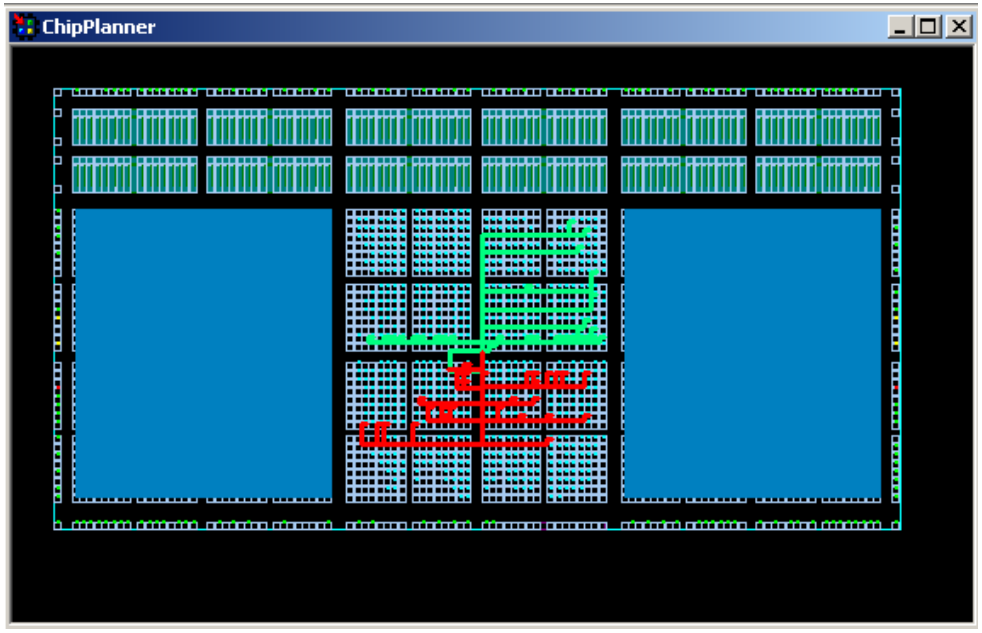


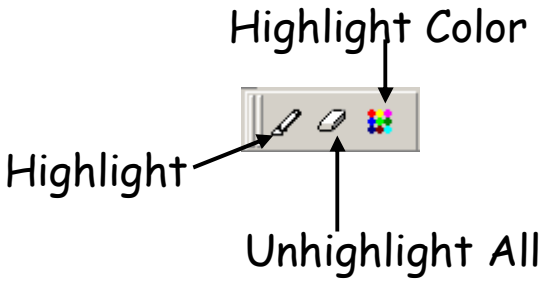
View Routes

ChipPlanner
Highlighting Nets

- Selected Nets Can Be Highlighted to Aid Analysis
  - Select Net from Design Window Nets Tab or Searc Results
  - Change Highlight Color from Toolbar or Edit Menu



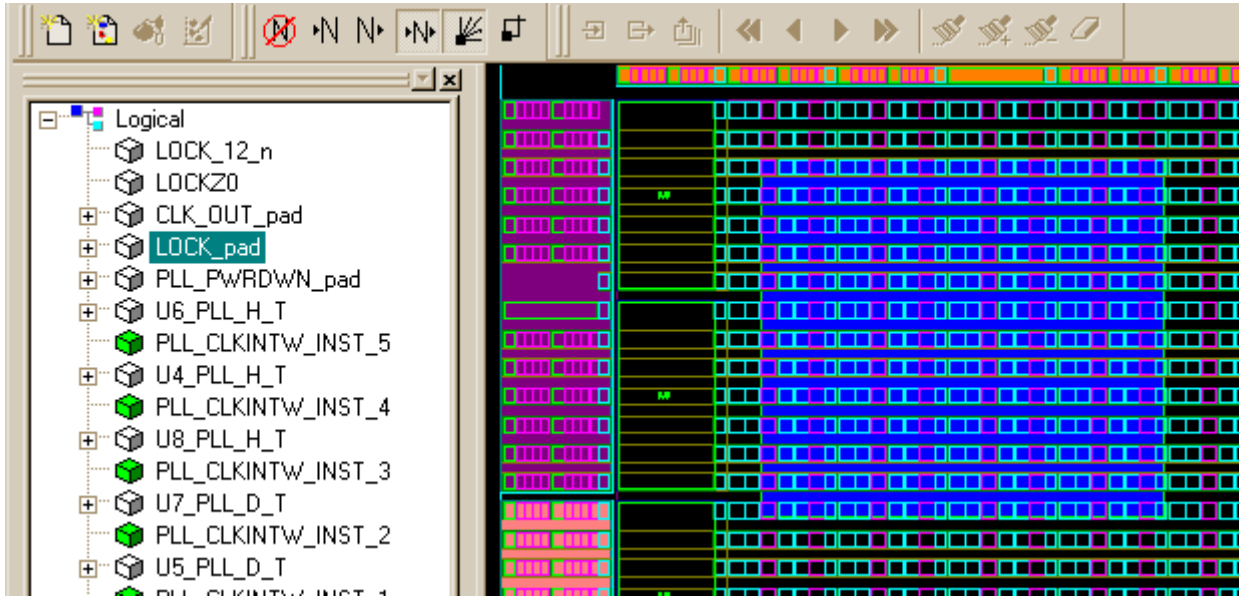




ChipPlanner

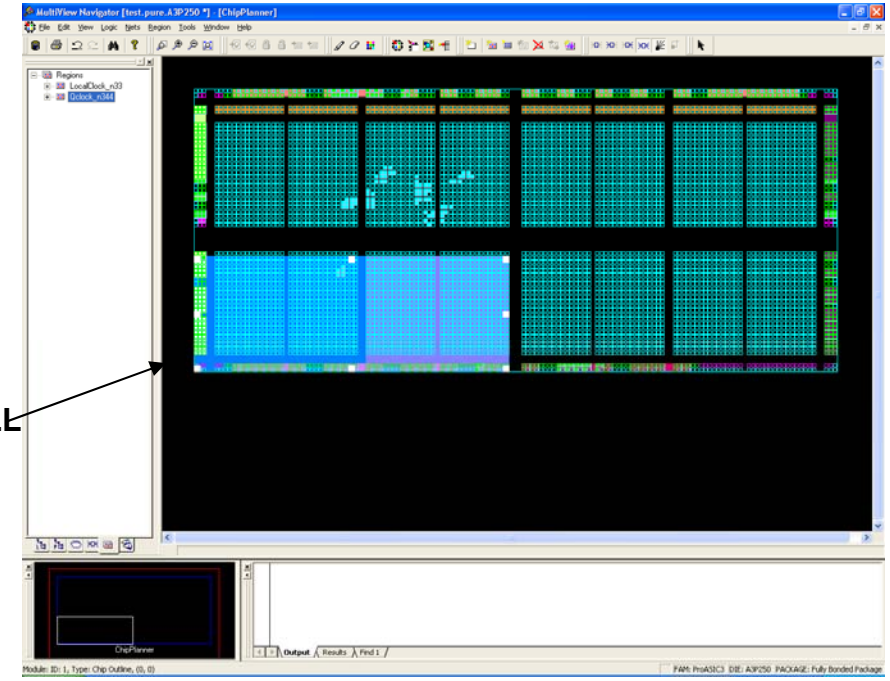
ProASIC3/E, ProASIC3L, Fusion, IGLOO/e, Axcelerator

- Floorplanning Capabilities
  - Define Rectangular Regions
    - Empty, Exclusive and Inclusive Region Support for Axcelerator
    - Modify Region types (Inclusive / Exclusive)
  - Assign Logic and Nets to Region
    - I/O Assignments
    - PLL and RAM Assignments
  - Drag and Drop Assignments
  - Multi-region Assignments NOT Recommended



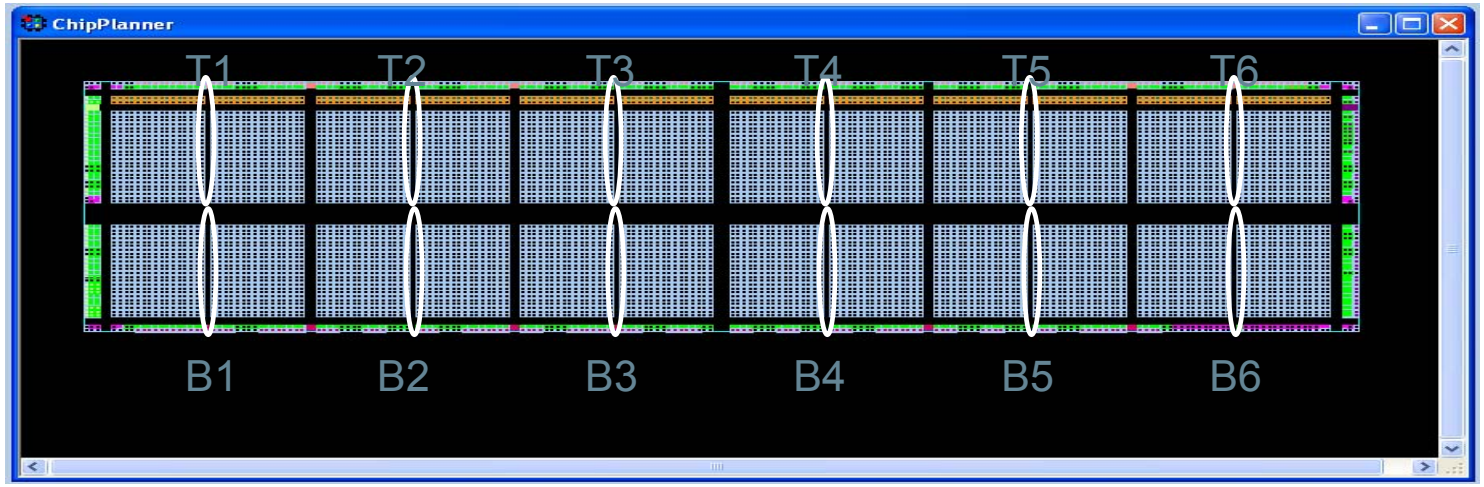
Quadrant Clock Region Support
ProASIC3/E, ProASIC3L, Fusion, IGLOO/e

- Create Quadrant Clock Regions with PDC or in ChipPlanner
  - PDC Syntax:
    - assign_quadrant_clock -net *netname* -quadrant *quadrant_clock_region [-fixed value]*
    - *quadrant_clock_region is UL, UR, LL, LR*



Local Clock Region Support
ProASIC3/E, ProASIC3L, Fusion, IGLOO/e

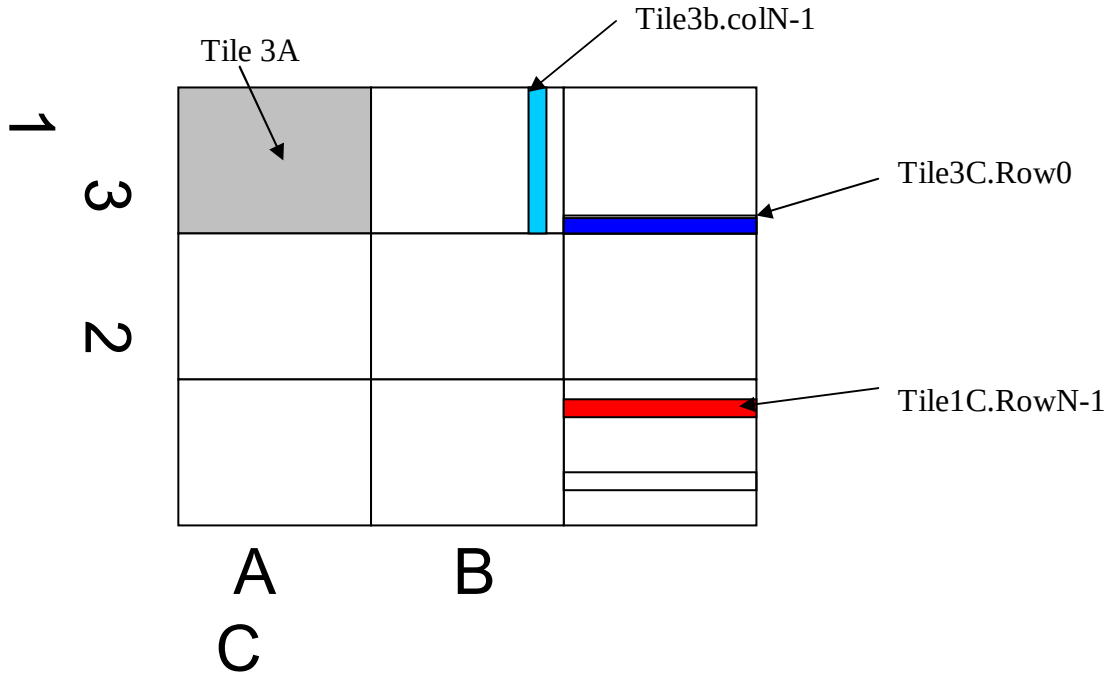
- ProASIC3/E, IGLOO/e and Fusion Local Clock Regions
  - Created through PDC only
  - PDC Syntax:
    - *assign_local_clock -net netname -type clock_type clock_region*
    - *clock_type* is either chip or quadrant
    - *clock_region*:
      - A multi-spine rectangle defined as [T | B]#:[T | B]#
      - A single spine defined as T# (Top spine) or B# (Bottom spine)



A3P400 Local Clock Locations

Local Clock Region Support
Axcelerator

- Axcelerator Local Clock Regions
  - Created through PDC only
  - PDC Syntax:
    - *assign_local_clock –type routing_resource_type –net netname <local_clock_region> [local_clock_region] [local_clock_region] …*
    - *routing_resource_type* is either hclk or rclk
    - *local_clock_region* is Hierarchical Resource Name of Specific Clock Region

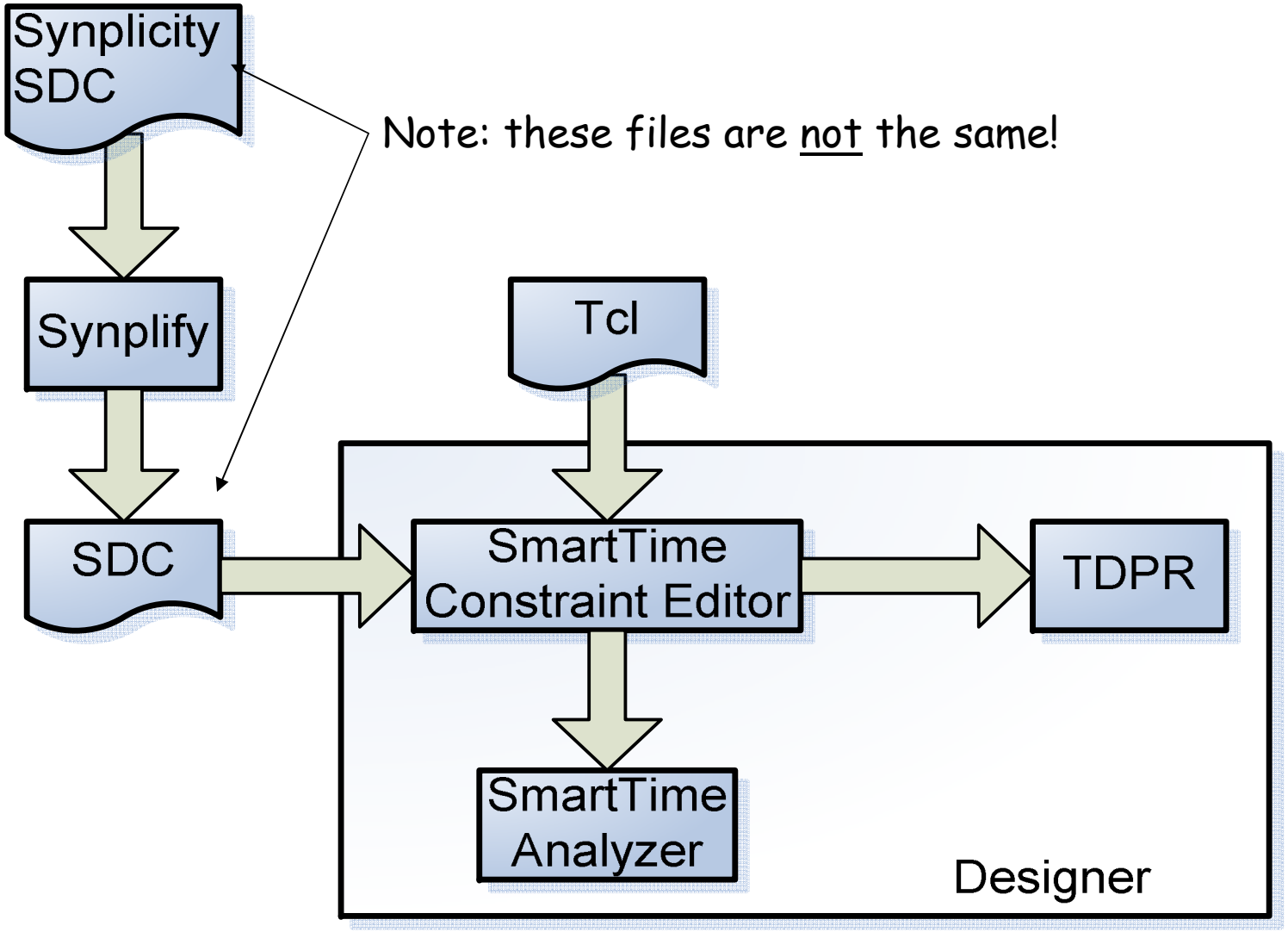


Hands-on Labs 5 and 6

- Complete Labs 5 and 6 in the Lab Guide
- Lab 5
  - Open Designer and Compile Design
  - Save your Work
- Lab 6
  - Use PinEditor to Make pin assignments
  - Export Pin Report

Entering Timing Constraints

Timing constraint flow

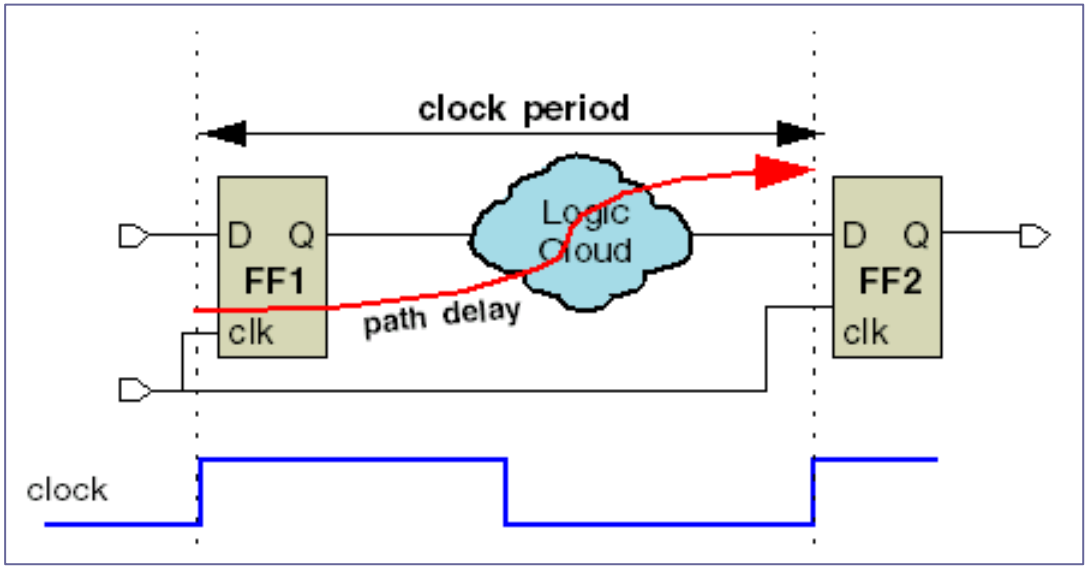


Timing Requirements / Exceptions

- Timing Requirements Are:
  - Clock Frequency
  - Input Delay
  - Output Delay
- Timing Exceptions Are:
  - Multi-cycle Paths
  - False Paths
  - Maximum Delay
  - Minimum Delay
- Timing Requirements And Exceptions:
  - Direct Synthesis Mapping
  - Guide Place and Route and Timing Analysis of a Design

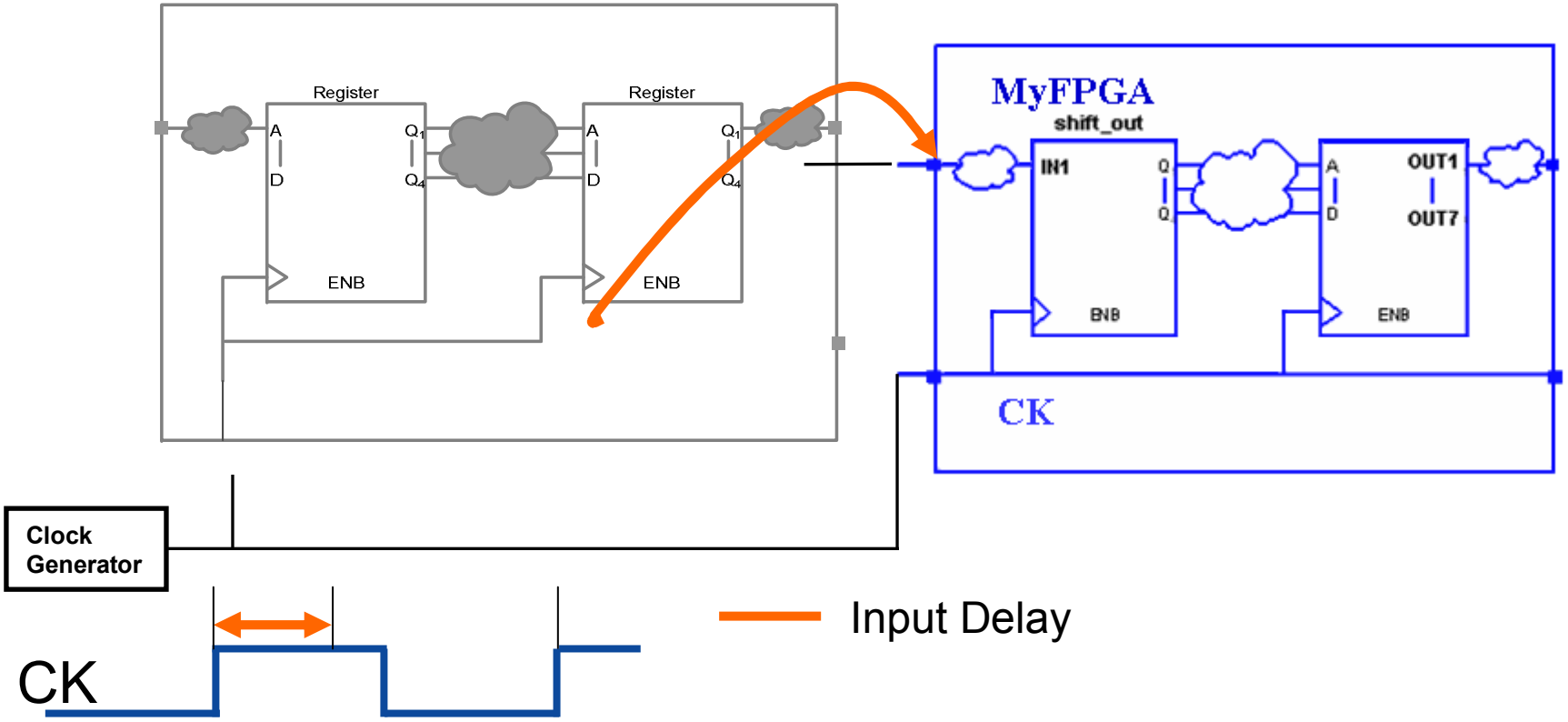
Timing Constraints
Clock Frequency

- Clock Constraints Define the Timing Requirements Between Flip Flops For Each Clock Network



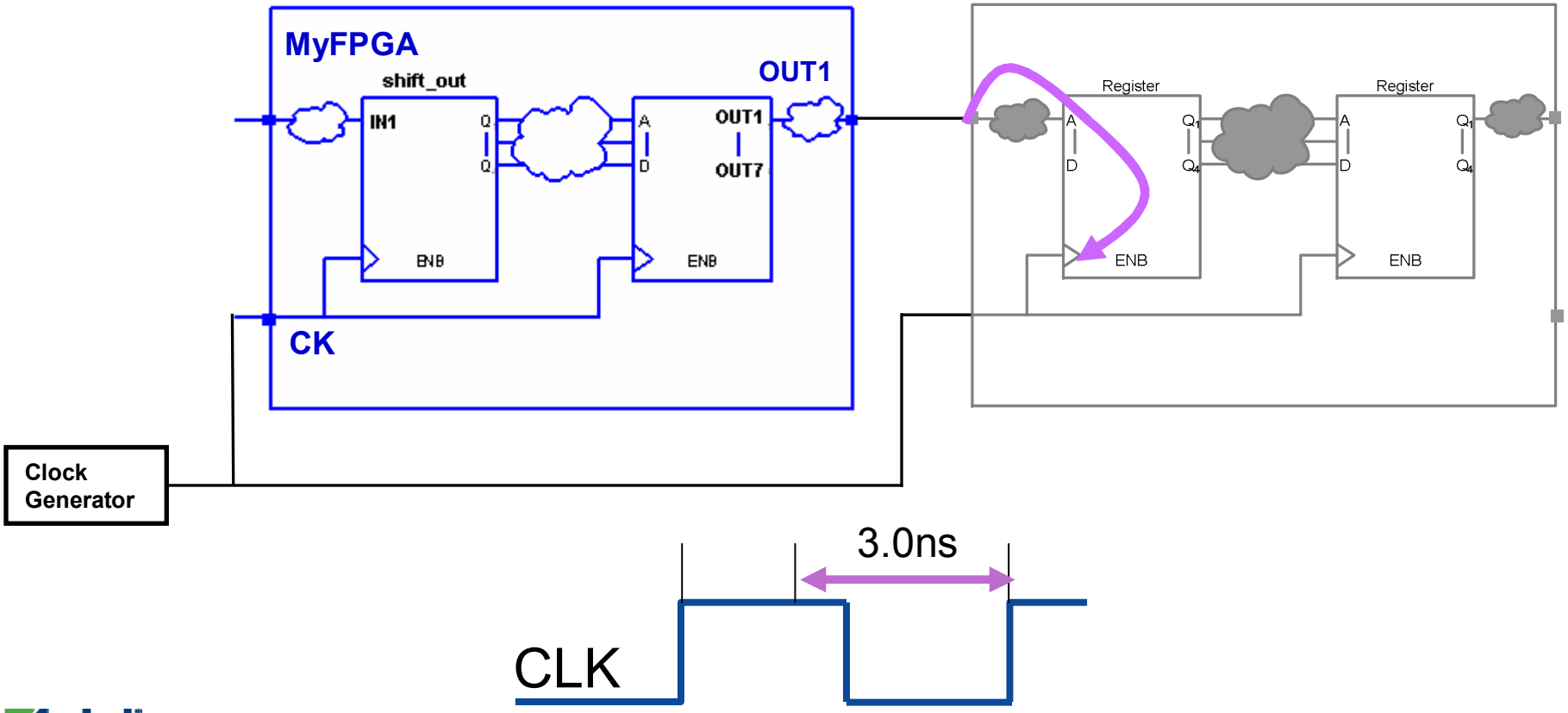
Timing Constraints
Input Delay

- Input Delay Specifies When a Signal Arrives at The Input Ports of the FPGA
  - Referenced from the *Launch Edge* Outside the FPGA



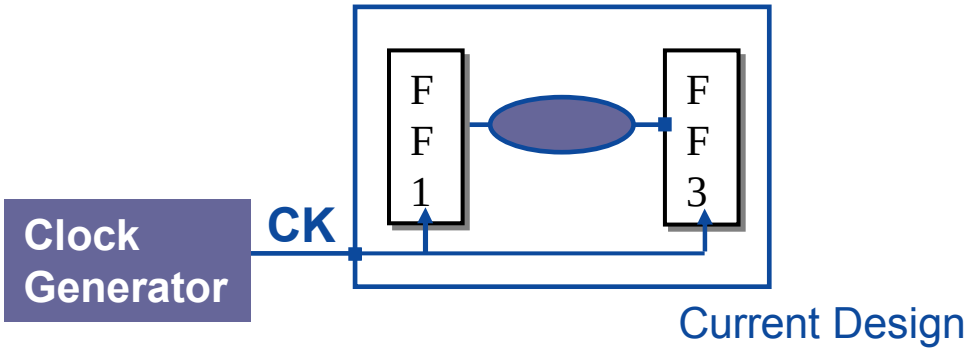
Timing Constraints
Output Delay

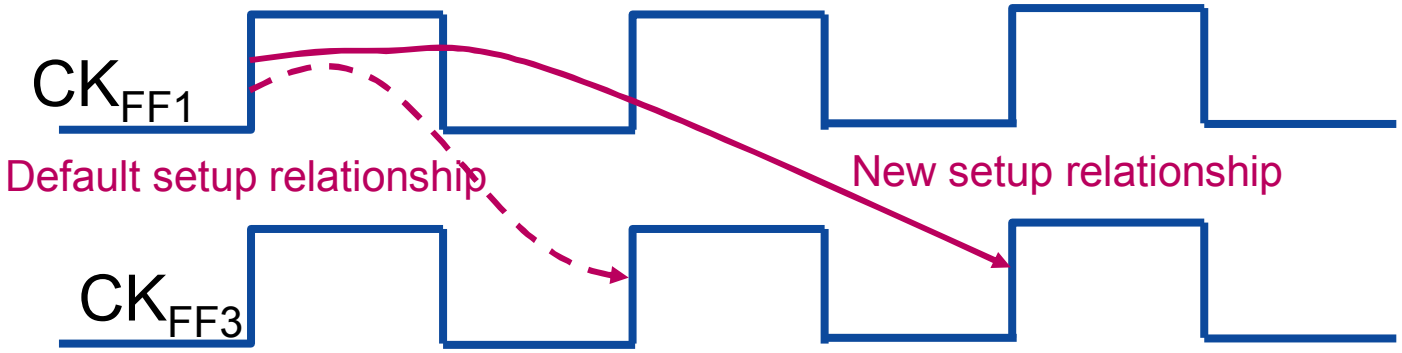
- Output Delay Specifies When the Arrives at the Input Ports of Another Device
  - Referenced to the Capture Edge Outside the FPGA



Timing Exceptions
Multi-cycle Paths

- Multi-cycle Paths Are Paths Where the Allowable Data Path Delay is More Than One Clock Cycle



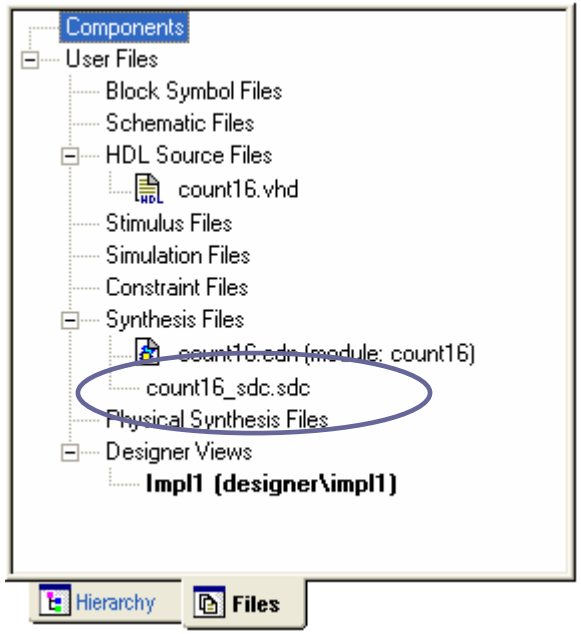


Timing Exceptions
False Paths

- False Paths are Design Paths That are Ignored for Optimization and Timing Analysis
- A Path May be False Because:
  - It Doesn't Propagate Data Forward
  - The Circuitry Involved Will Operate Slower Than The Constraints Indicate
    - For Example, Test Logic May Be Operated At A Lower Frequency

Synplicity SDC File

- Synplicity Generates an SDC Constraint File That can be Imported Into Designer
  - Constraint file is visible on Libero File Manager tab

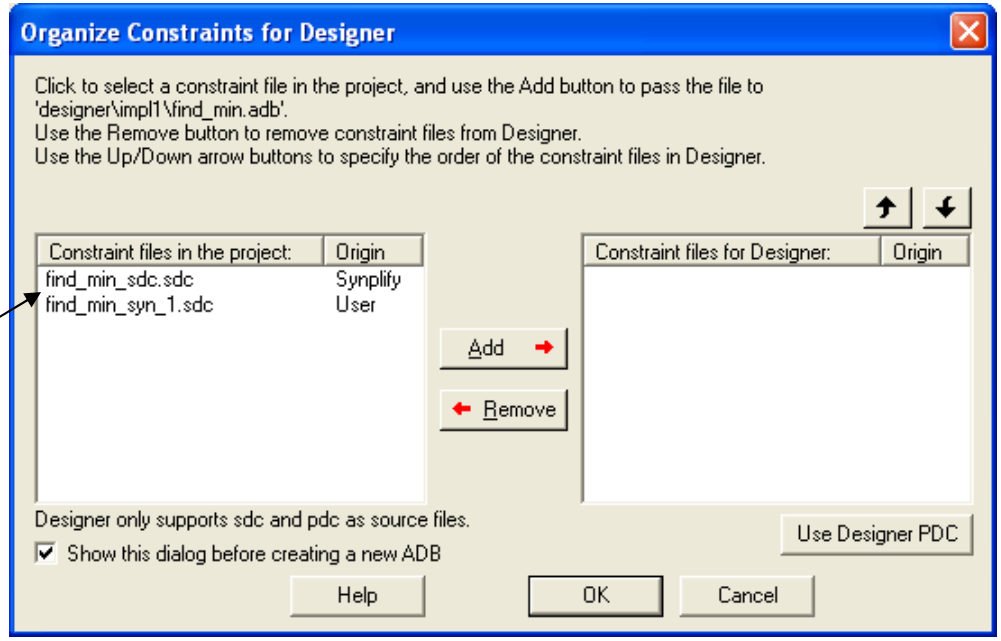



```
# Top Level Design Parameters
# Clocks
create_clock -period 10.000000 -waveform {0.000000 5.00000} CLK

# False Paths Between Clocks

# False Path Constraints

# Maximum Delay Constraints

# Multicycle Constraints

# Virtual Clocks
# Output Load Constraints
# Driving Cell Constraints
# Wire Loads
# set_wire_load_mode top

# Other Constraints
set_input_delay 3.500 -clock CLK {EN}
set_input_delay 3.500 -clock CLK {RST}
set_output_delay 4.000 -clock CLK {Q(0)}
```

Importing SDC Timing Constraints

- SDC Timing Constraints can be Imported into Designer as Source File From Libero
  - Libero Automatically Sends Timing Constraints to Designer
- Specify Constraint file from Libero Menu
  - Project > Files Organization > Designer Constraint files

Select <name>_sdc.sdc
then click Add
DO NOT select
<name>_syn.sdc



SmartTime Overview

- SmartTime is Actel's Static Timing Analysis Tool for:
  - ProASIC3, ProASIC3L, Fusion, IGLOO, APA and 500K
  - SX-A, RTSX-S, eX, AX, RTAX-S
  - All Future Products

SmartTime Features

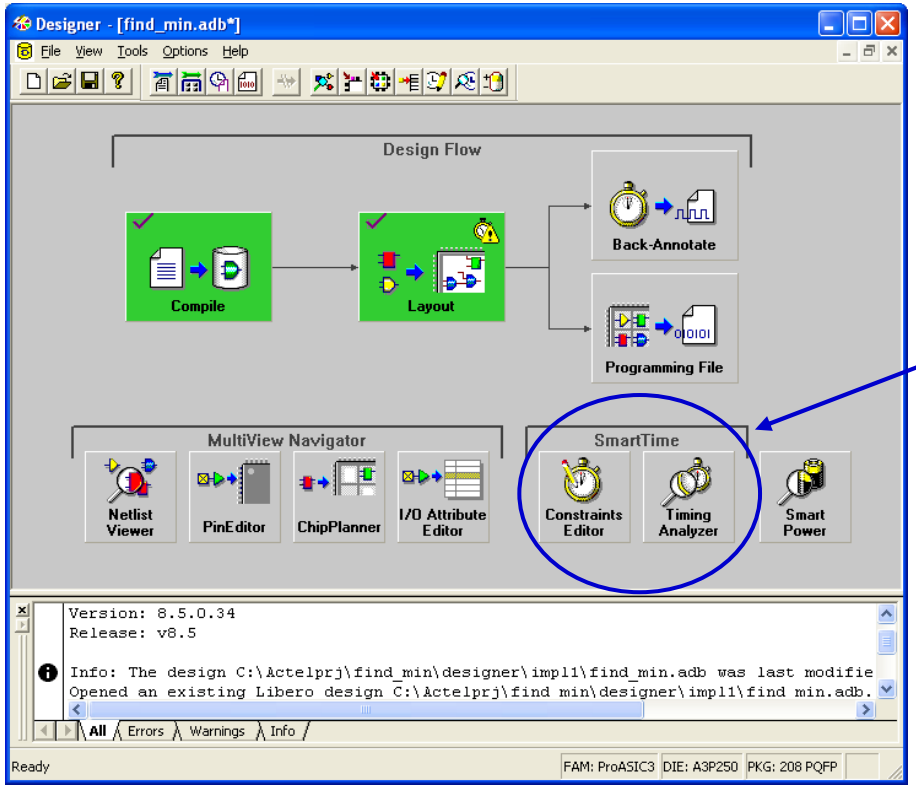
- Graphical Constraint Entry
- SDC Constraint Support
- SDC Constraint Checker
- Analysis
  - Cross-clock Domain Analysis
  - Flexibility in Clock Domain Selection
- Improved Ease-of-Use
  - Separate Constraint & Analysis Views
    - Separate Maximum & Minimum Analysis Views
    - Clock Domain Browser for the Analysis View
  - Visual Constraint Dialogs
  - Customizable Timing Information for the Paths List
  - Constraint Entry from the Analysis Window
  - Customizable Timing Reports
  - Filtering Capabilities
  - Persistence of User Settings

SmartTime
Supported SDC Constraints

- Timing Requirements and Exceptions can be Entered
  - From a Synopsys Design Constraint (SDC) File or
  - The SmartTime Visual Dialog GUIs
- SDC Constraints Supported in SmartTime:
  - **`create_clock`**
  - **`create_generated_clock`**
  - **`set_clock_latency`**
  - **`set_input_delay`**
  - **`set_output_delay`**
  - **`set_load`**
  - **`set_false_path –from –through –to`**
  - **`set_ multicycle_path –from –through –to`**
  - **`set_ maximum_delay –from –through –to`**
  - **`set_ minimum_delay –from –through –to`**

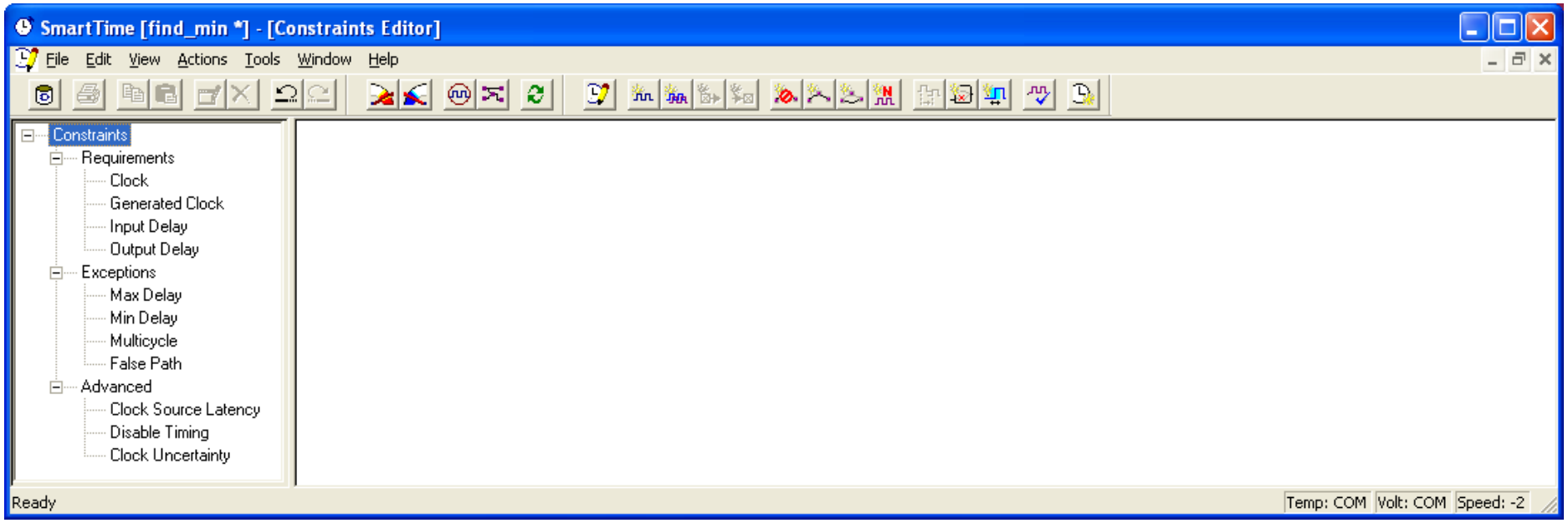
SmartTime Windows

- SmartTime Includes of Separate Constraint Editor and Timing Analysis Windows
  - Constraint Editor – GUI Based Constraint Entry
  - Timing Analyzer Provides Delay Information



Click to launch SmartTime Constraint Editor or Analysis Views

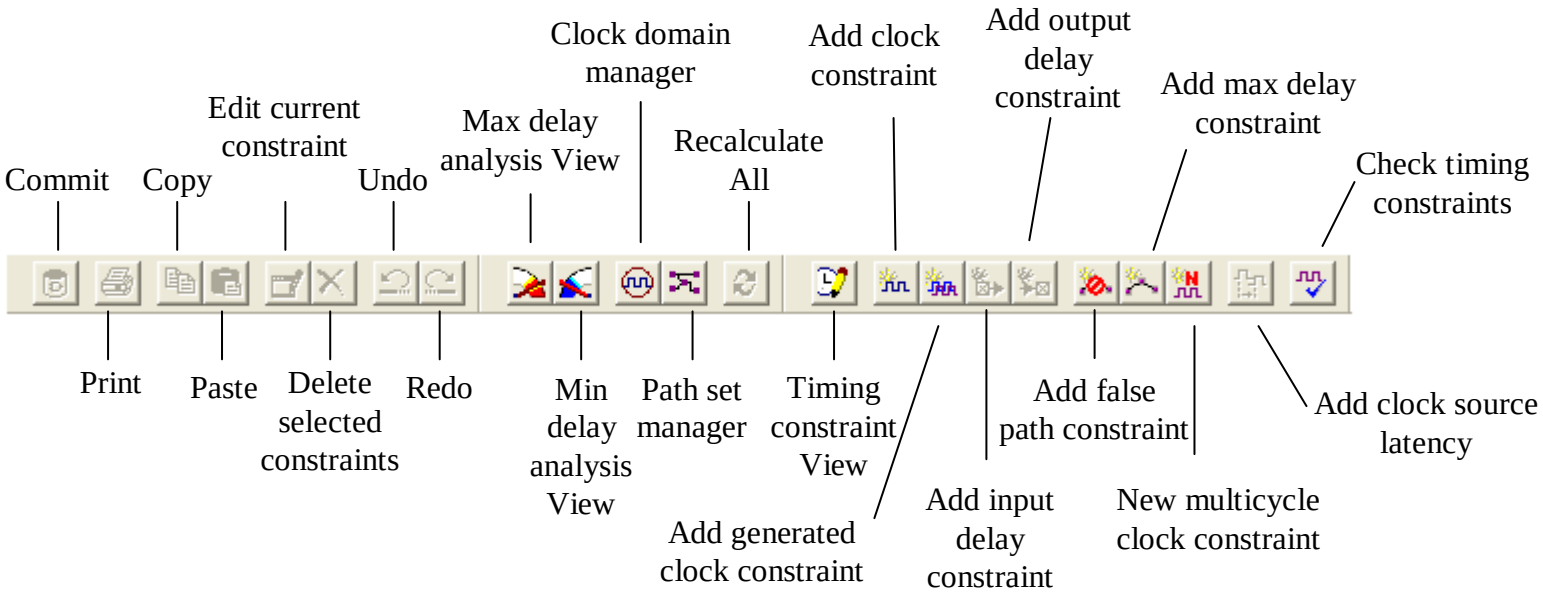
SmartTime
Constraints Editor



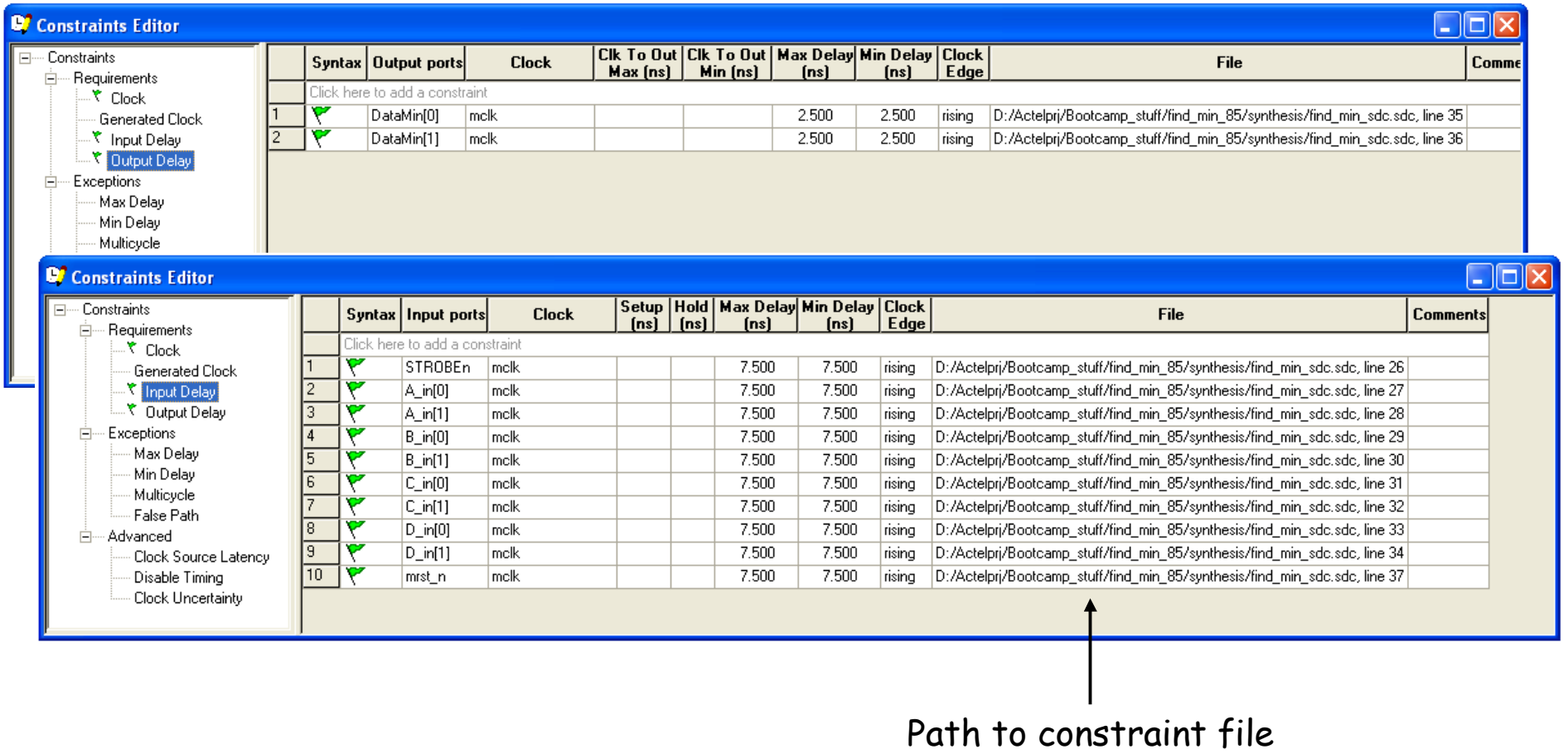
- Use Constraints Editor to:
  - Enter or Edit Timing Requirements:
    - Clock Frequency, Clock Latency, Input and Output Delays
  - Enter or Edit Timing Exceptions:
    - Max Delay, Multicycle Paths, False Paths
  - Browse Clock Domains
- Constraints are grouped as Requirements and Exceptions

SmartTime Toolbar

- The SmartTime Toolbar Contains Commands for Performing Common Operations.
  - Tool Tips Are Available for Each Button.



SDC Constraints in SmartTime



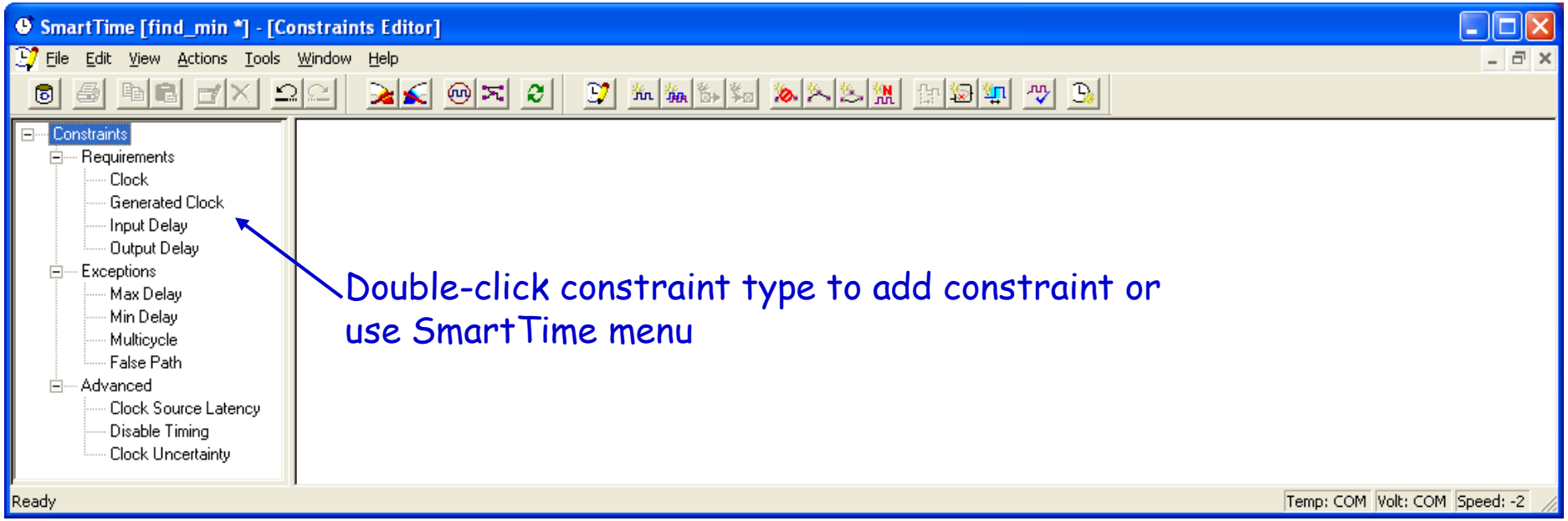
Clock Constraints

SmartTime Clock Domains

- SmartTime Detects Possible Clocks by Tracing Back from the Clock Pins of all Sequential Components Until it Finds:
  - An Input Port
  - The Output of Another Sequential Element, or
  - The Output of a PLL
- SmartTime Classifies Clock Sources Into Three Types:
  - Explicit Clocks
  - Potential Clocks
  - Clock Network
- Each Clock Domain Contains at Least 3 Path Sets:
  - Register to Register
  - External Setup or Hold
  - Clock to Out

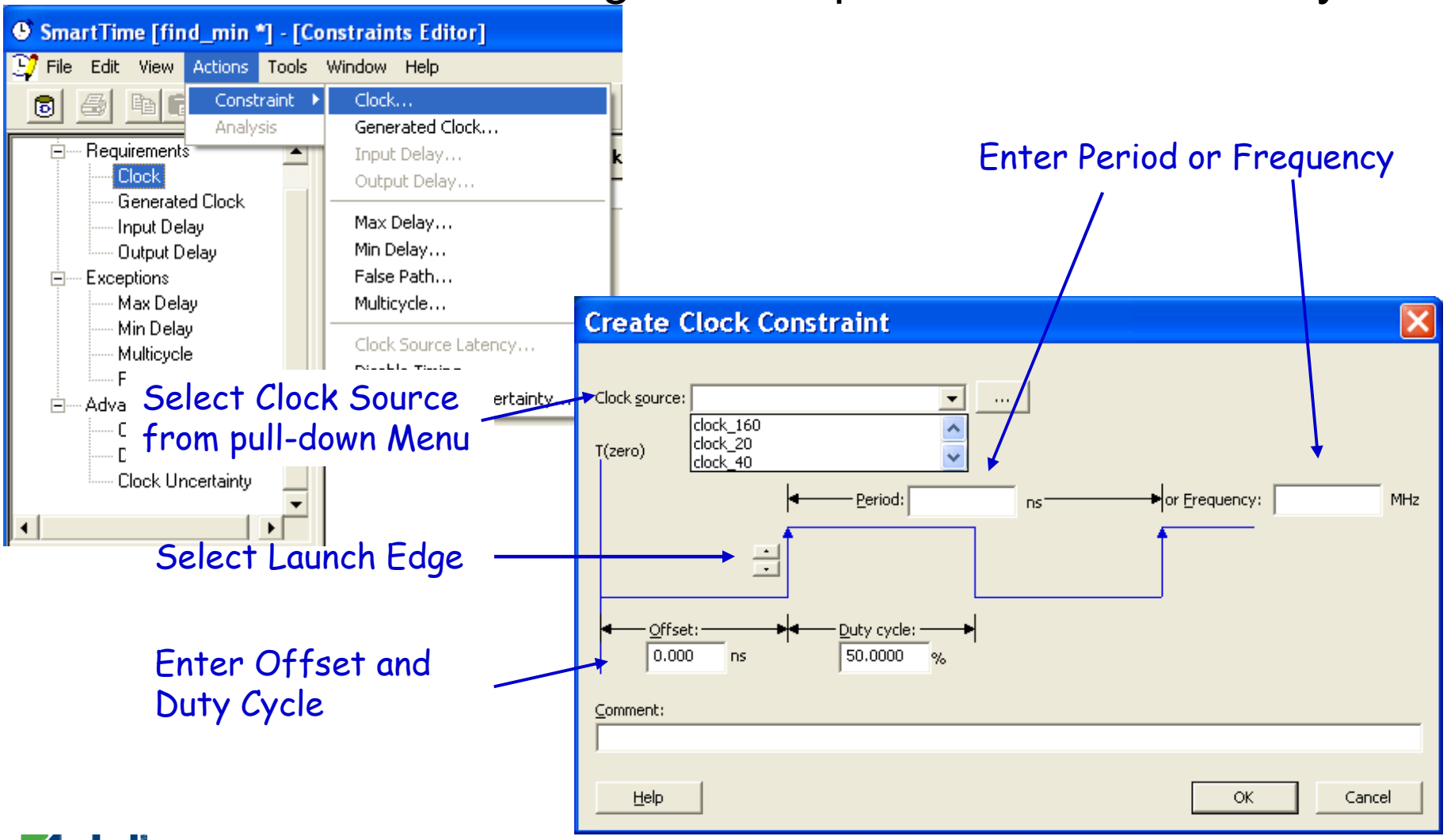
SmartTime Constraint Entry

- Enter Timing Requirements Prior To Layout For Timing Analysis And To Guide Layout Engine
  - Clock Frequency, Input and Output Delays
- Timing Exceptions Can Be Entered After Layout If Negative Slack Exists To Ensure Correct Timing Analysis
  - Max Delay, Multicycle Paths, False Path



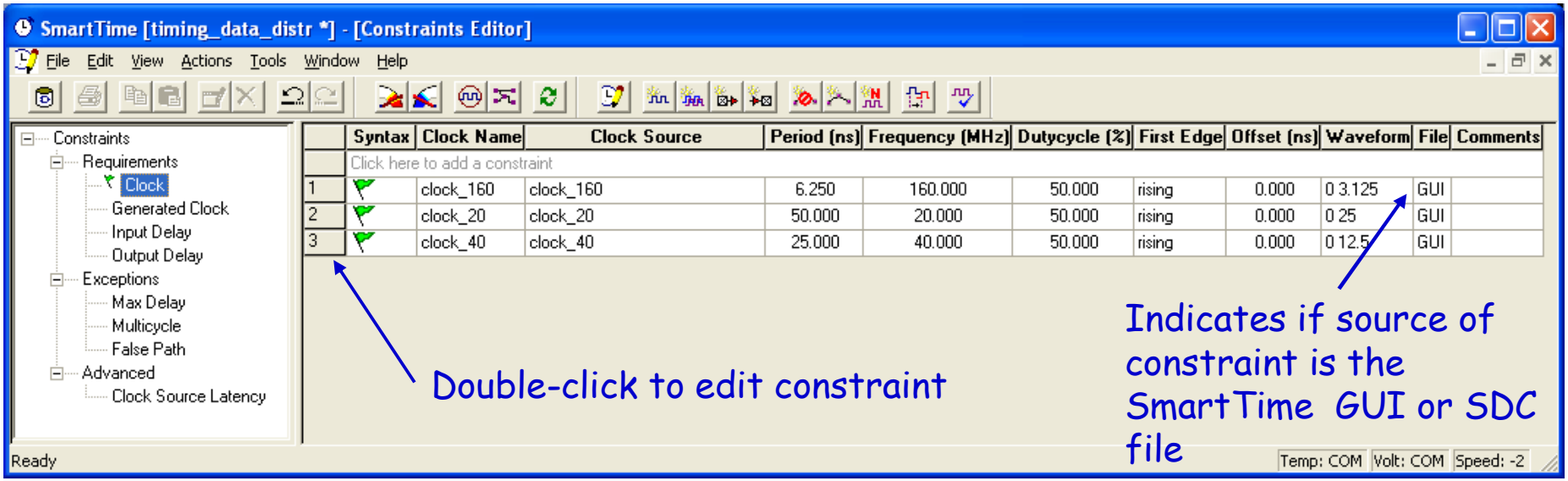
Entering Clock Constraints

- Visual Constraint Dialog Box Simplifies Constraint Entry



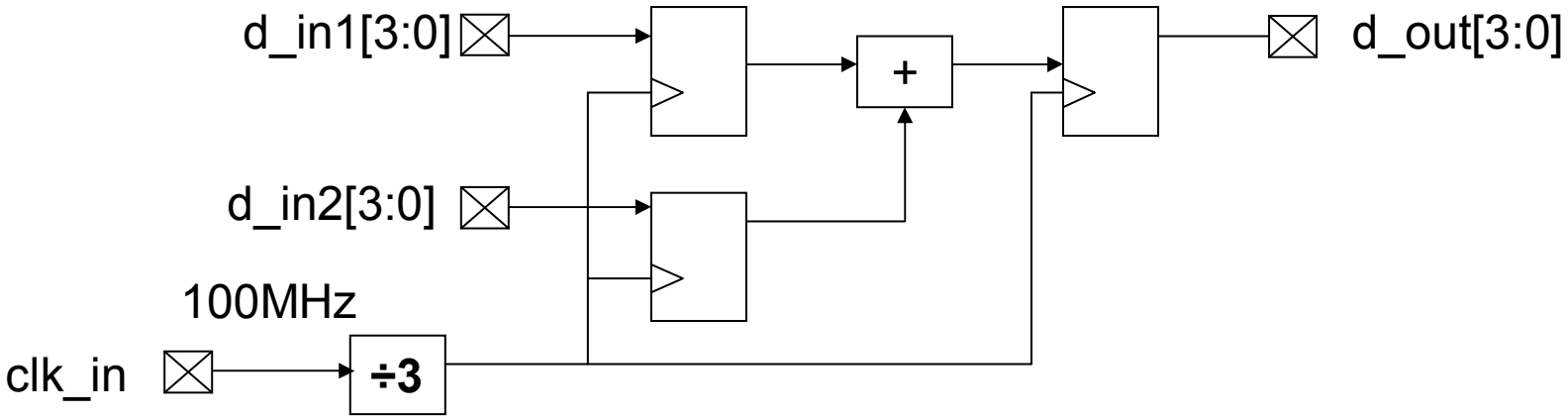
SmartTime Clock Constraints

- Clock Constraints in Constraint Editor
  - Select Constraint to Add, Edit or Delete



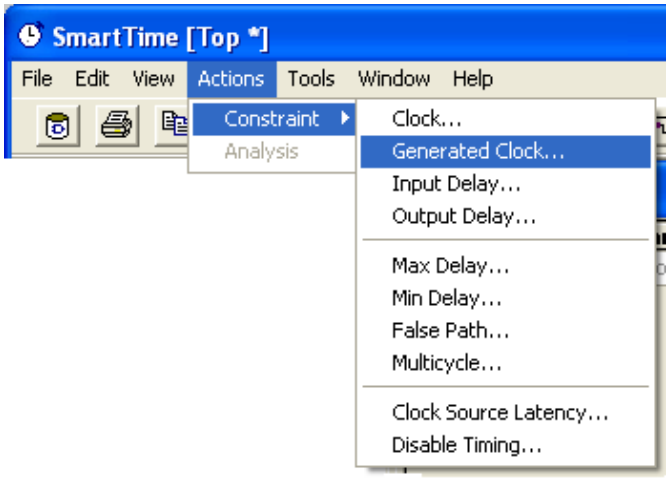
SmartTime Generated Clock Constraints

- Constraints For Internally Generated Clock Can Be Specified Using Generated Clock Constraints
  - Useful For Constraining Designs with Clock Dividers



Entering Generated Clock Constraints

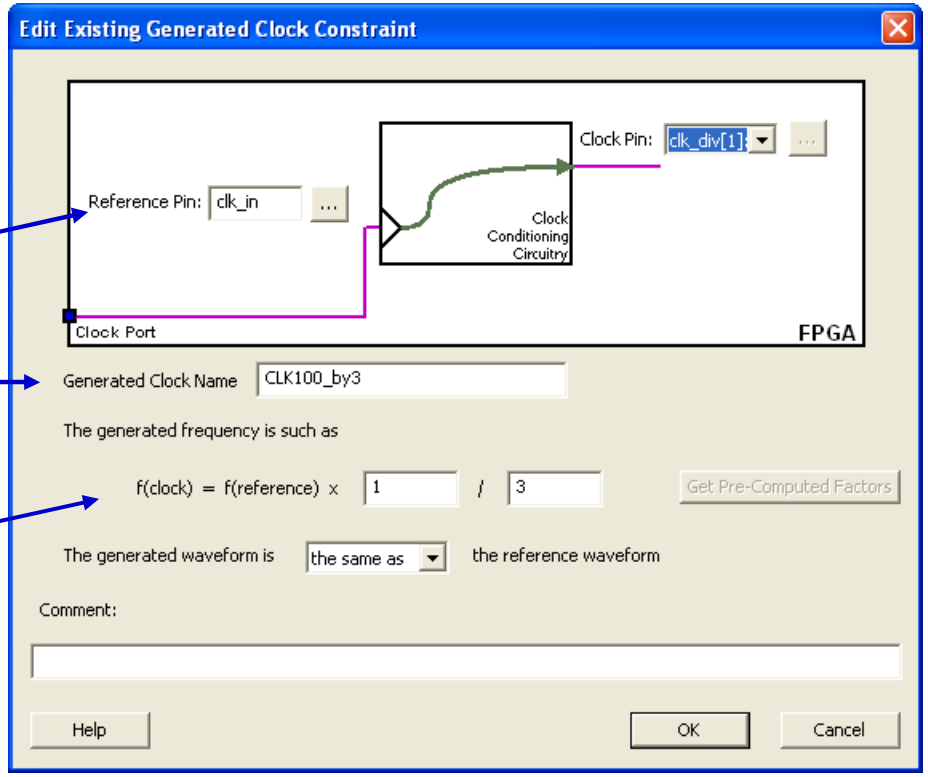
- Visual Constraint Dialog Box Simplifies Constraint Entry



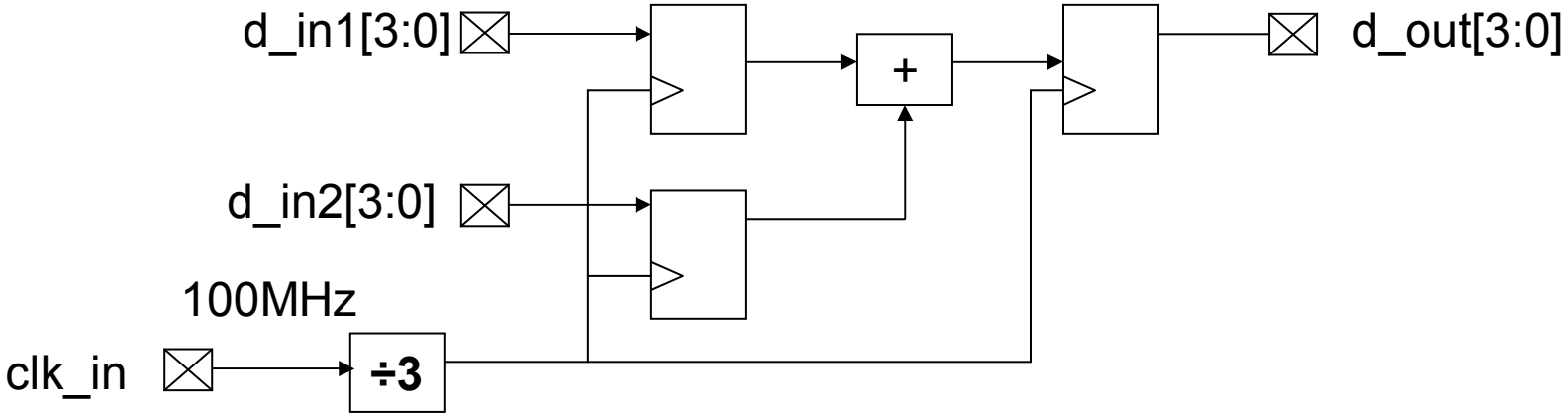
Select Reference Clock from pull-down Menu

Enter clock name

Enter Multiplier and Divider



SmartTime Generated Clocks



Constraints Editor

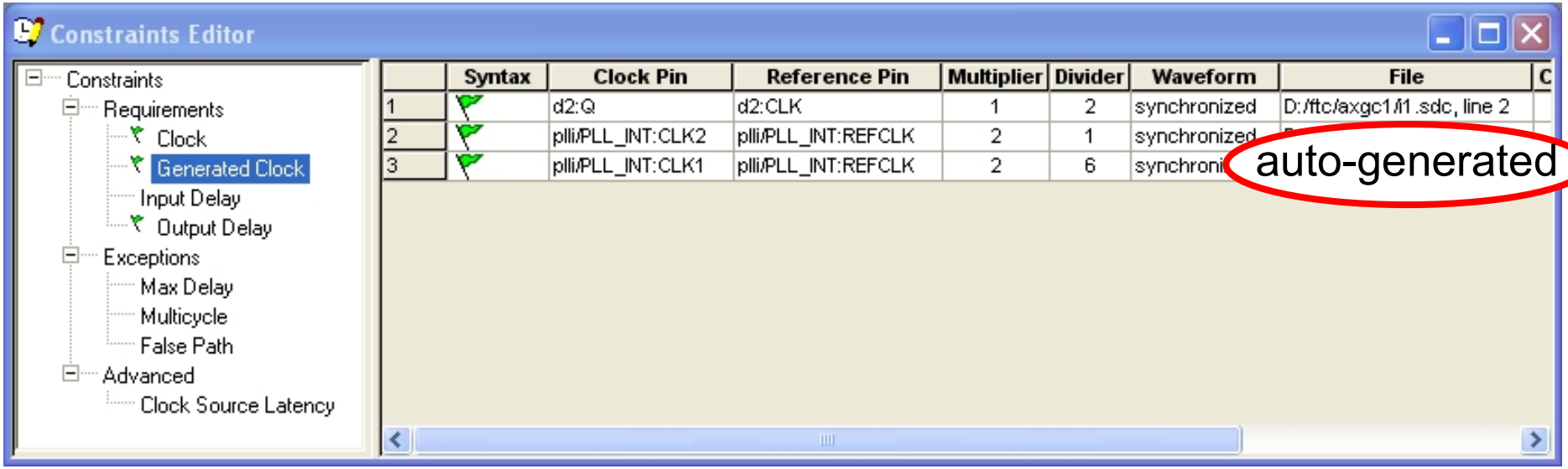
- Constraints
  - Requirements
    - Clock
    - Generated Clock
    - Input Delay
    - Output Delay
  - Exceptions
    - Max Delay
    - Multicycle
    - False Path
  - Advanced
    - Clock Source Latency

	Syntax	Clock Name	Clock Pin	Reference Pin	Multiplier	Divider	Waveform	File	Comments
	Click here to add a constraint								
1		CLK100_by3	clk_div[1]:Q	clk_in	1	3	synchronized	GUI	

SmartTime

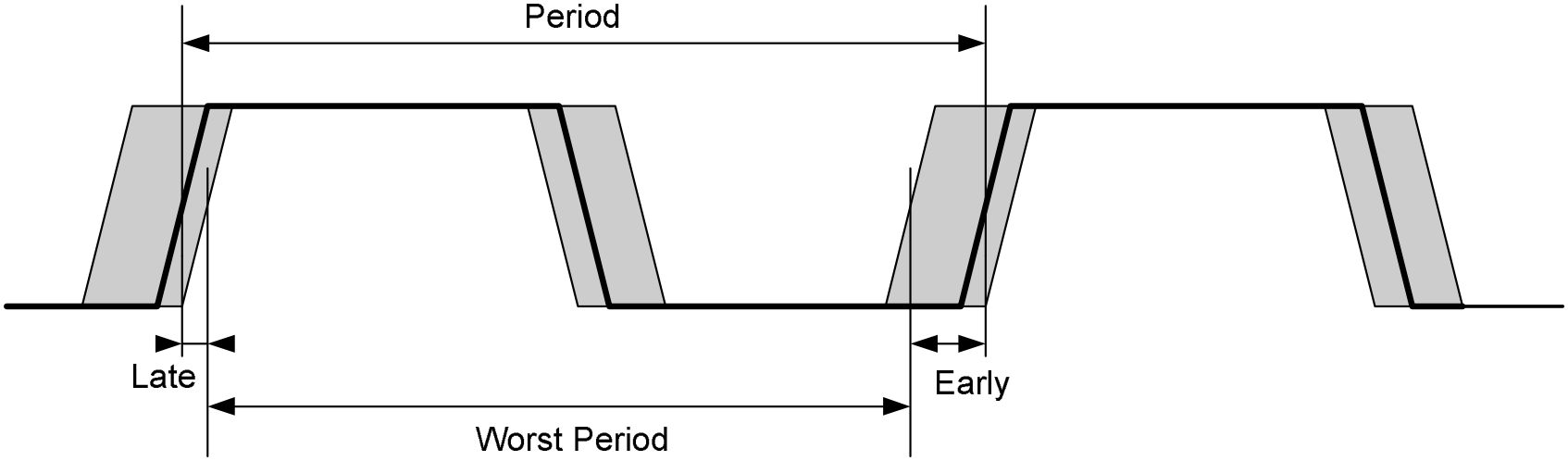
Automatic Creation of Generated Clocks

- Automatic Creation of Generated Clock Constraint Based on Static Configuration of PLLs for Fusion, ProASIC3, IGLOO, AX, RTAX-S
- No Effect Unless the Reference Clock is Constrained



Clock Source Latency:
Definition

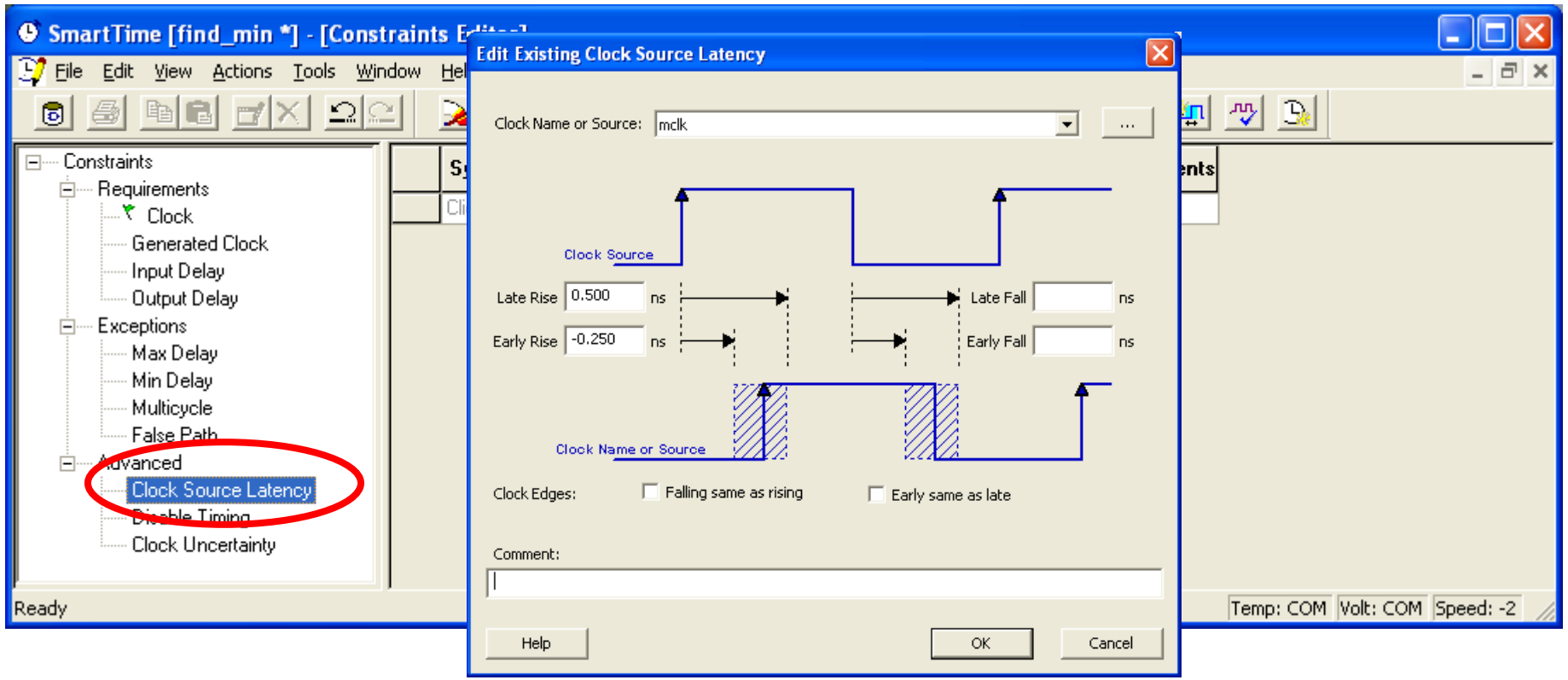
- Specify Delay When Clock is Generated Outside the Design
- Specify Clock Jitter



- Clock Source Latency can be Attached to any Clock Constraint (Internal or External)

Clock Source Latency:
Constraint Specification

- Specify Clock Insertion Delay and Jitter
- Specify Early and Late Times for Rising and Falling Edges



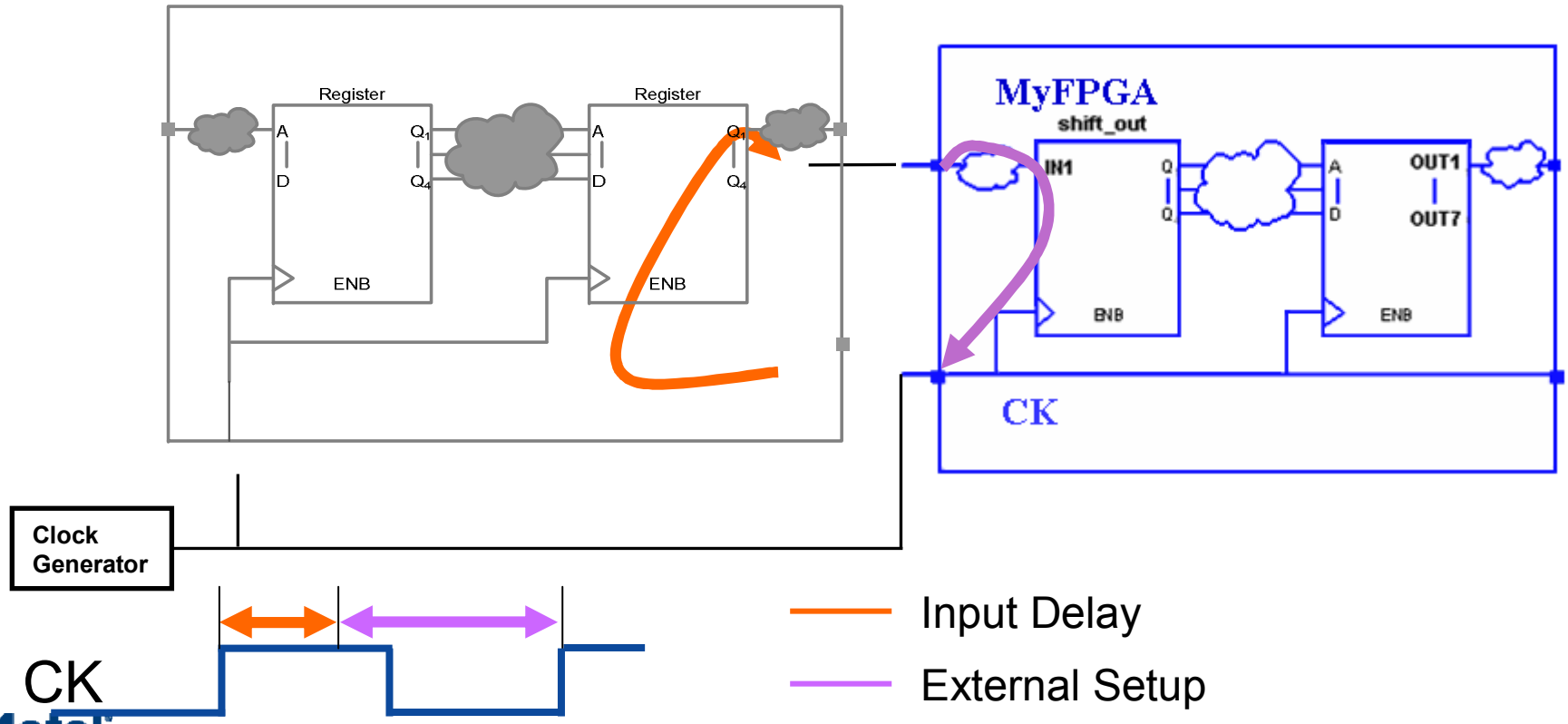
SmartTime
Virtual Clocks

- A Virtual Clock has no Associated Source
  - Used to describe clocks outside the FPGA that have an impact on timing analysis inside the FPGA
- Clock Analysis for Virtual Clocks
  - Specify a constraint for a clock that is running outside the FPGA and do timing analysis on the design as it interacts with this “virtual” circuitry

Input and Output Delay Constraints

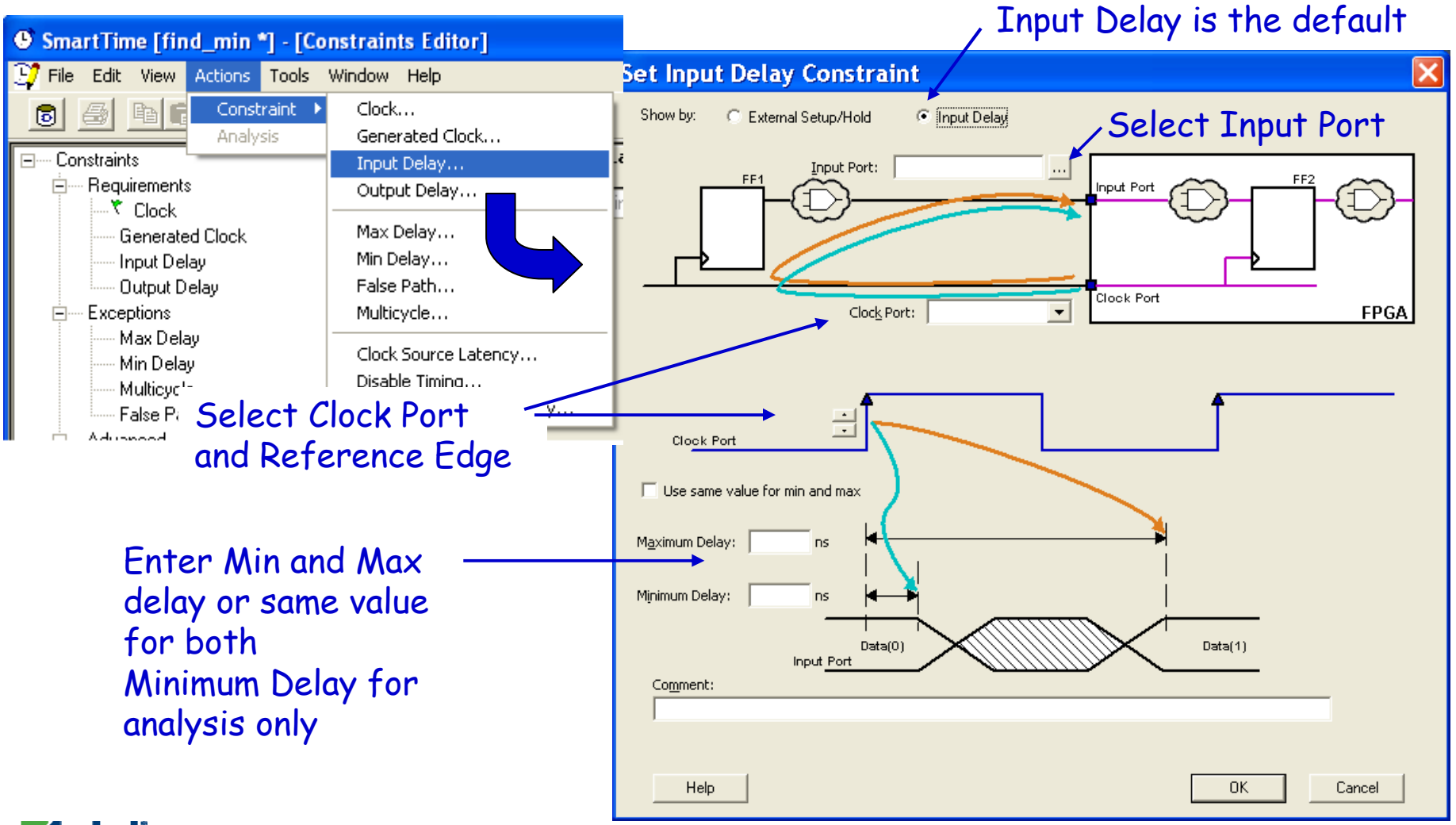
SmartTime
Input Timing Constraints

- Input Timing Constraints can be Entered as Input Delay or External Setup/Hold
  - Input Delay – Delay in Terms of Delay Budget *Outside* FPGA (Default)
    - Referenced from the *Launch Edge* Outside the FPGA
    - Generally Used by Synthesis Tools
  - External Setup / Hold – Specify Input Delay Budget *Inside* FPGA
    - Referenced to the *Capture* Edge Inside the FPGA
    - Used by Previous Actel Timer

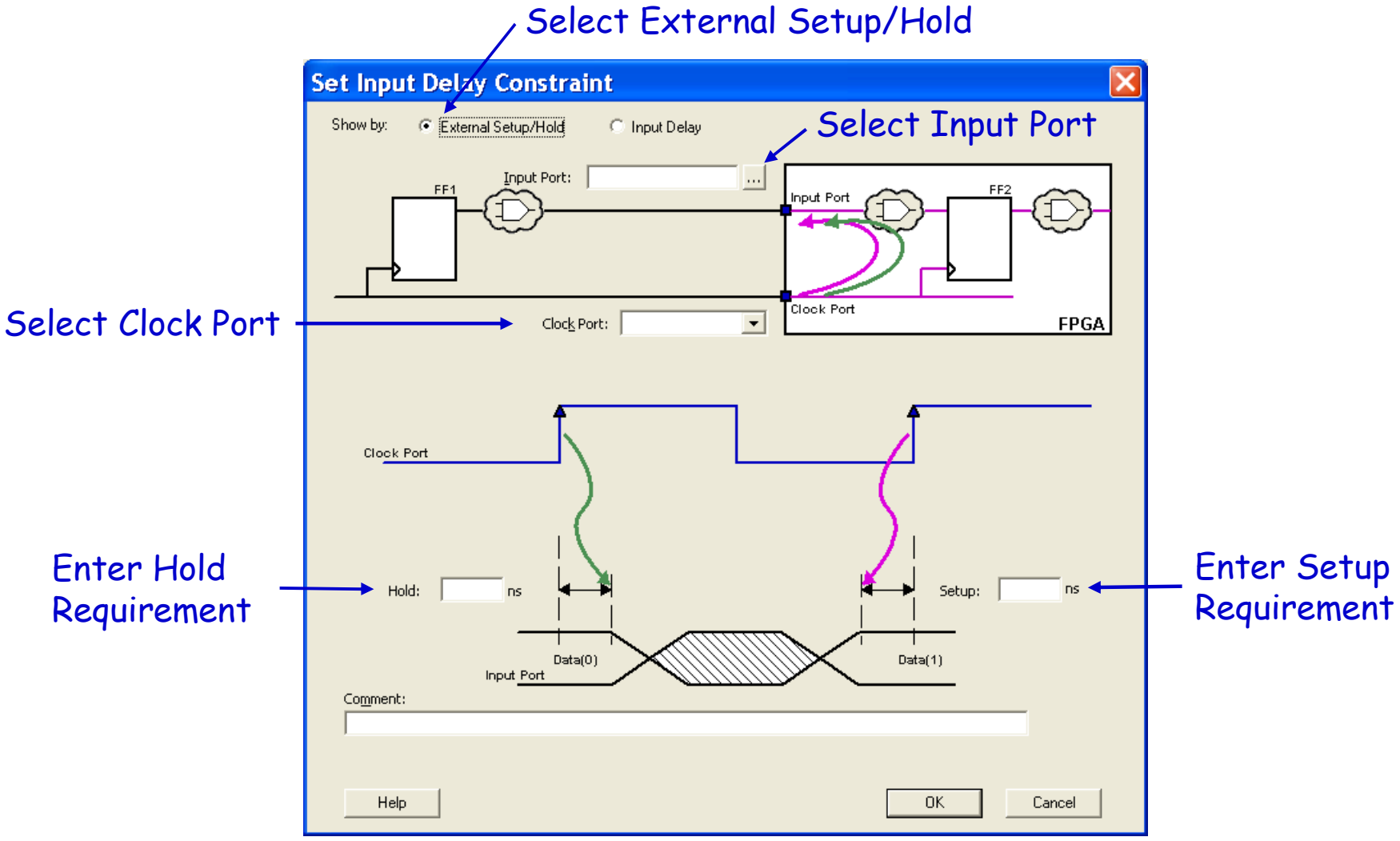


Input Constraints
Input Delay

- Visual Constraint Dialog Box Simplifies Constraint Entry

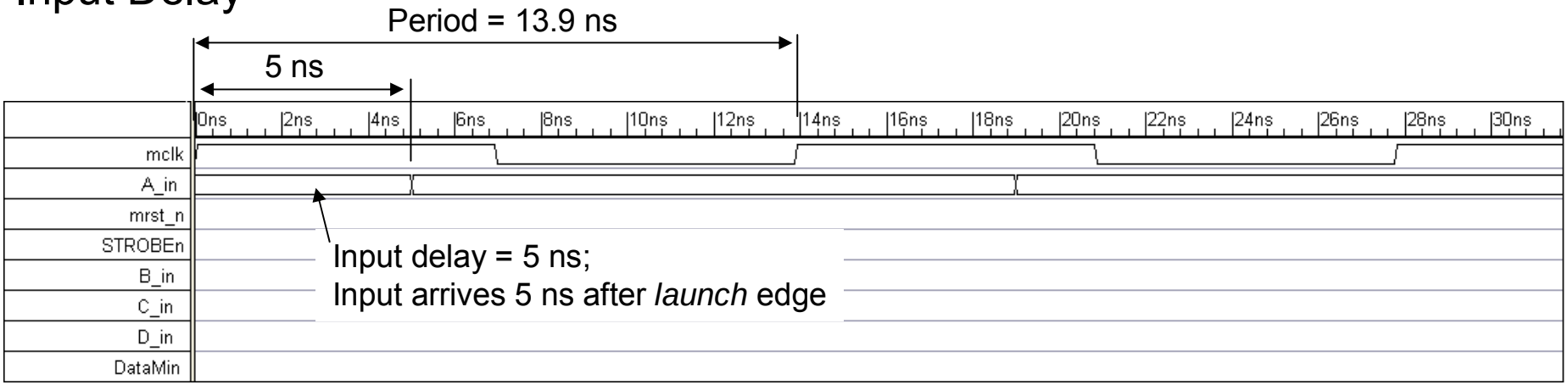


Input Delay Constraints
External Setup / Hold

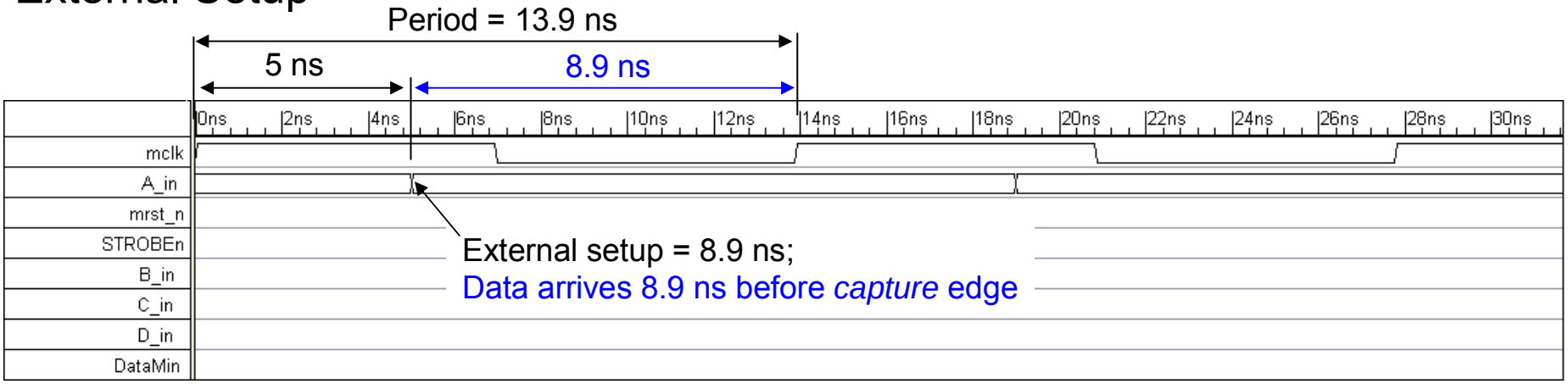


Input Delay / External Setup
What do These Constraints Mean?

Input Delay

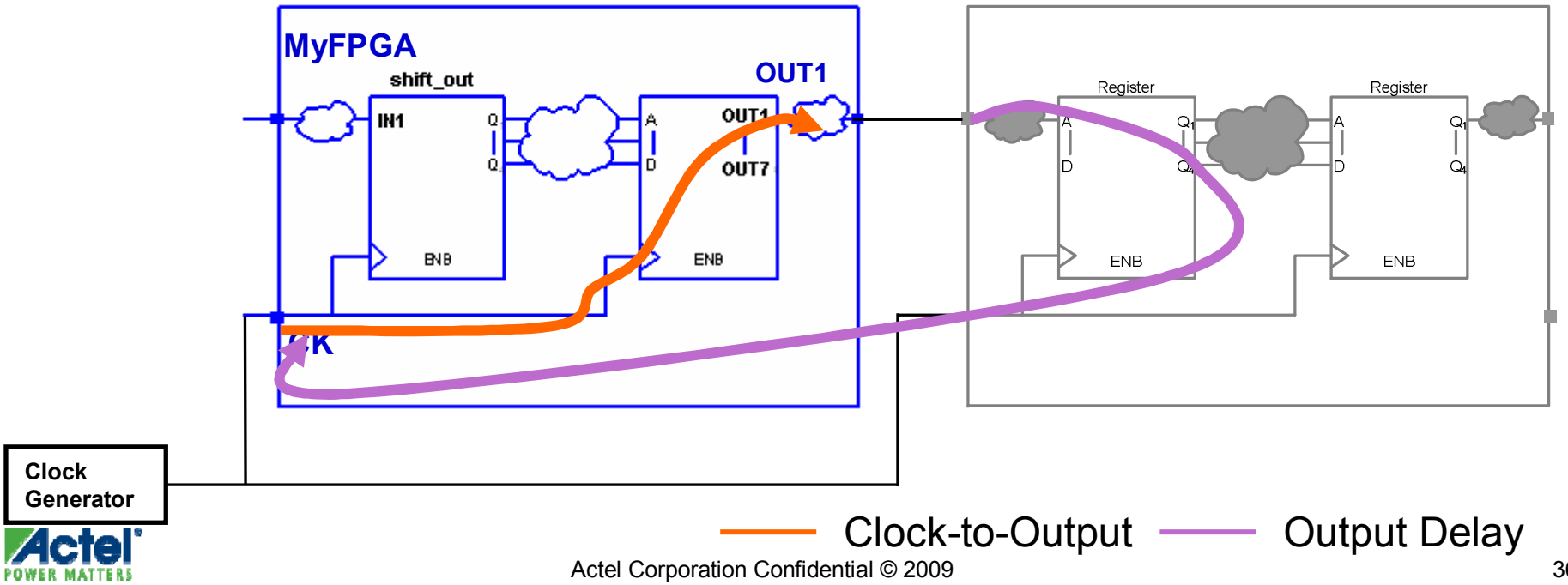


External Setup



SmartTime
Output Delay Constraints

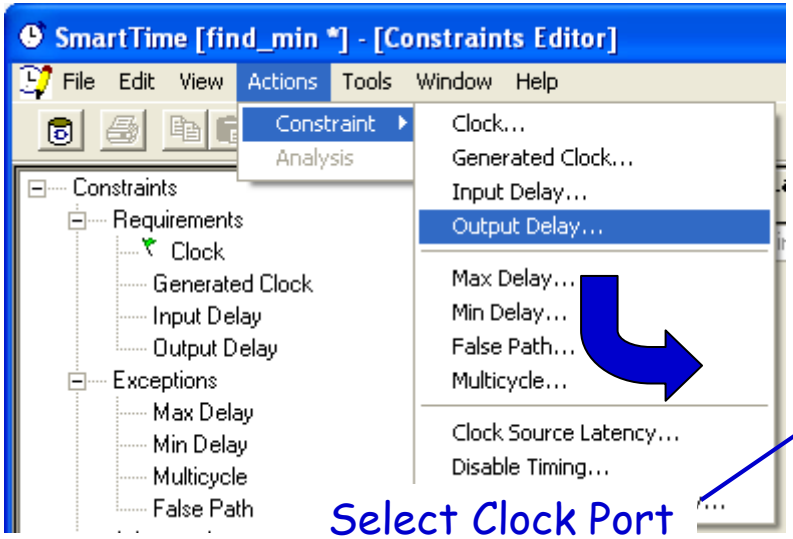
- Output Delay Defines the Delay of an Output Relative to a Clock
  - Clock-to-Output: Specifies the Output Delay Timing Budget *Inside* the FPGA (default)
    - Referenced from the *Launch* Edge Inside the Current Design
  - Output Delay: Specifies Output Delay Timing Budget *Outside* the FPGA
    - Referenced to the *Capture* Edge Outside the Current Design
    - Used by Synthesis tools



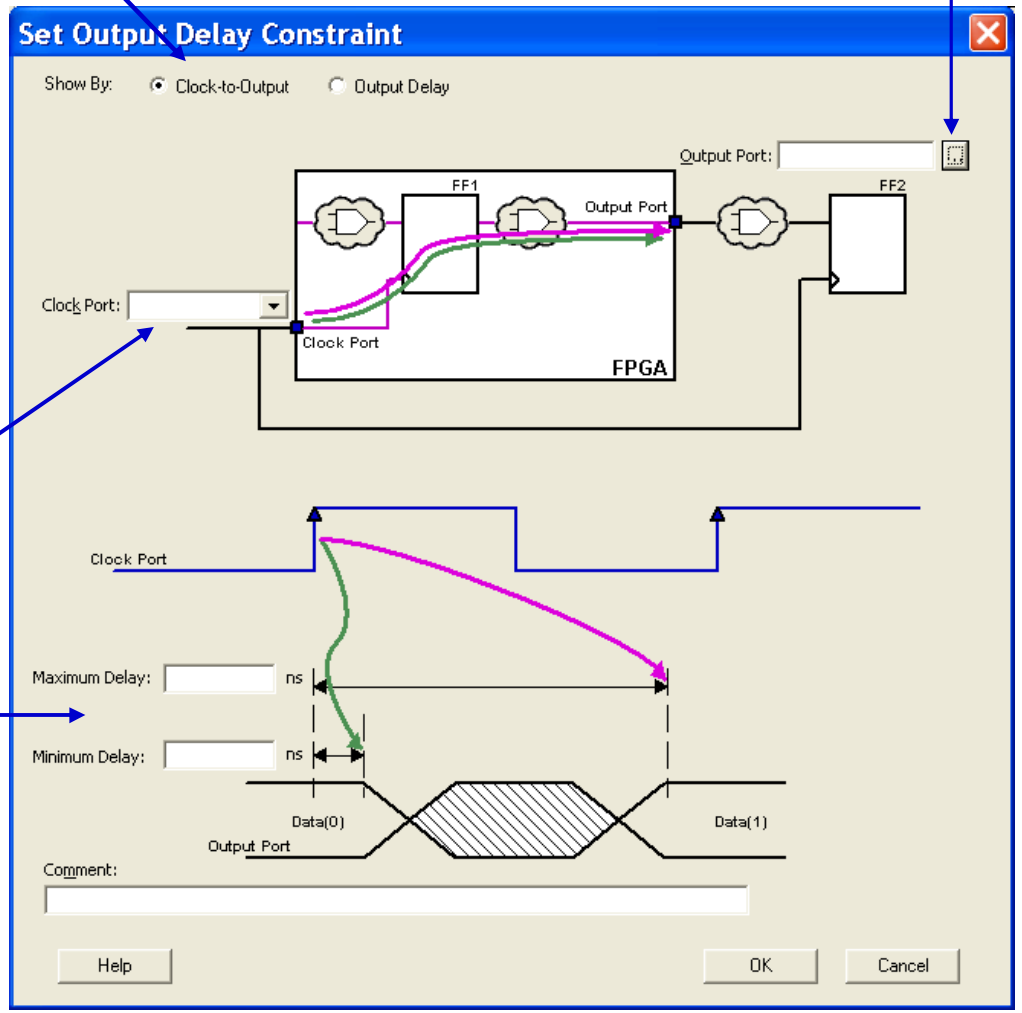
Output Constraints
Clock-to-Output

Clock-to-Output is the Default

Select Output Port



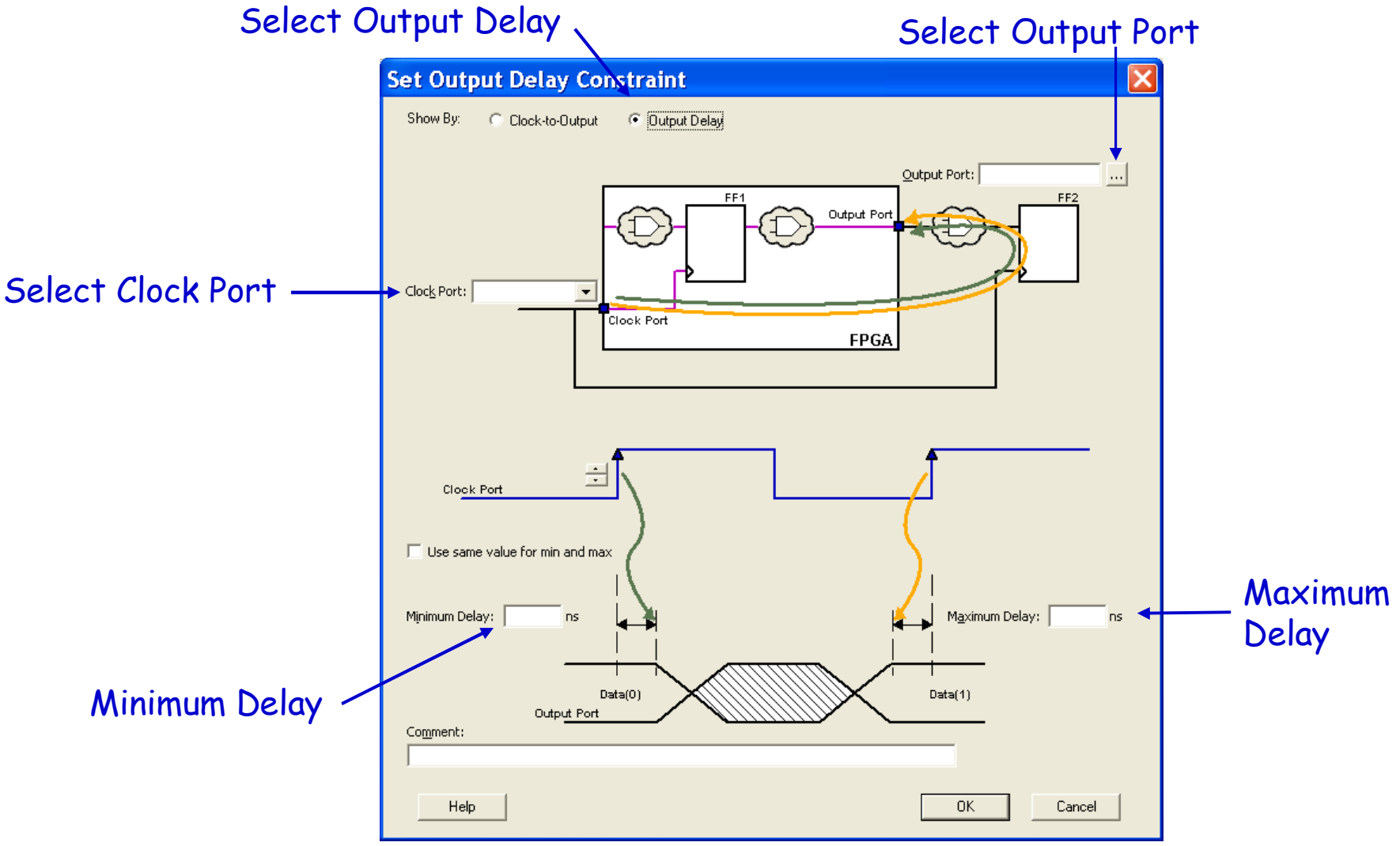
Select Clock Port



Enter Min and Max delay
Minimum Delay for analysis only

Output Delay Constraints

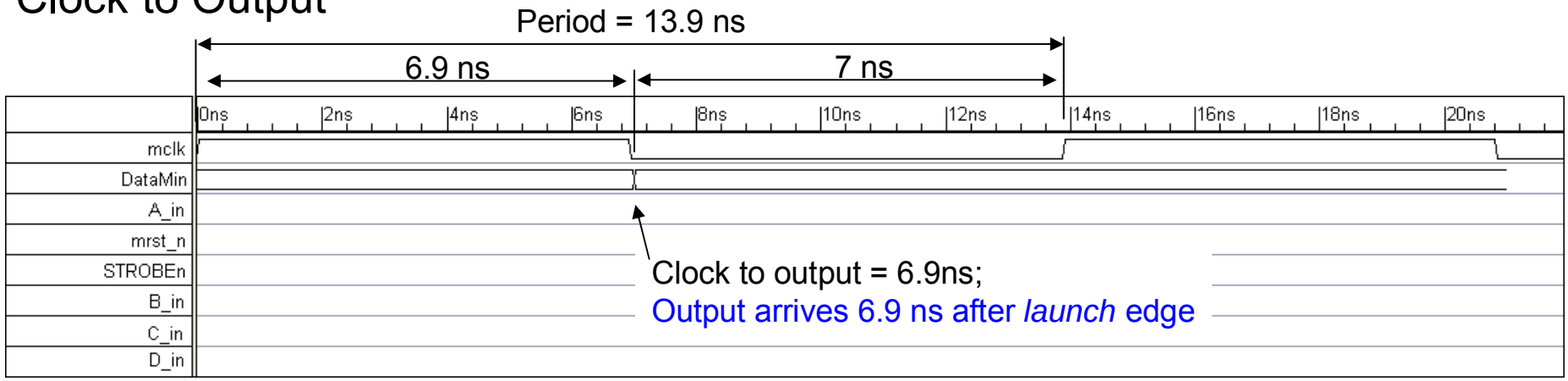
Output Delay



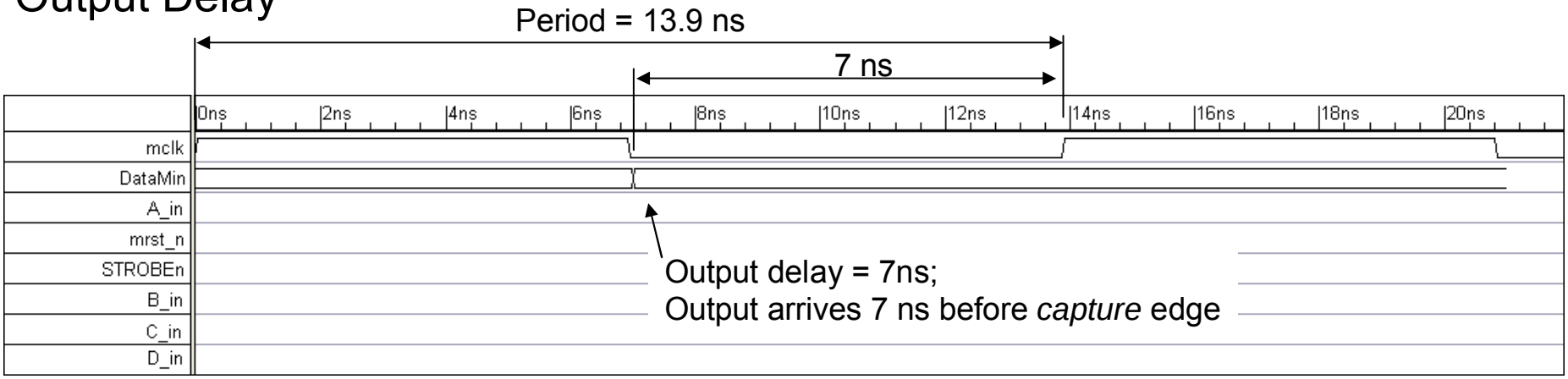
Clock to Output / Output Delay

What do These Constraints Mean?

Clock to Output



Output Delay



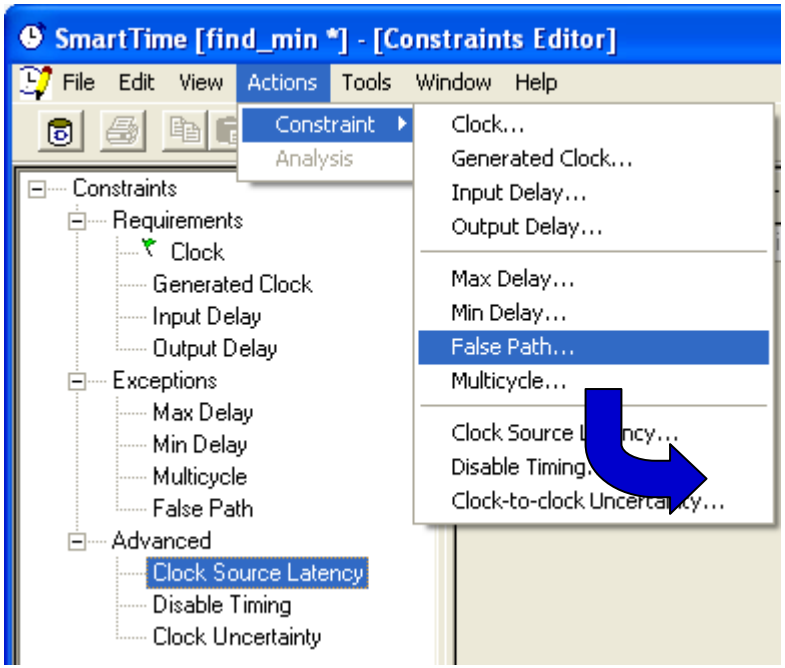
Timing Exceptions

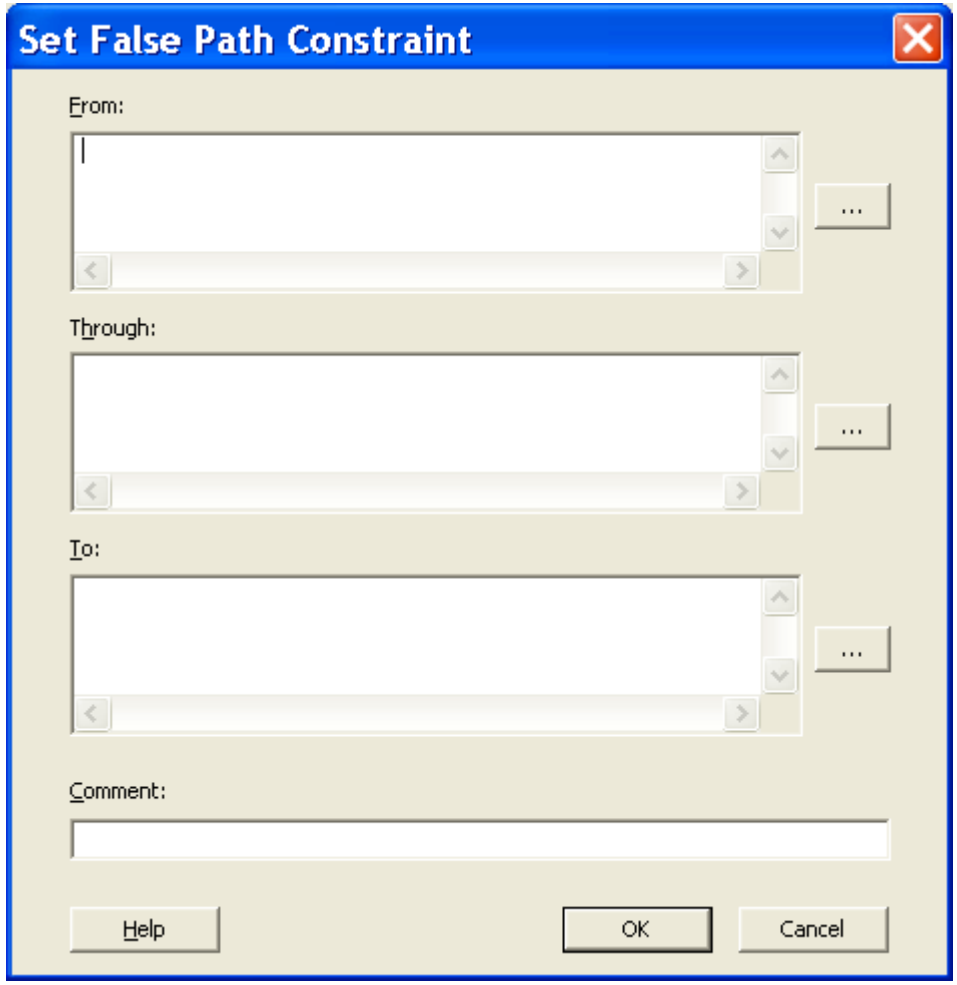
Timing Exceptions
False Paths

- The False Path Constraint Removes Timing Requirements on Specified Paths
  - Starting Points Are Input Ports or Register Clock Pins
  - Ending Points Are the Register Data Pins or Output Ports
- False Paths Are Not Considered During the Timing Analysis
- False Path Constraints Take Precedence Over Multiple Cycle Path Constraints and Overrides Maximum Delay Constraints

SmartTime
Entering False Path Constraints

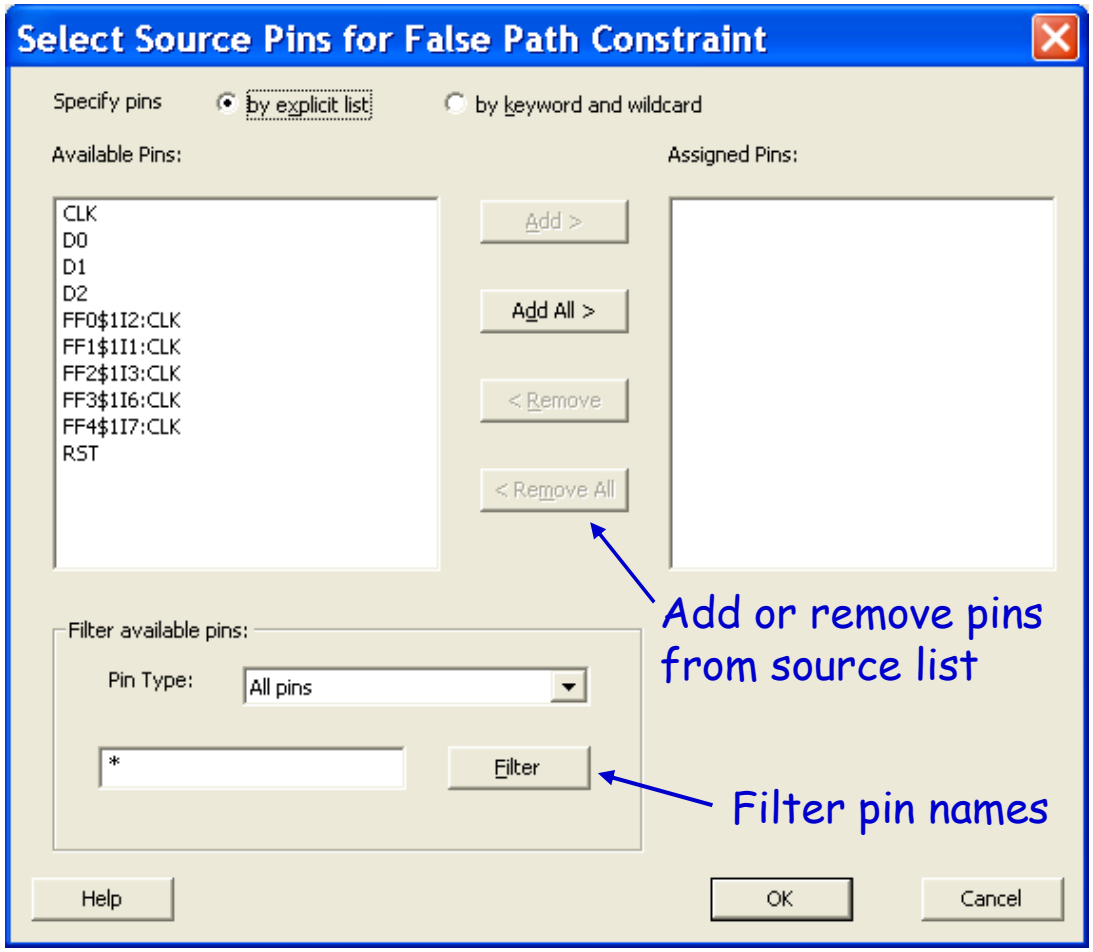
- Specify From, To, Through Pins





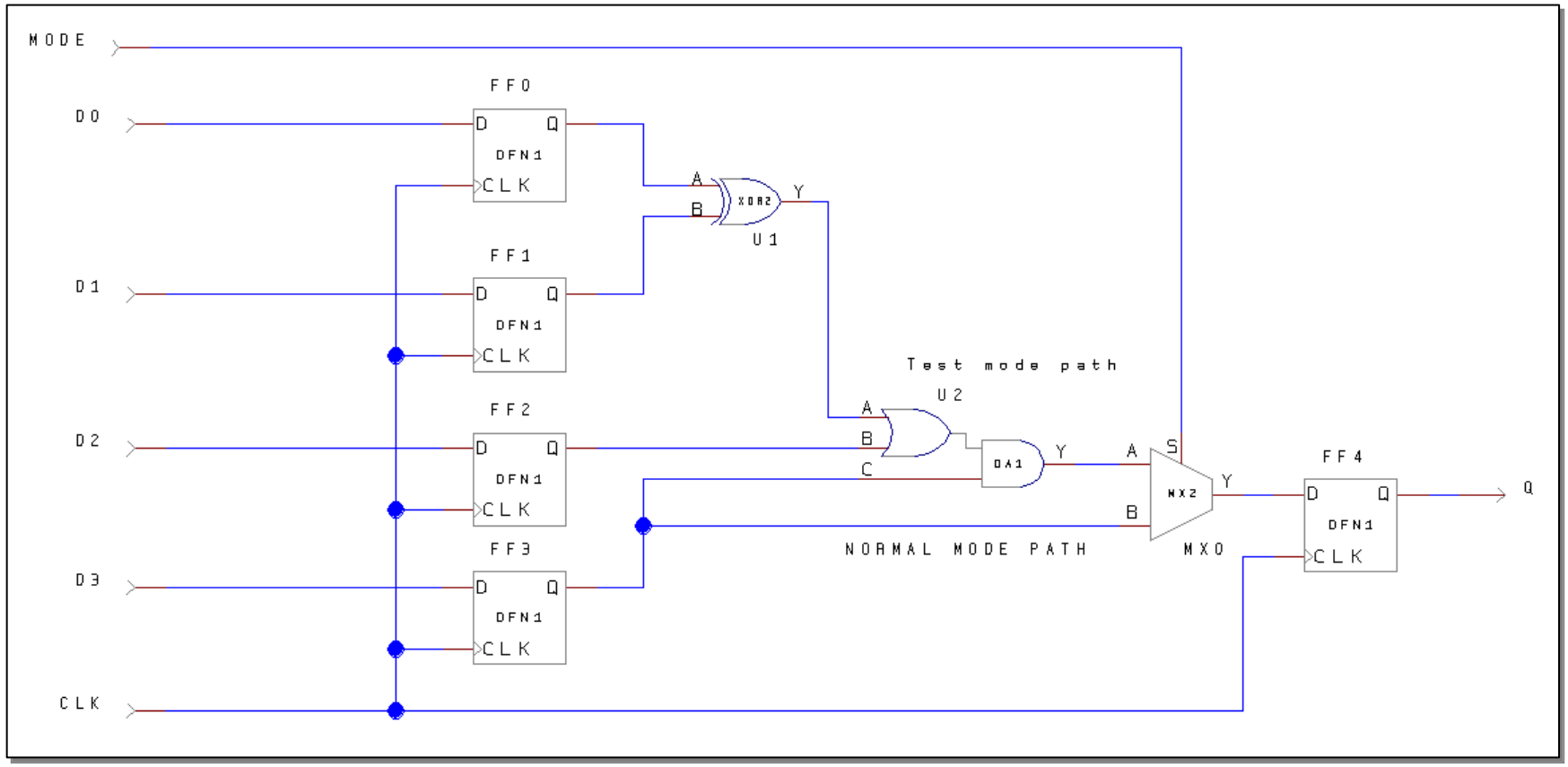
False Path Constraint
Source Pins

- Specify Starting Point for False Path



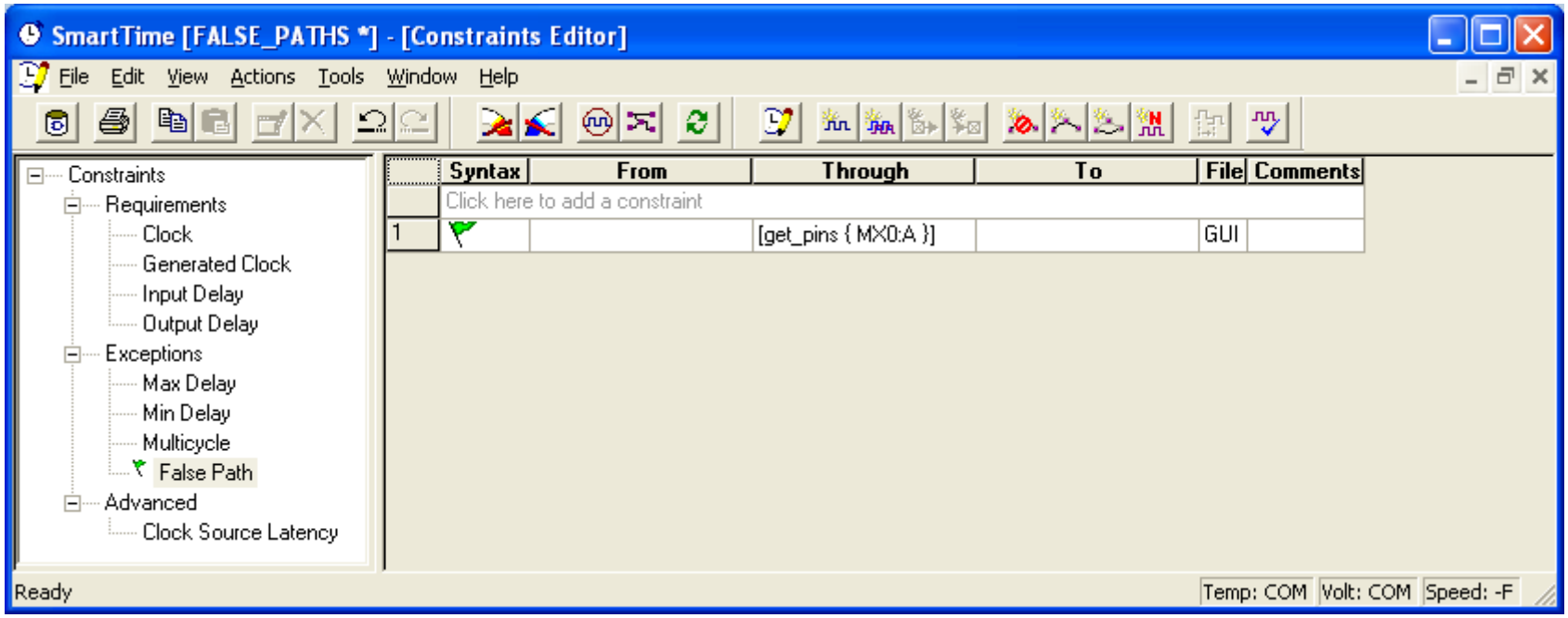
False Path Example

- Multiplexer S Input Selects Test / Normal Modes
  - Paths through MX0:A are False Paths



False Paths in Constraint Editor

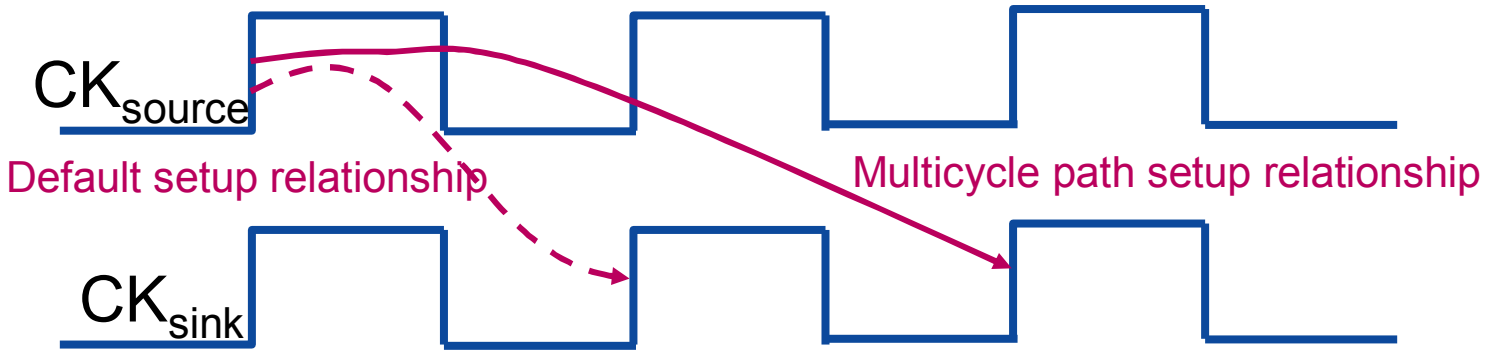
- False Path Constraint in Constraint Editor
  - Select Constraint to Add, Edit or Delete



Timing Exceptions

Multicycle Paths

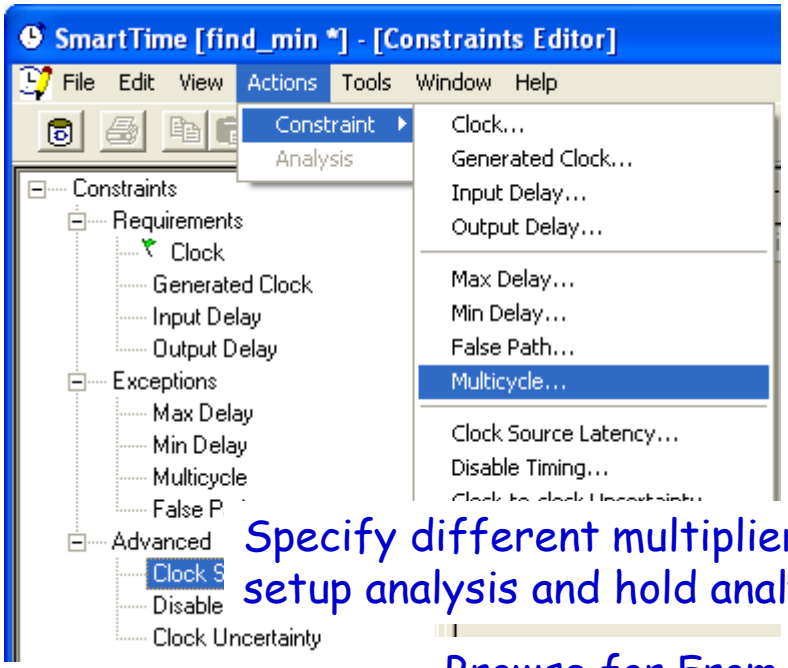
- The Multicycle Path Constraint Overrides the Single Cycle Timing Requirement Between Sequential Elements
  - The Number of Clock Cycles for Setup or Hold Check Is Specified in the Constraint
- False Path Constraints Take Precedence Over Multicycle Path Constraints



SmartTime

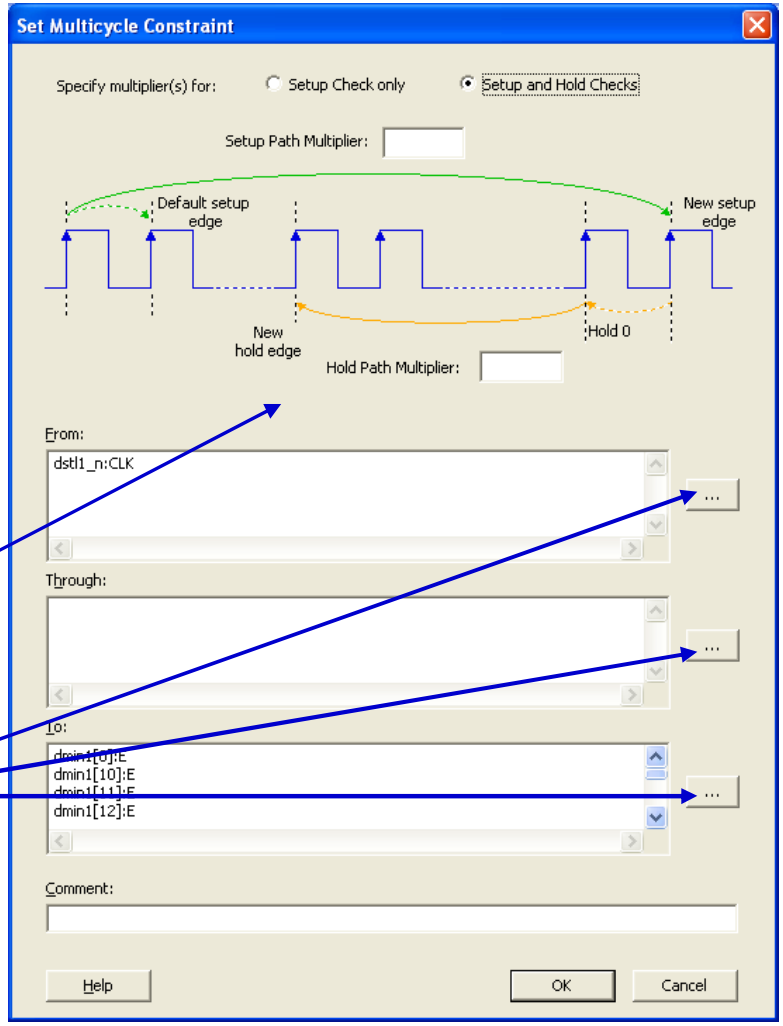
Entering Multicycle Path Constraints

- Specify Path Multiplier and From, To, Through Pins
- Add Path Multiplier



Specify different multiplier for setup analysis and hold analysis

Browse for From, Through or To pins

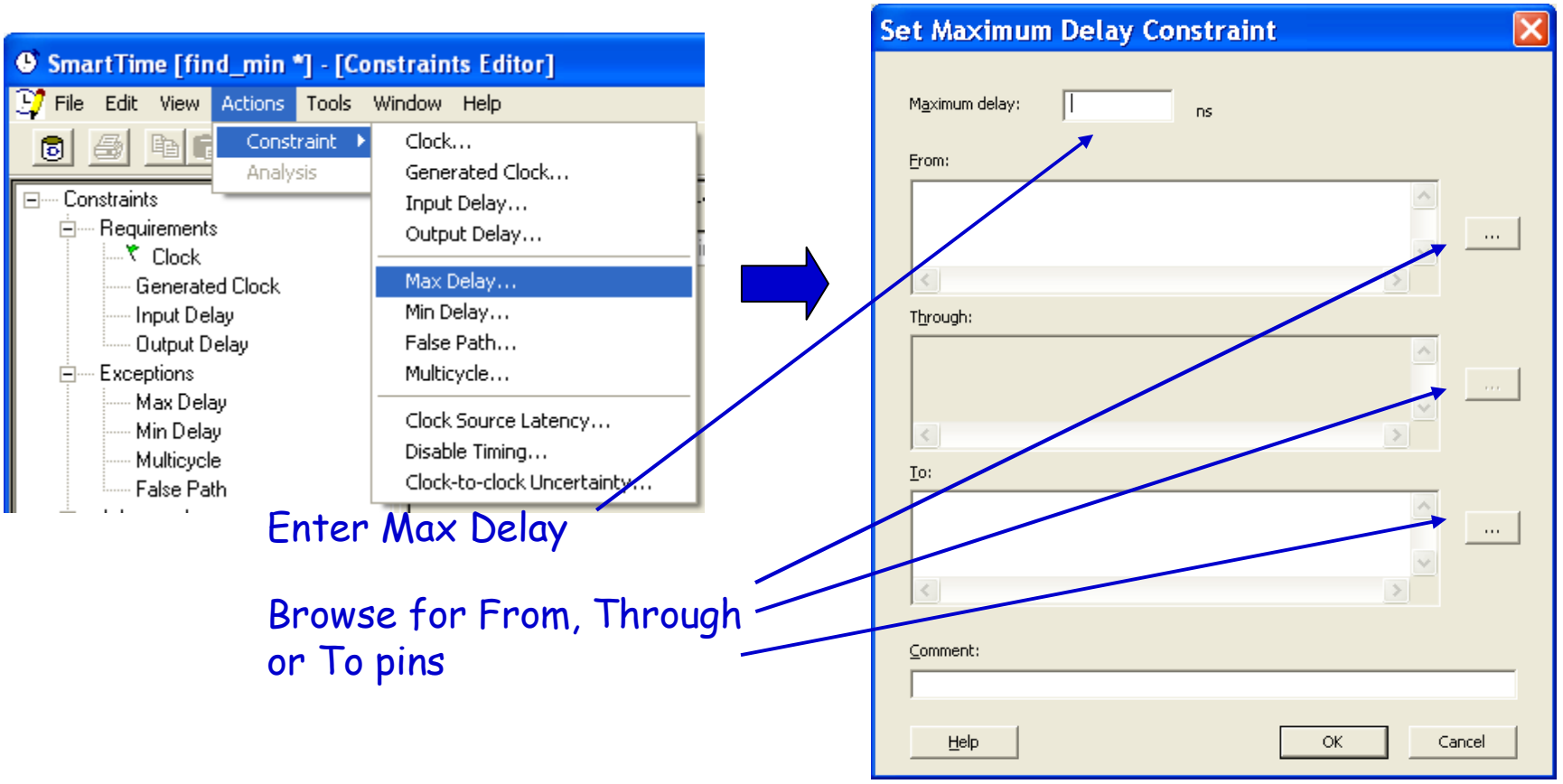


Timing Exceptions
Maximum Delay

- The Max Delay Constraint is Used to Specify The Maximum Delay for Timing Paths in The Design
- Max Delay is a Timing Exception and Overrides The Timing Requirement Between Flip Flops
  - Maximum Delay Involving Registers Takes Clock Insertion and Setup/Hold Delays Into Account
- Max Delay Can Also be Used to Constrain Combinatorial Input to Output Paths

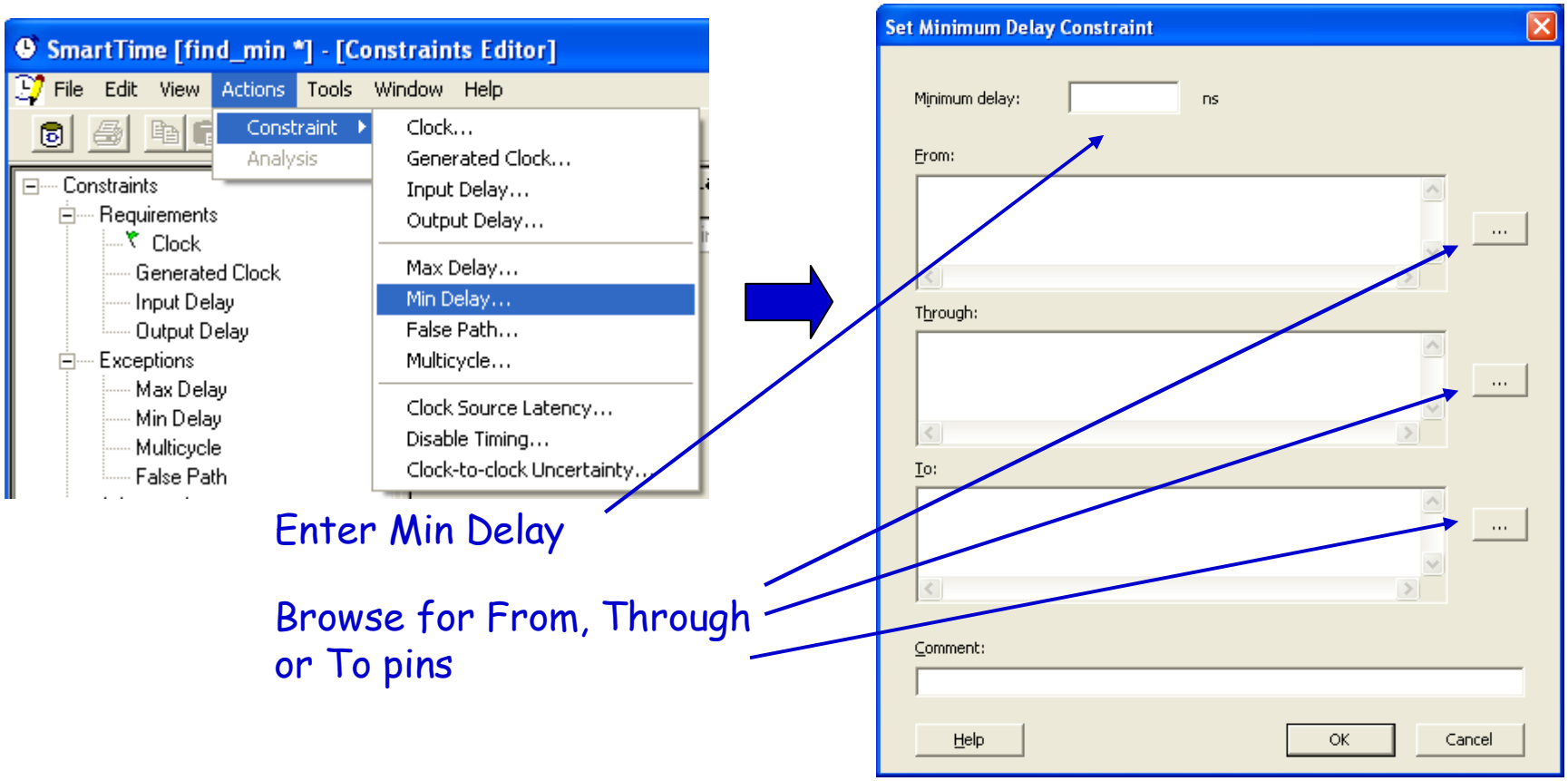
Timing Exceptions
Maximum Delay

- Specify Maximum Delay Constraint
  - Overrides Clock Constraint



Timing Exceptions
Minimum Delay

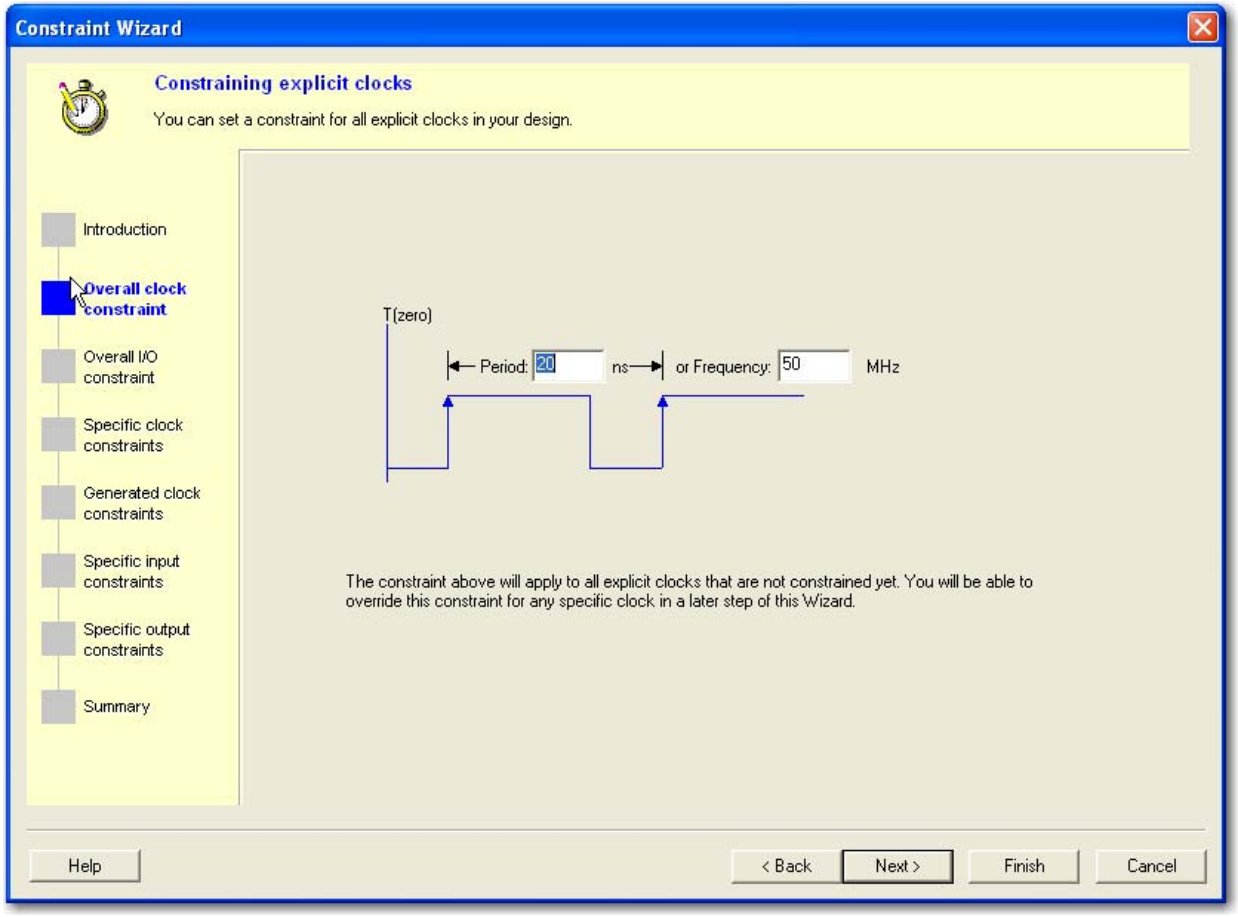
- Specify Minimum Delay Constraint
  - Overrides Clock Constraint



SmartTime Constraint Wizard

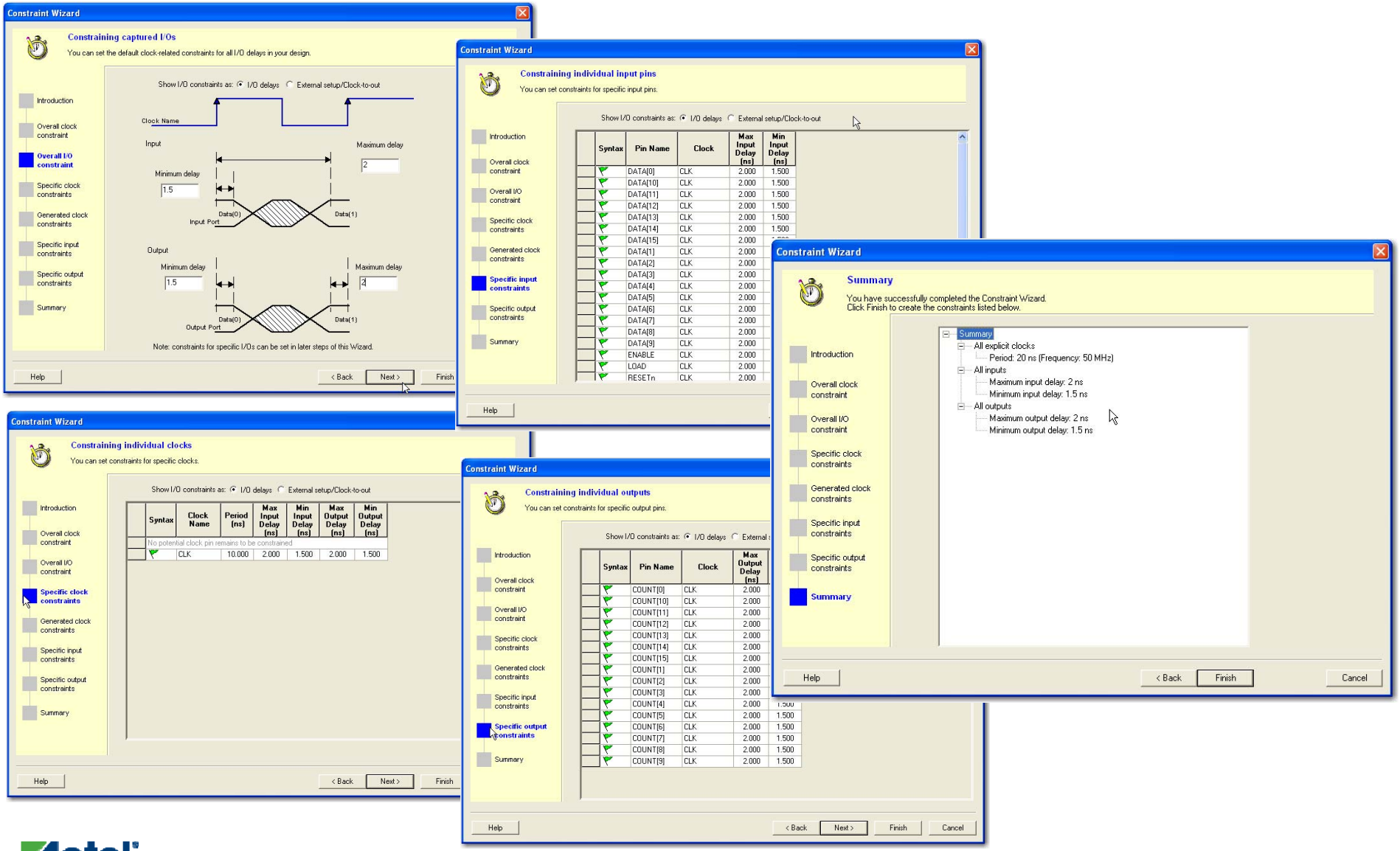
SmartTime Constraints Wizard

- Step by Step Guidance to Create Timing Constraints
  - Overall Clock
  - Overall I/O
  - Specific Clocks
  - Generated Clocks
  - Specific Inputs
  - Specific Outputs
- Summary Displays all Constraints Entered



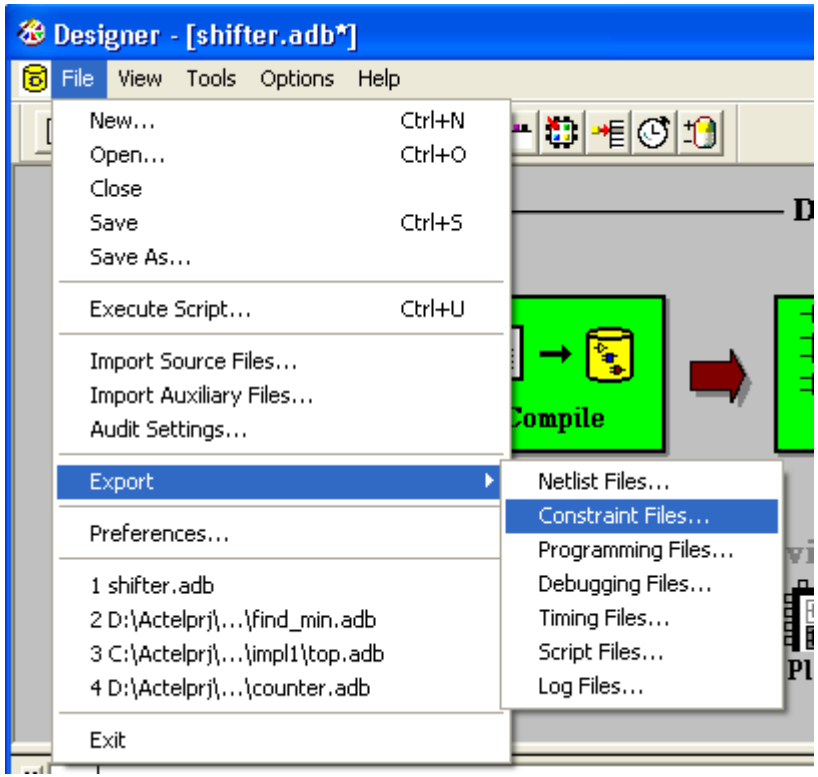
Opening the Constraint Wizard: Tools > Constraint Wizard

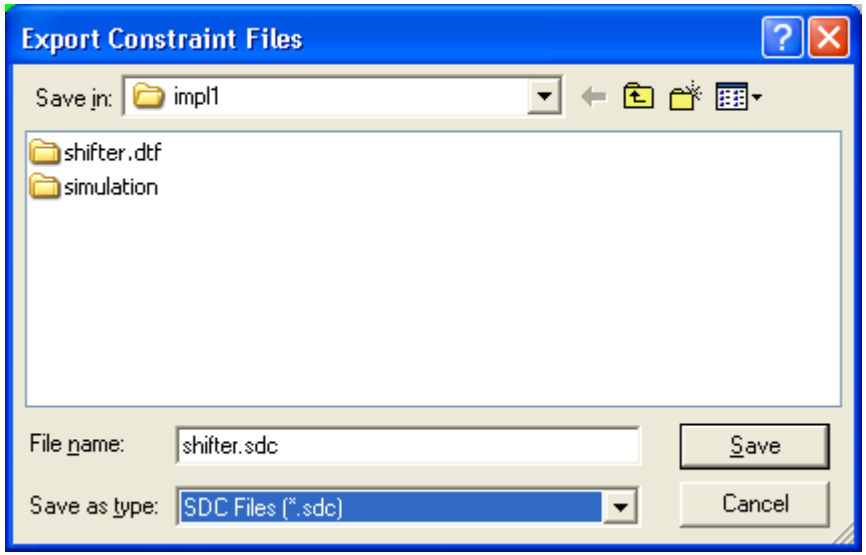
SmartTime Constraints Wizard



Exporting SDC Timing Constraints

- Constraints Entered in SmartTime GUI Can Be Exported from Designer





Constraint Recommendations

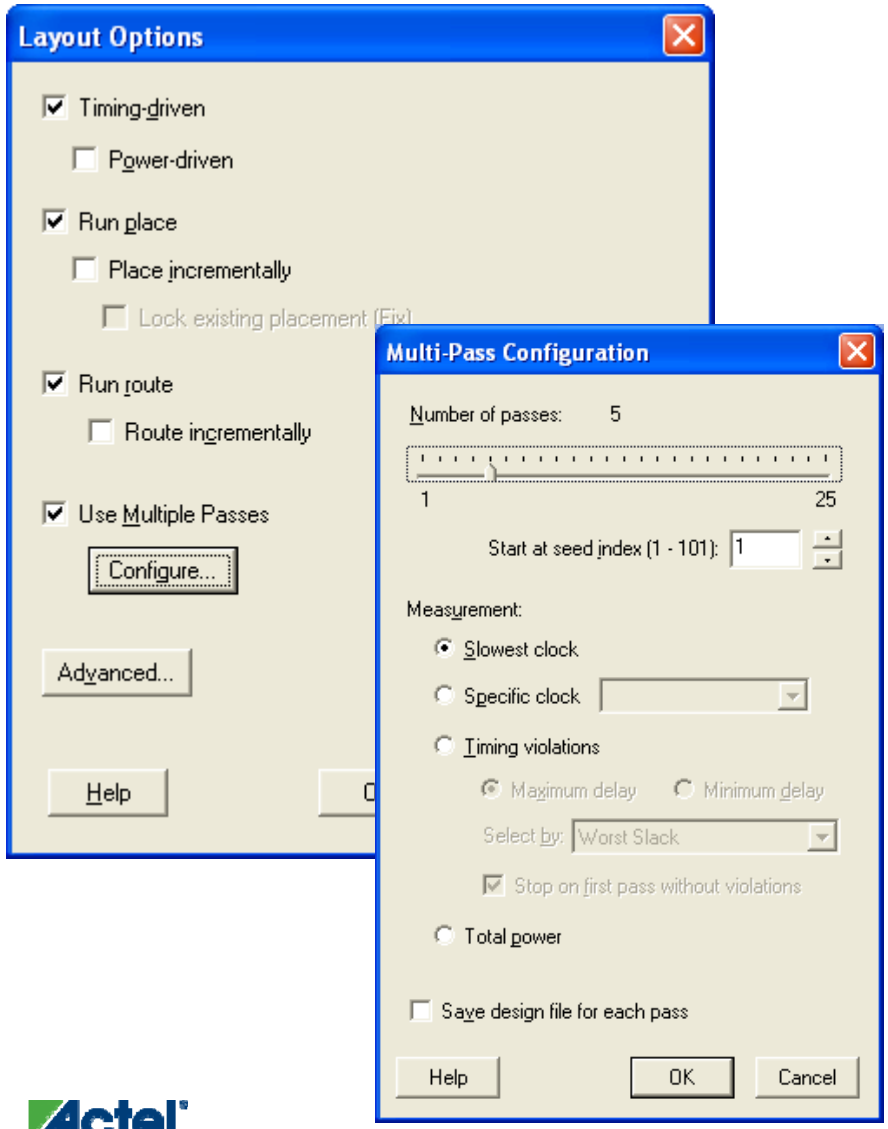
- Set Realistic Constraints
- Set Sufficient Constraints
- Don’t Over constrain
  - Improperly-Constrained Design Can Lead to Long Run Times, Multiple Iterations and/or Sub-optimal Results
- Enter any Known Exceptions
  - Exceptions can be Added During Timing Analysis

Designer Layout Options

Layout

- Assign Physical Locations to Unassigned I/Os
- Place Logic Modules
- Assign Routing Tracks to Nets
- Calculate Detailed Delays for All Paths

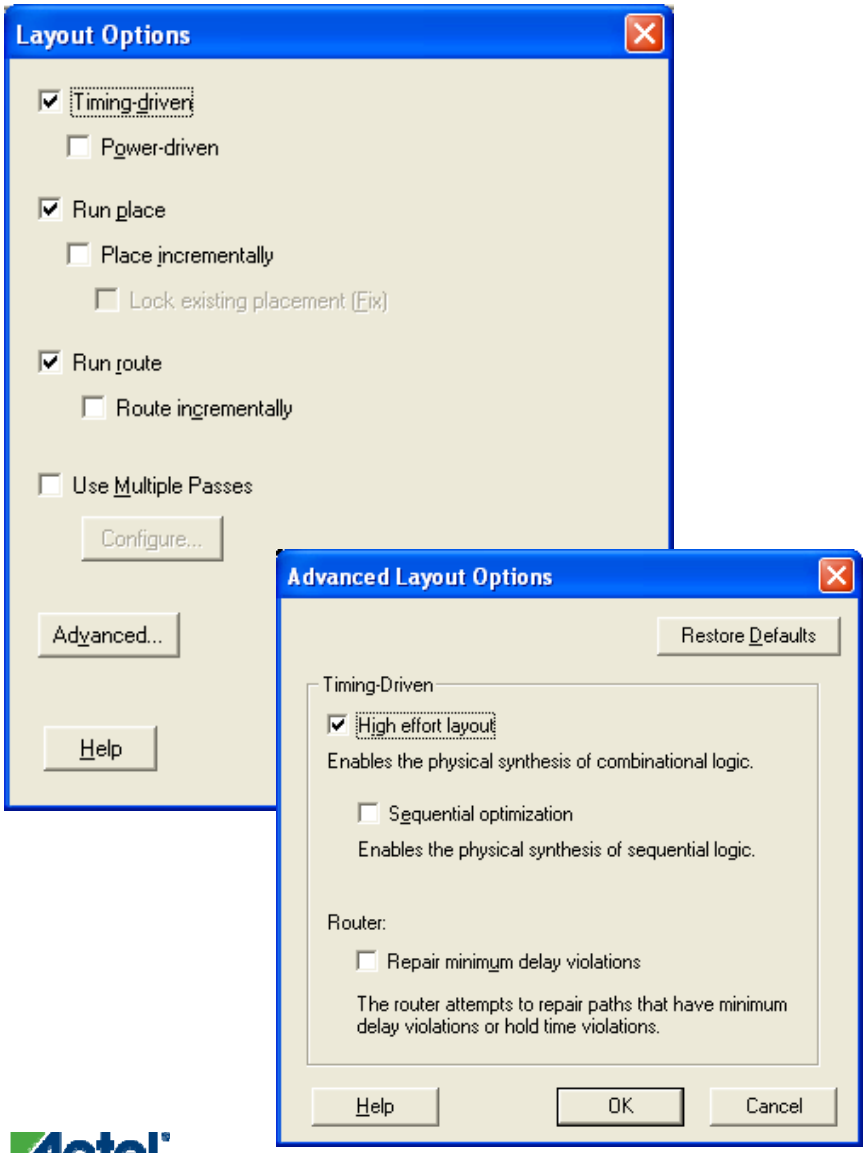
Layout Options
ProASIC3, ProASIC3L, Fusion, IGLOO



- Layout Mode
  - Timing-Driven: Constraints Defined in SmartTime
  - Power-Driver: Reduces Power Consumption
- Incremental Placement *and* Routing
- Multi Pass Layout Option
  - P&R Runs Multiple Times
    - Specify:
      - Number of Iterations
      - What to Optimize (Specific Clock, Timing Violations, Total Power)
      - Which Results to Save (Best or All)
- Advanced Layout Options
- Known Limitations
  - Router Cannot Run in Incremental Mode if there Has Been Change in Global Assignments
    - Users Must Manually Uncheck Incremental Routing Option and Re-run Layout

Advanced Layout Options

ProASIC3 ProASIC3L, Fusion, IGLOO



- High Effort Layout Mode
  - Placer uses additional netlist optimizations to improve performance (up to 10% QOR increase)
  - Layout runtime will increase (average 2x)
  - Can combine this option with the Multi-Pass mode to achieve the best possible performance
- Repair Minimum Delay Violations
  - An additional route is performed that attempts to repair paths that have minimum delay and hold time violations
  - Done by increasing the length of routing paths and inserting routing buffers to add delay to paths
    - No additional tiles or modules are inserted
  - The amount of delay inserted is limited
    - Best suited to repair paths with small (0 to 3 ns) hold and minimum delay violations

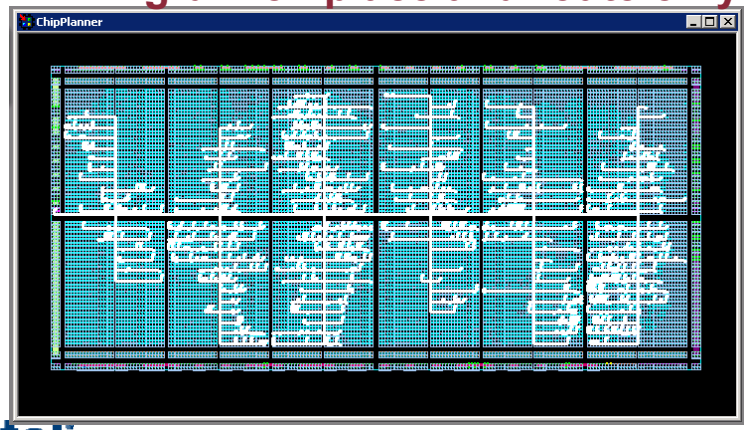
Sequential Optimization

- Power-driven + High-effort + Sequential optimization
  - 8.4
    - No-op
      - Otherwise, SEQ tiles increase by 6.5% on average
      - High-effort itself reduces COMB tiles by 6.1% on average
  - 8.4_SP1
    - Remove duplicate registers that are driven by the same net
      - SEQ tiles reduce by 3.3% on average
  - 8.5_SP1
    - Perform forward retiming of registers towards output I/Os

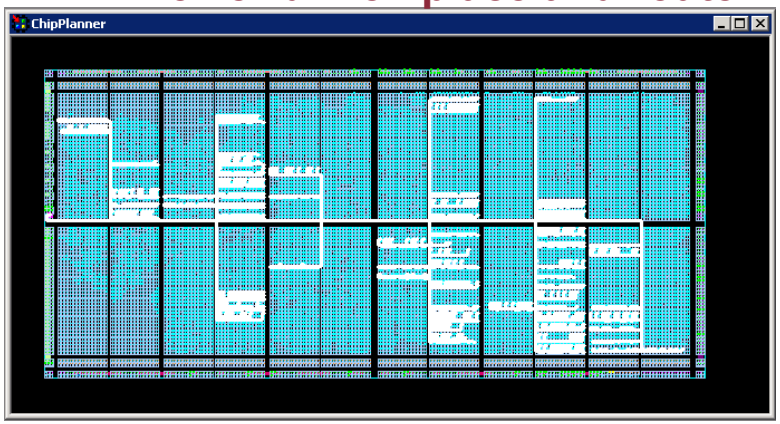
Designer Power-driven Layout
ProASIC3, Fusion, IGLOO, Axcelerator

- Layout Optimization Reduces Power Consumption
  - Reduces net capacitance based on estimated activities
    - Re-places instances and Optimizes Clock Tree distribution
  - Average ~18% reduction in *net power*
    - Up to 50% in some designs. *Includes clocks and regular nets*
  - Average ~13% reduction in *total power*
    - Up to 33% in some designs. *IGLOO designs with 5pf output load and LVCMOS 1.5V I/Os*
  - Little to no effect on timing performance
    - Typically less than -2%

Timing-driven place and route only



Power-driven place and route

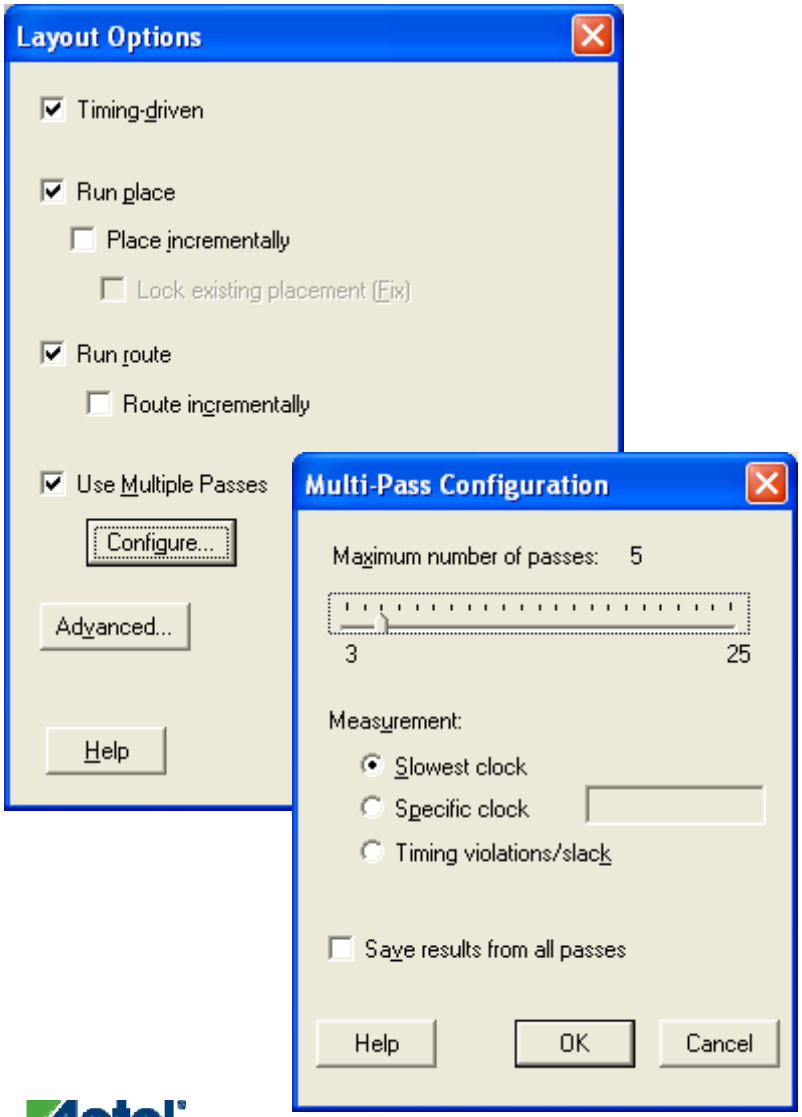


Quadrant Clock Assignment

ProASIC3, Fusion, IGLOO

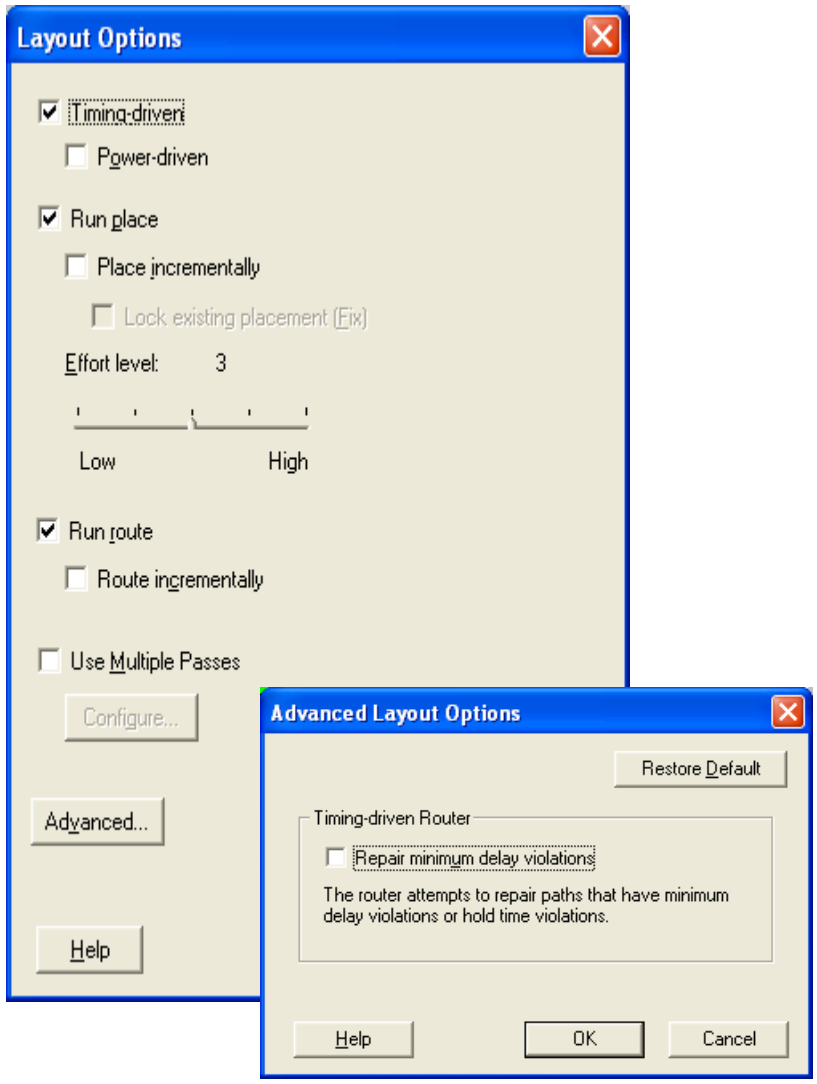
- Finds And Creates Legal Clock Assignments for Clock nets not Already Assigned by the User
- Runs Automatically As Part Of Layout
- Performs Auto-assignment of nets Given Certain Criteria
  - Runs when > 6 globals (or > 2 PLLs are detected)
    - Globals come from user netlist or PDC constraints
      - CCC macro instantiation
      - Global promotion through PDC constraints
    - If > 6 global nets, LCA will assign the global nets to
      - Chip wide globals or
      - Quadrant regions
    - The choice of chip wide globals or a quadrant region will depend on
      - Fanout of the nets
      - Resources and user constraints
      - Shared loads

Layout Options
ProASIC[PLUS]



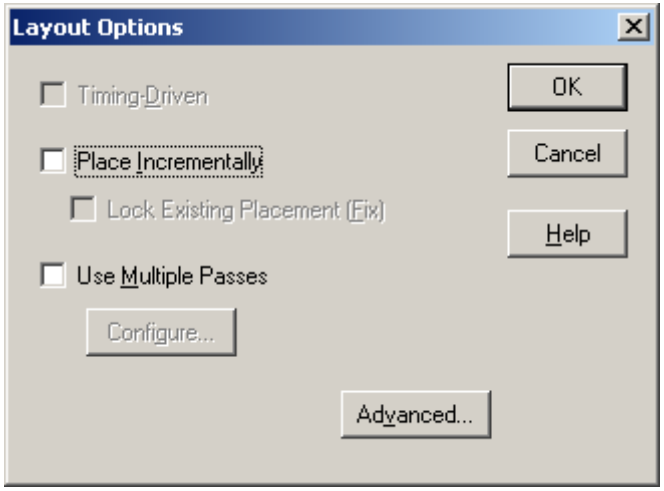
- Layout Mode
  - Timing-Driven: Constraints Defined in SmartTime
    - De-selecting Causes Standard Layout to be Used
- Place and Route Tools
  - Can Be Turned On or Off
- Incremental Placement *and* Routing
  - Lock Existing Placement: Treats All Unchanged Macros as Fixed Placements
- Multi Pass Layout Option

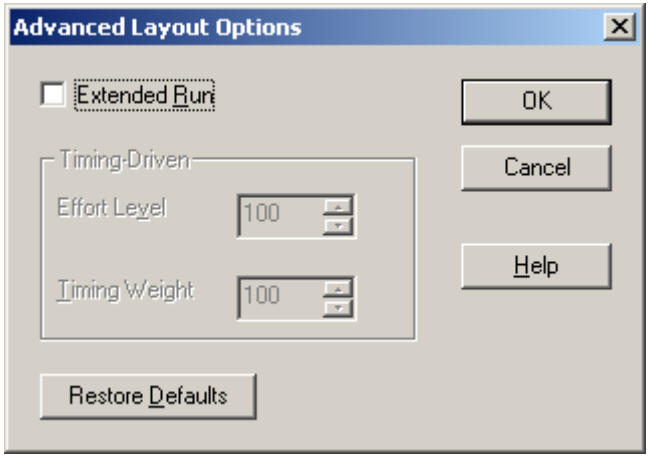
Layout Options
Axcelerator



- Layout Mode
  - Timing-Driven
    - Constraints Defined in SmartTime
      - De-selecting Causes Standard Layout to Be Used
    - Power Driven
- Incremental Placement *and* Routing
  - Lock Existing Placement: Treats All Unchanged Macros as Fixed Placements
- Placement Effort Level
  - Provides Degree of Control over Timing-Driven Placement Engine
  - Range is from “Low” to “High”
- Multi Pass Layout Option
- Repair Minimum Delay Violations

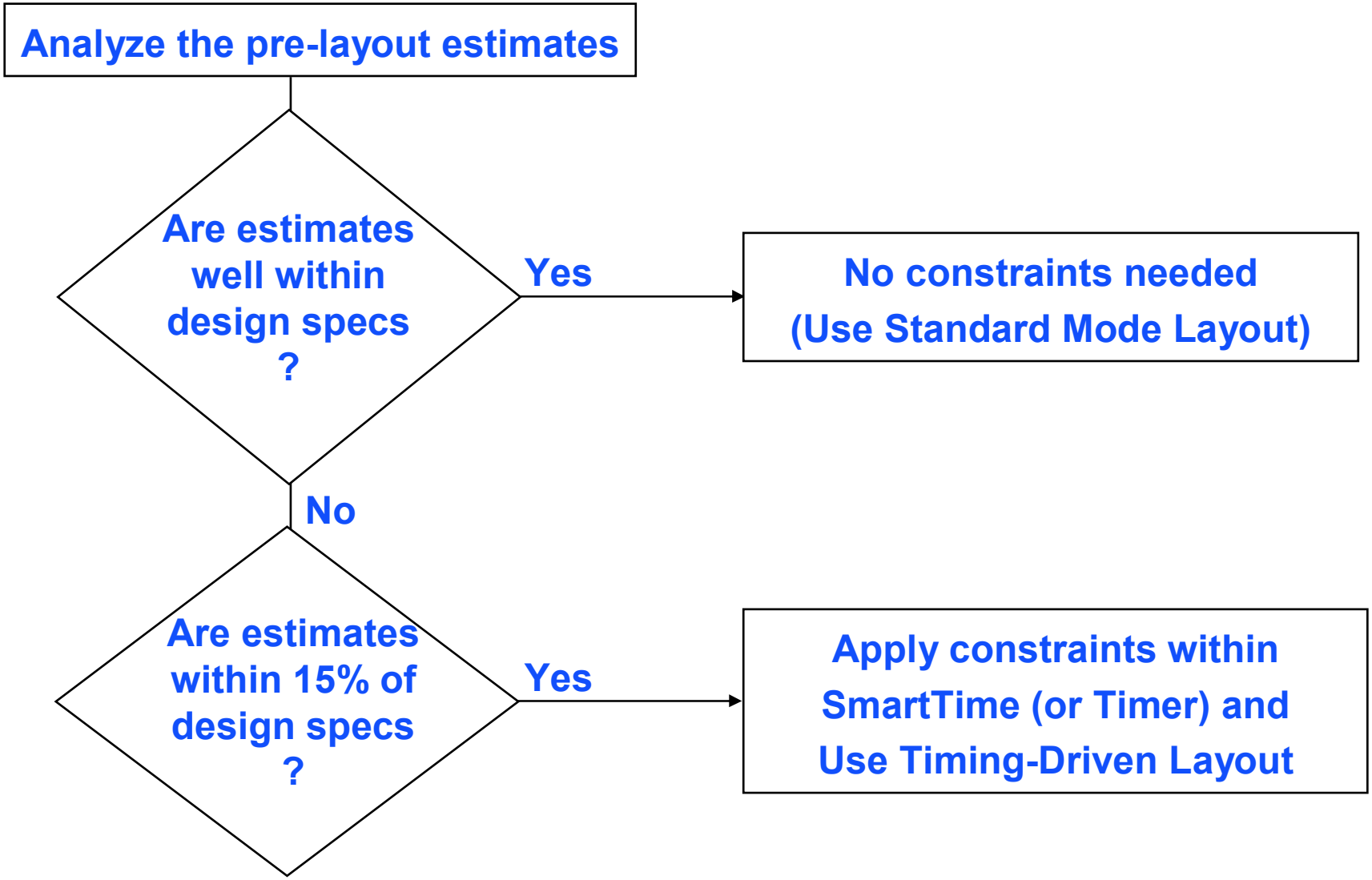
Layout Modes
SX Architecture



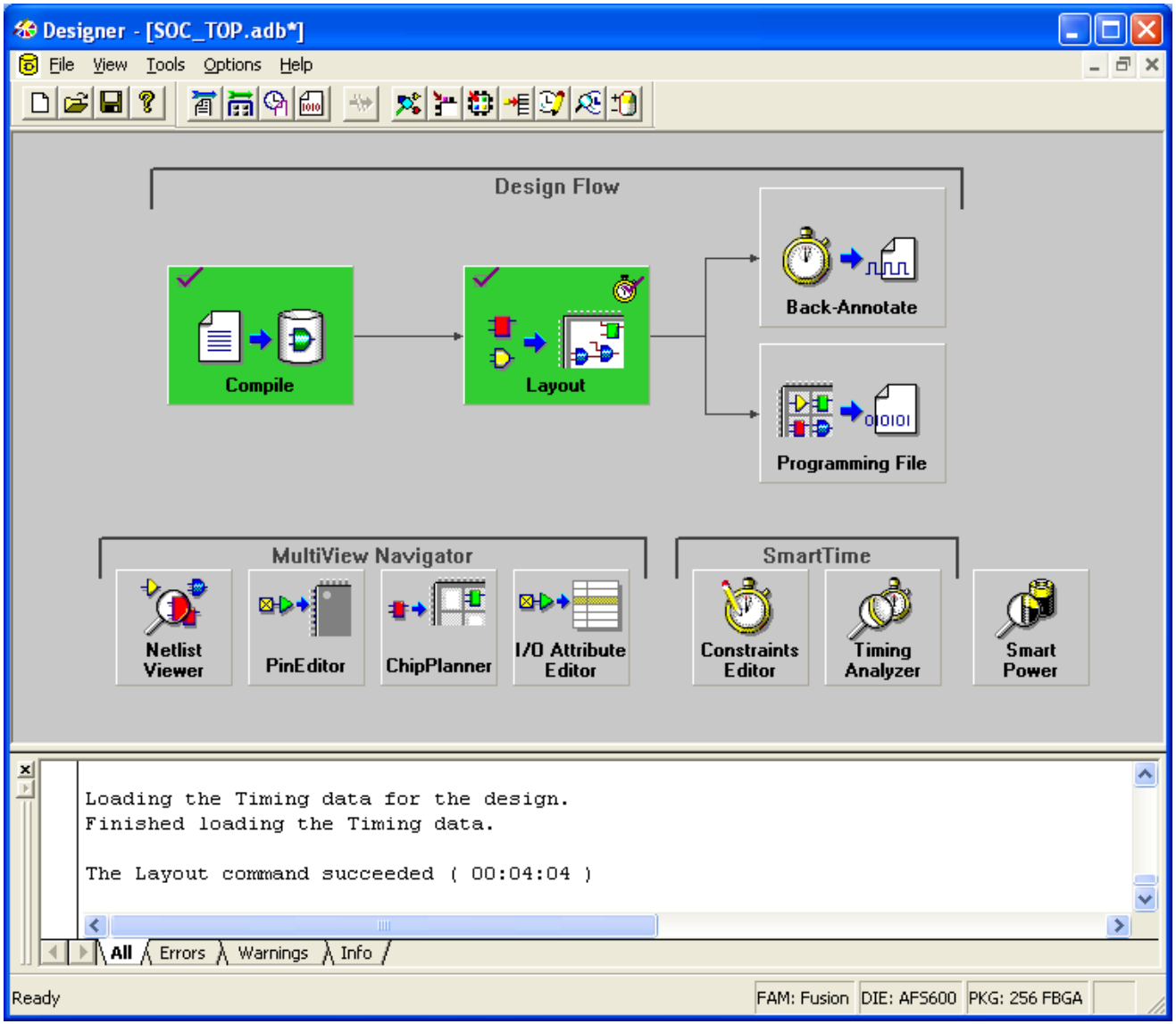


- Layout Mode
  - Timing-Driven: Constraints Defined in SmartTime (SX-A) or Timer (SX)
    - De-selecting Timing-Driven Layout Selects Standard Layout
- Incremental Placement
  - Lock Existing Placement: Treats All Unchanged Macros as Fixed Placements
    - De-selecting Allows Placer to Relocate Unchanged Macros if Necessary
- Multiple Passes
  - P&R Runs Multiple Times
- Advanced Options Allow Additional Control of Timing-driven Placement Engine
  - SX, SXA and eX Families

Which Layout Mode to Use?



Completed Layout



Designer Files and Reports

Exporting Files

- Supported File Types (File>Export)

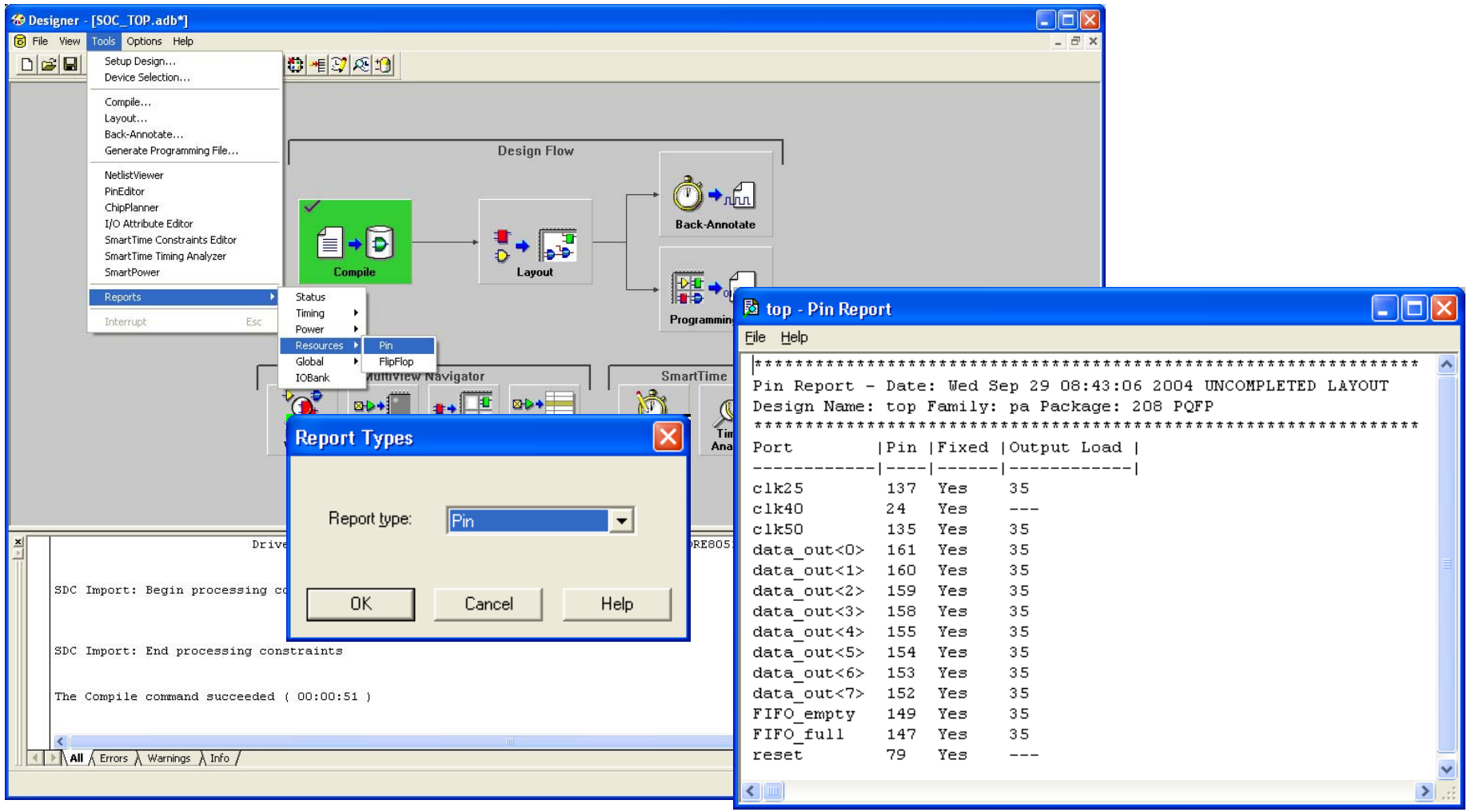
Files	File Extension	Export Type	Family
Actel Flattened Netlist	*.afl	Netlist file	All
Actel Internal Netlist	*.adl	Netlist file	All
Standard Delay Format	*.sdf	Timing file	All
STAMP	*.mod, *.data	Timing file	SX-A, eX, Axcelerator, IGLOO/e, ProASIC3/E, Fusion, ProASIC, ProASIC PLUS
Tcl script file	*.tcl	Script file	All
Verilog Netlist	*.v	Netlist file	All
VHDL Netlist	*.vhd	Netlist file	All
EDIF Netlist file	*.edn	Netlist file	All
Log File	*.log	Log file	All
STAPL	*.stp	Programming file	Fusion, IGLOO/e, ProASIC3/E, ProASIC, ProASIC PLUS
Bitstream	*.bit	Programming file	Fusion, IGLOO/e, ProASIC3/E, ProASIC, ProASIC PLUS
Programming file (legacy)	*.fus	Programming file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX
Actel programming file	*.afm	Programming file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX, SX, SX-A, eX, Axcelerator
Location constraint file	*.loc	Other file	SX-A
Routing Segmentation file	*.seg	Other file	Axcelerator, IGLOO/e, ProASIC3/E, Fusion, ProASIC, ProASIC PLUS
Silicon Explorer Probe file	*.prb	Debugging file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX, SX, SX-A, eX, Axcelerator

Exporting Files - Cont.

Files	File Extension	Export Type	Family
ProASIC Constraints file	*.gcf	Constraints file	ProASIC
ProASIC PLUS Constraints file	*.gcf	Constraints file	ProASIC PLUS (Timing constraints in GCF are not supported)
Combiner Info	*.cob	Other file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX, SX, SX-A, eX, Axcelerator, IGLOO/e, ProASIC3/E
BSDL file	*.bsd	Debugging file	3200DX, MX, SX, SX-A, eX, Axcelerator, Fusion, IGLOO/e, ProASIC3/E, ProASIC, ProASIC PLUS
Criticality	*.crt	Constraints file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX
I/O buffer information specification (IBIS)	*.ibs	Other file	MX, ProASIC PLUS, IGLOO/e, ProASIC/E; the IBIS file provides a standard file format for recording parameters like driver output impedance, rise/fall time, and input loading, which may then be used by any software application.
PIN	*.pin	Constraints file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX, SX, SX-A, eX
SDC	*.sdc	Constraints file	SX-A, eX, Axcelerator, ProASIC PLUS, IGLOO/e, ProASIC3/E, Fusion
Physical Design Constraint	*.pdc	Constraints file	Fusion, IGLOO/e, ProASIC3/E, Axcelerator
Design Constraint file file	*.dcf	Constraints file file	ACT1, ACT2, ACT3, MX, 1200XL, 3200DX, SX, SX-A, eX

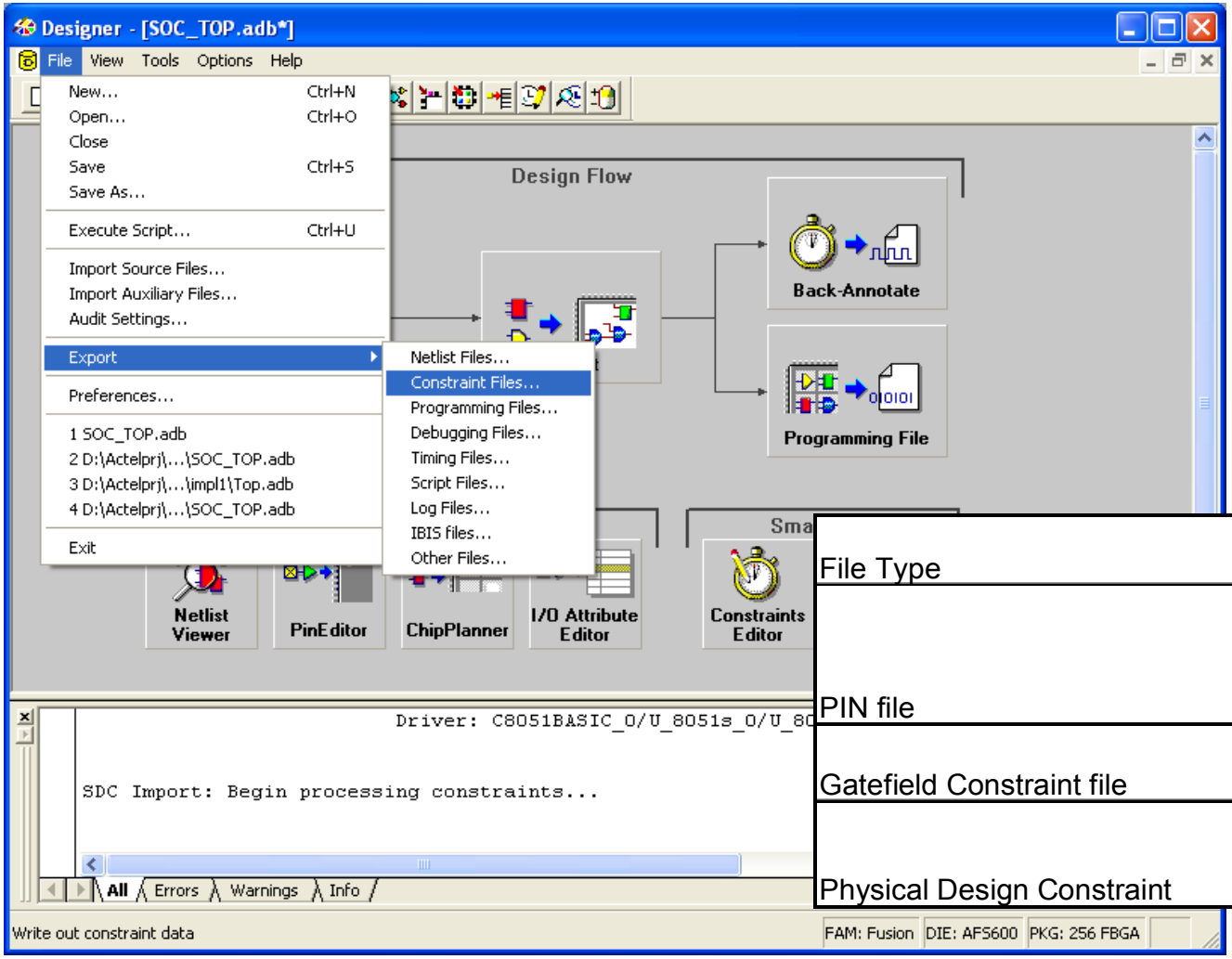
Exporting Pin Report

- Pin Report Can be Exported from Designer



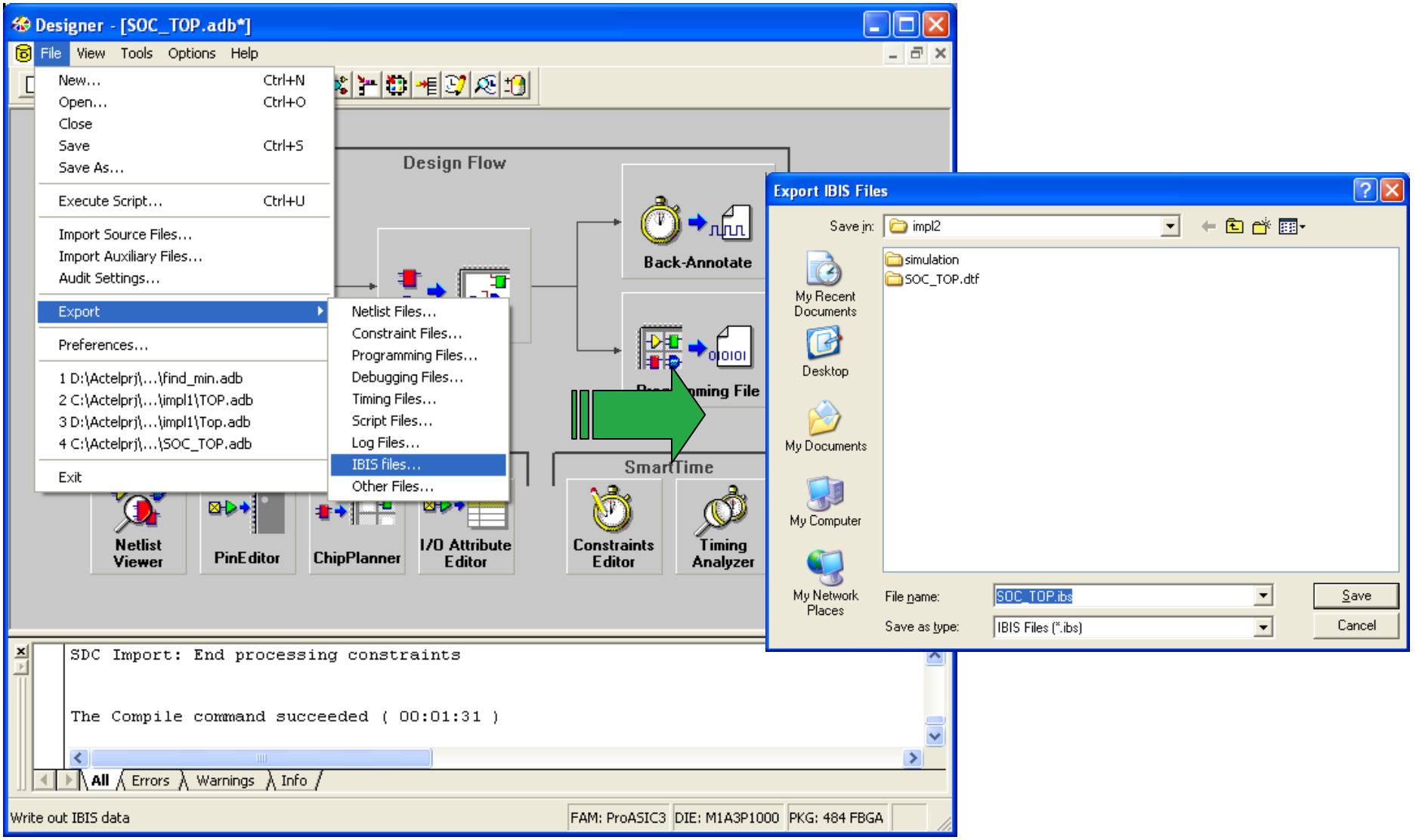
Exporting Pin Constraint File

- Pin Constraint File Can be Exported from Designer

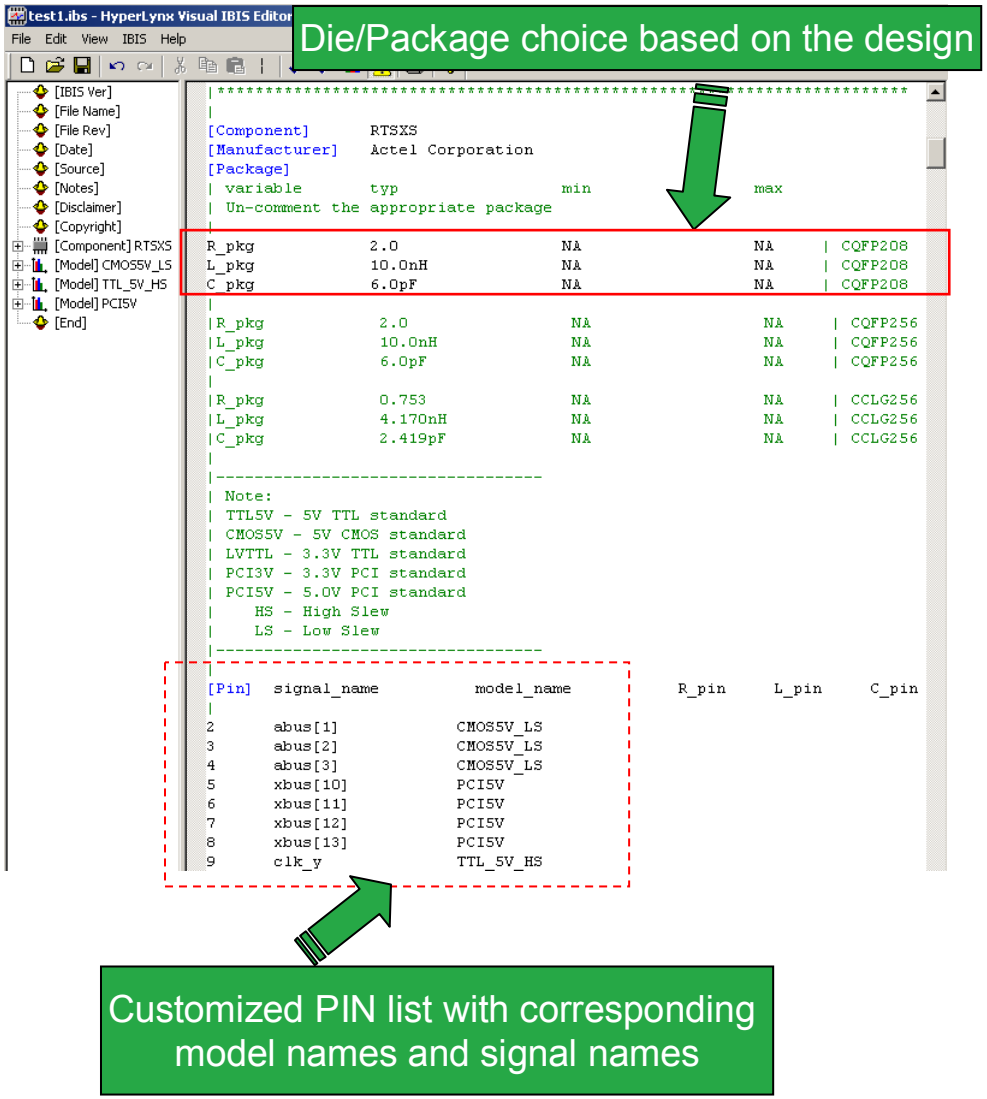


File Type	File Extension	Family
PIN file	*.pin	ACT1, ACT2, ACT3, MX, XL, DX, SX, SX-A, eX
Gatefield Constraint file	*.gcf	ProASIC, ProASIC PLUS
Physical Design Constraint	*.pdc	Axcelerator, ProASIC3\E, Fusion

Exporting IBIS File



IBIS File



More on IBIS later in the week . . .

Post-Layout Timing Analysis

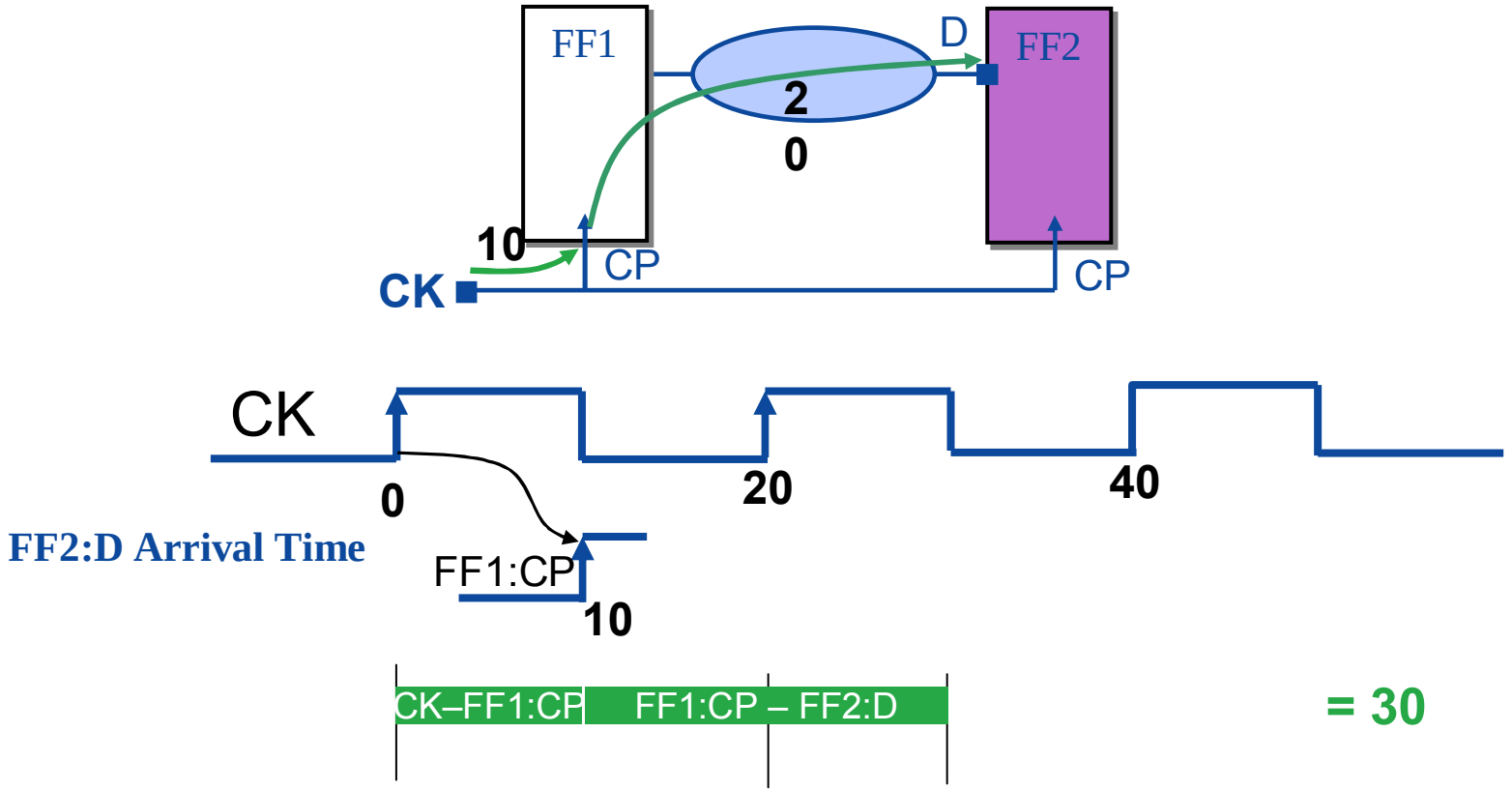
Timing Analysis

- Timing Terminology
- Timing Analysis with SmartTime
- SmartTime Timing Reports
- Reference

Timing Terminology

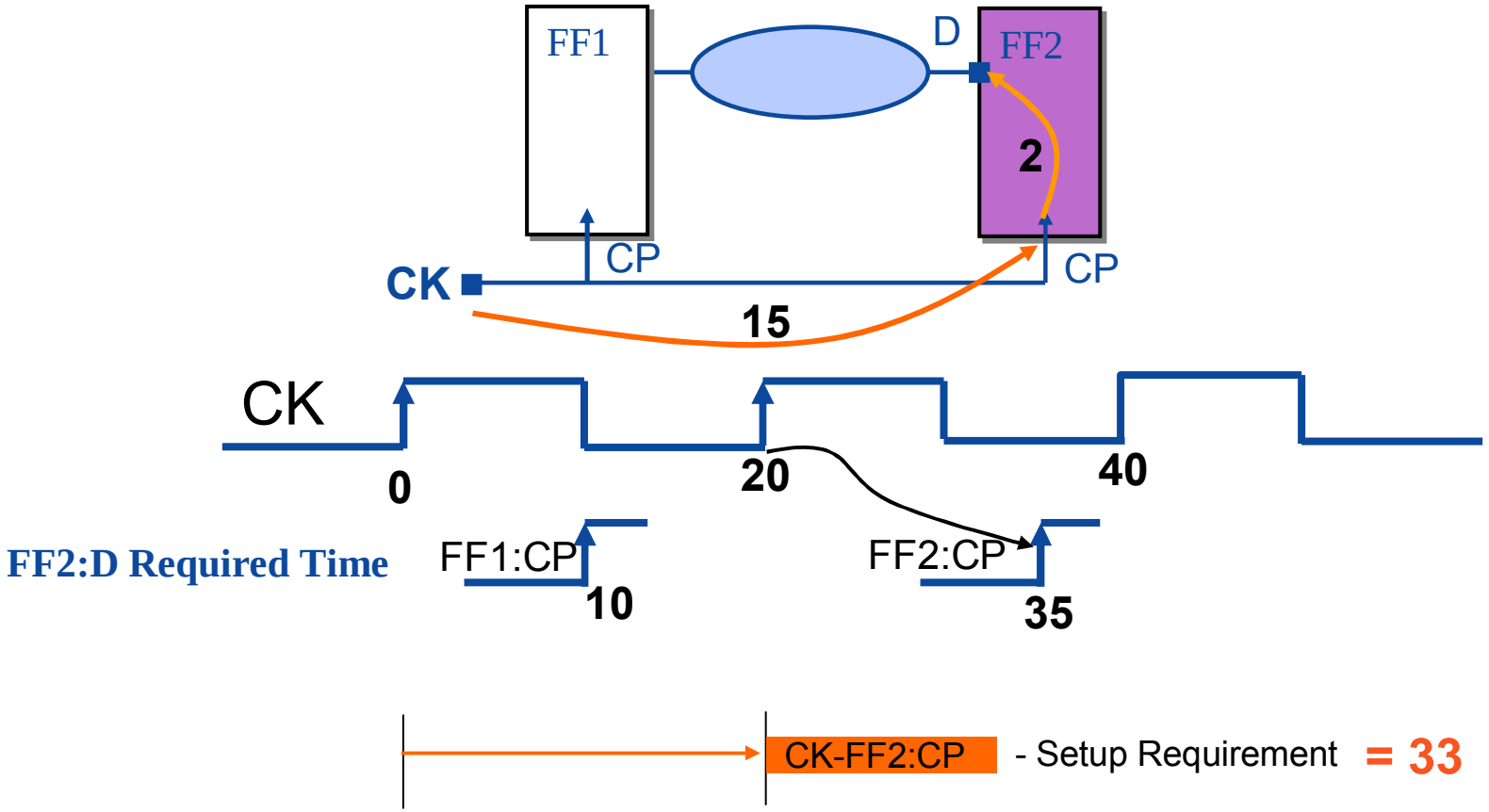
Timing Terminology:
Arrival Time

- Arrival Time is the Sum of:
  - The Clock Network Delay to the Source Flip-Flop
  - The Delay from the Source Flip-Flop to the Destination Flip-Flop



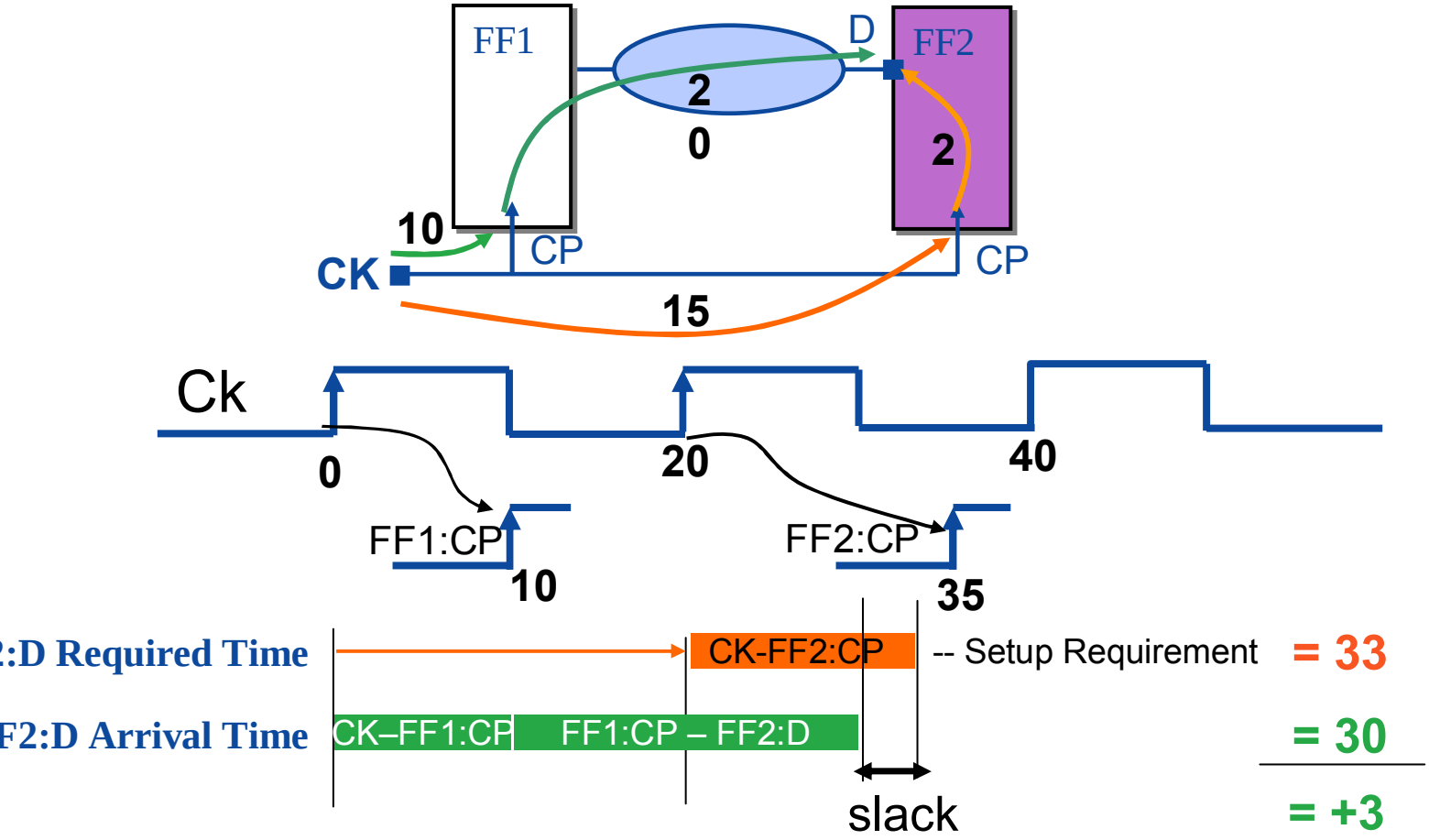
Timing Terminology:
Required Time

- Required Time is:
  - Clock Period *Plus* Clock Network Delay to Destination Flip-Flop
  - *Minus* the Setup Requirement for the Destination Flip-Flop



Timing Terminology:
Slack

- Slack =
  - Required Time – Arrival Time (setup check)
  - Arrival Time – Required Time (hold check)
  - Violation of Slack < 0



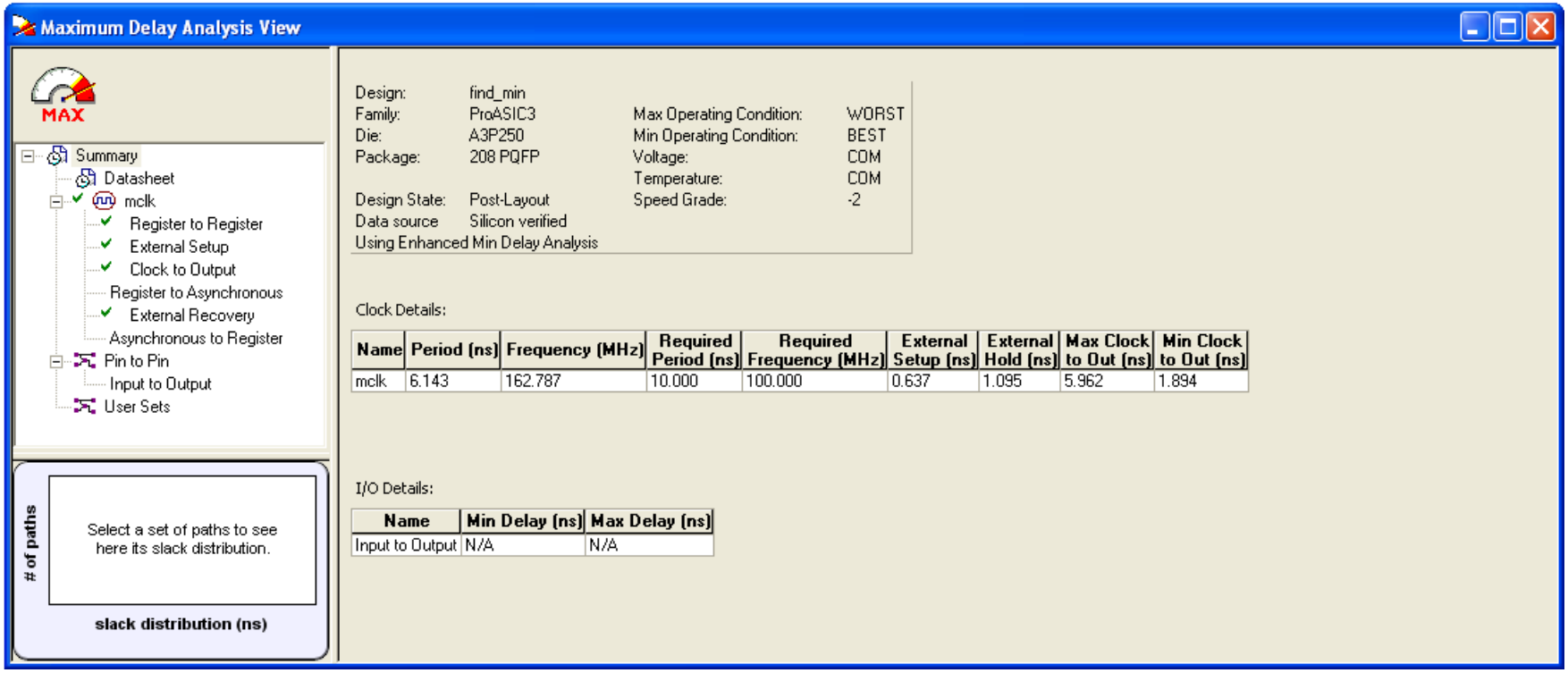
Timing Analysis with SmartTime

Timing Analysis with SmartTime

- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

SmartTime Analysis View
Design Summary

- Performance Summary is Displayed in Maximum and Minimum Delay Analysis View
  - √ - Indicates Timing Requirements Met for That Domain
  - X – Indicates There Are Violations Within the Domain

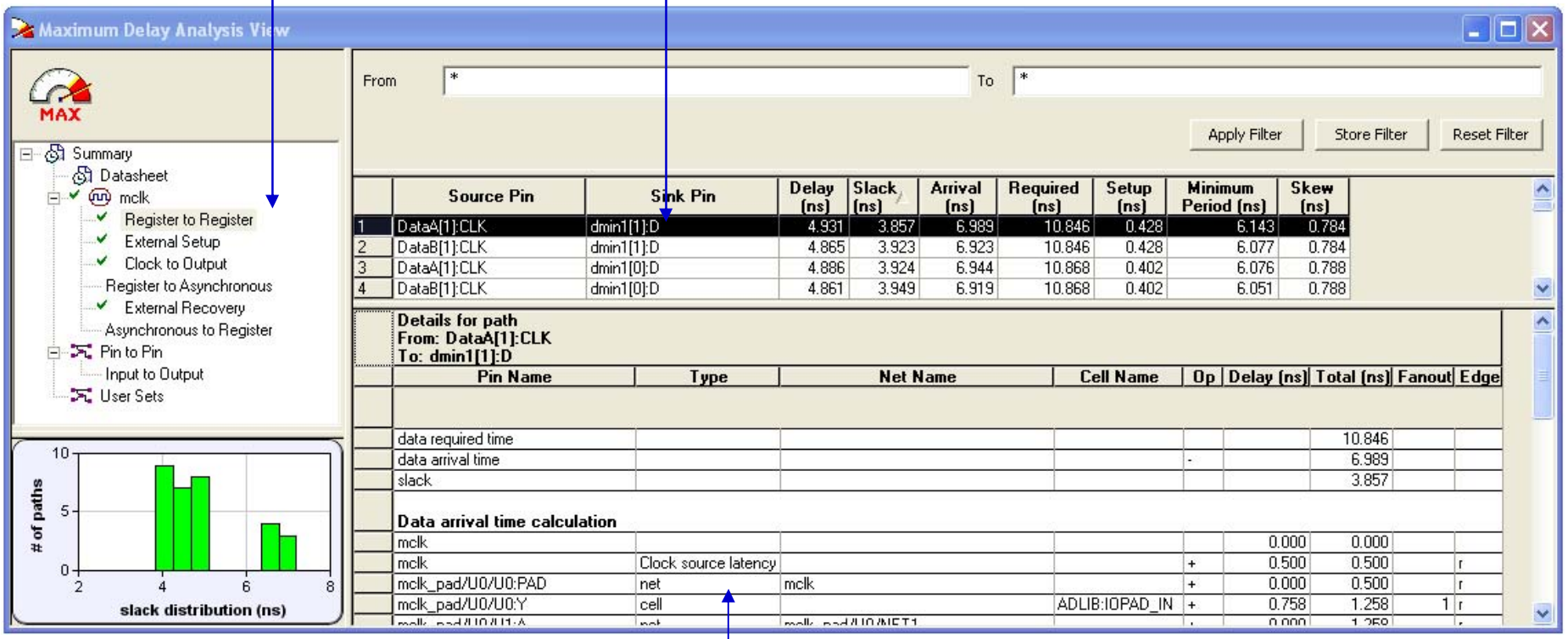


Timing Analysis with SmartTime

- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

Maximum Delay Analysis View
Register to Register Path Set

- Identify Internal Setup Violations

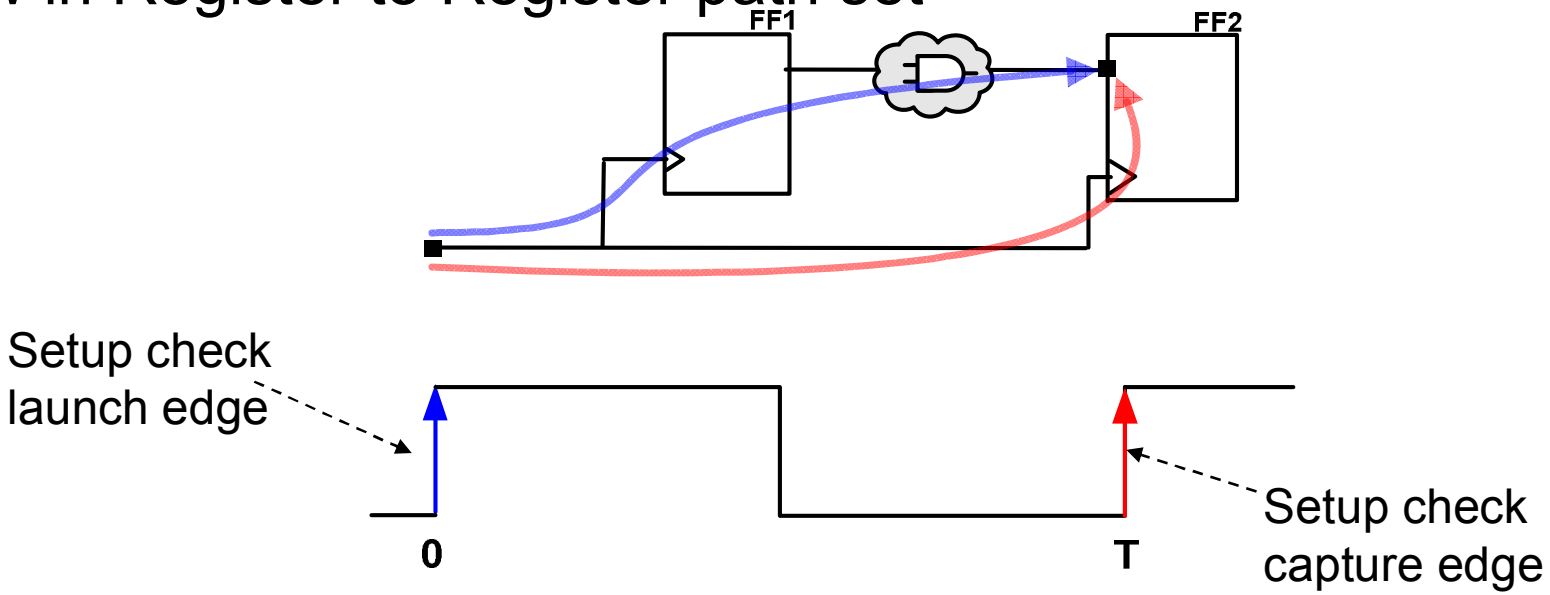


Maximum Delay Analysis View
Register to Register Path Set (cont.)

- Verify Internal Timing Requirements and Identify Paths With Violations
- By Default The Register to Register Path Displays:
  - Source Pin And Sink Pin – Beginning And Ending Points For Delay Calculation
  - Delay – Path Delay From The Clock Pin Of The Source Flip-flop To The Input Of The Destination Flip-flop
  - Slack – Required Time – Arrival Time
  - Arrival – Delay From The Launch Edge At The Source Flip-flop To The Data Input Of The Destination Flip-flop
  - Required – Time Data Must Be Valid To Avoid A Timing Violation; Calculated From Clock Period And Setup Requirement For Destination Flip-flop
  - Setup – Setup Time Requirement For The Destination Flip-flop
  - Minimum Period – Minimum Clock Period To Avoid A Timing Violation
  - Skew – Clock Insertion Delay To The Source Flip-flop Minus The Clock Insertion Delay To The Destination Flip-flop
- Timing Information For Each Type Of Set Can Be Customized

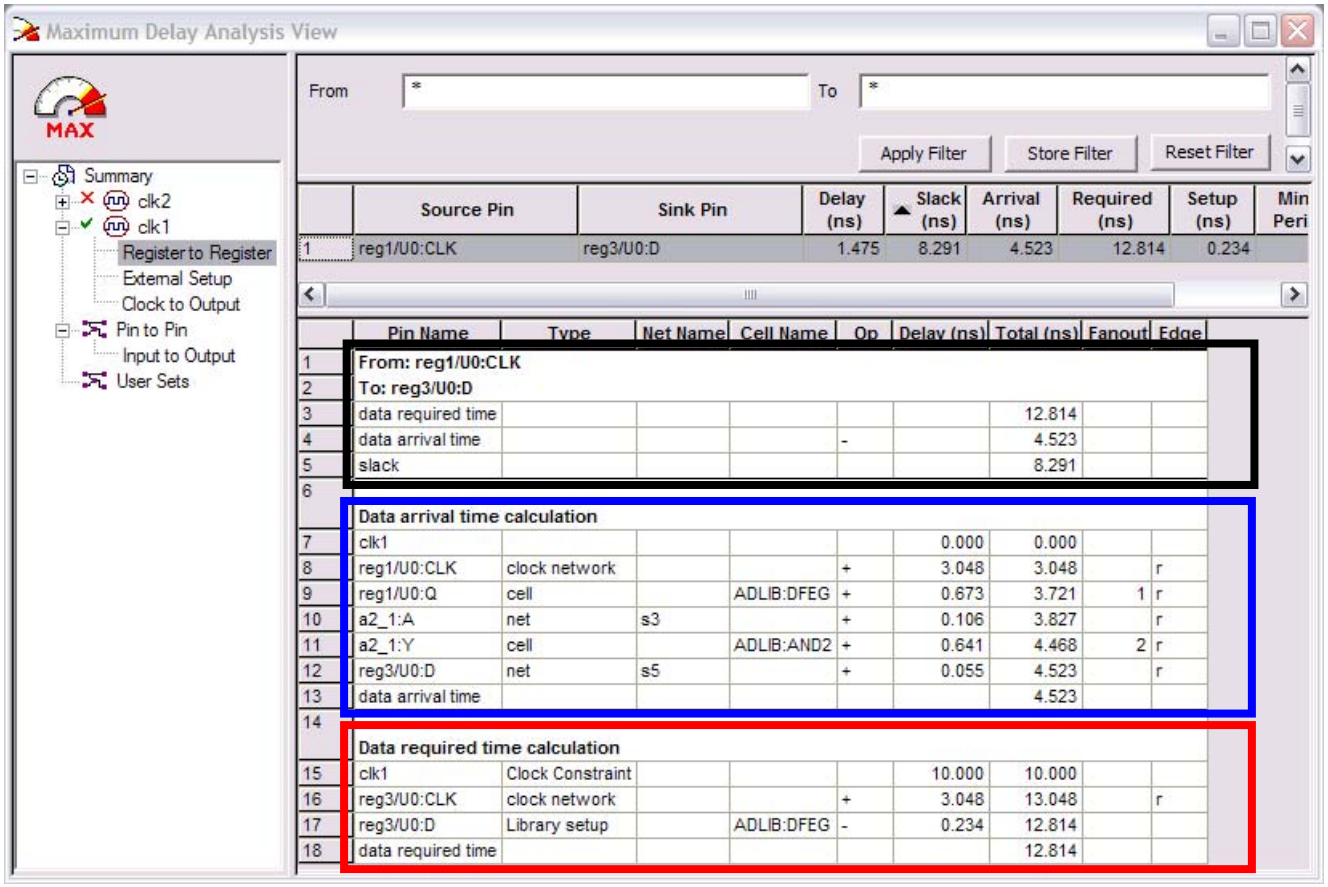
SmartTime
Internal Setup Check

- Setup Check is Displayed in Maximum Delay Analysis View in Register to Register path set



- Setup Check Calculation
  - Arrival time = Launch edge (0) + max Clock to FF1 + max Data path
  - Required time = Capture edge (T) + min Clock to FF2 – Setup of FF2
  - Slack = Required – Arrival = Violation if < 0

Setup Check in SmartTime



	Source Pin	Sink Pin	Delay (ns)	Slack (ns)	Arrival (ns)	Required (ns)	Setup (ns)	Min Peri
1	reg1/U0:CLK	reg3/U0:D	1.475	8.291	4.523	12.814	0.234	

	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	From: reg1/U0:CLK								
2	To: reg3/U0:D								
3	data required time						12.814		
4	data arrival time				-		4.523		
5	slack						8.291		
6									
	Data arrival time calculation								
7	clk1					0.000	0.000		
8	reg1/U0:CLK	clock network			+	3.048	3.048		r
9	reg1/U0:Q	cell		ADLIB:DFEG	+	0.673	3.721	1	r
10	a2_1:A	net	s3		+	0.106	3.827		r
11	a2_1:Y	cell		ADLIB:AND2	+	0.641	4.468	2	r
12	reg3/U0:D	net	s5		+	0.055	4.523		r
13	data arrival time						4.523		
14									
	Data required time calculation								
15	clk1	Clock Constraint				10.000	10.000		
16	reg3/U0:CLK	clock network			+	3.048	13.048		r
17	reg3/U0:D	Library setup		ADLIB:DFEG	-	0.234	12.814		
18	data required time						12.814		

- Setup
  - Arrival time = Launch edge (0) + max Clock to FF1 + max Data path
  - Required time = Capture edge (T) + min Clock to FF2 – Setup of FF2
  - Slack = Required – Arrival = Violation if < 0

Clock Source Latency: Analysis

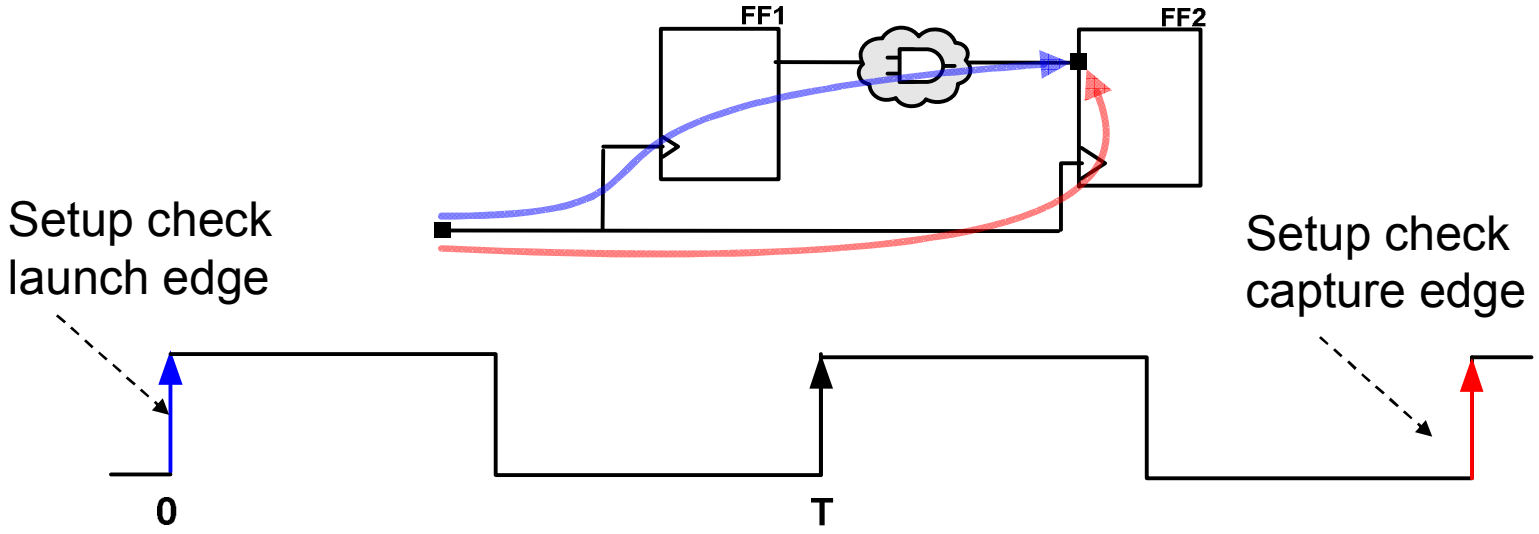
Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
From: irci:CLK **To: iry1:D**								
data required time						6.566		
data arrival time				-		8.285		
slack						-1.719		
Data arrival time calculation								
clk					0.000	0.000		
	Clock source latency			+	0.500	0.500		r
irci:CLK	Clock network			+	5.573	6.073		r
irci:Q	cell		ADLIB:DFEG	+	0.673	6.746	1	r
iadd2/add0_FCINST1:A	net	rci		+	0.151	6.897		r
iadd2/add0_FCINST1:FCO	cell		ADLIB:FCINIT_BUFF	+	0.543	7.440	1	r
iadd2/add0:FCI	net	iadd2/add0_FCNET1		+	0.000	7.440		r
iadd2/add0:FCO	cell		ADLIB:ADD1	+	0.075	7.515	1	r
iadd2/add1:FCI	net	iadd2/c0		+	0.000	7.515		r
iadd2/add1:S	cell		ADLIB:ADD1	+	0.610	8.125	1	r
iry1:D	net	sy[1]		+	0.160	8.285		r
data arrival time						8.285		
Data required time calculation								
clk	Clock Constraint				4.000	4.000		
	Clock source latency			+	-0.250	3.750		r
iry1:CLK	Clock network			+	3.050	6.800		r
iry1:D	Library setup time		ADLIB:DFEG	-	0.234	6.566		
data required time						6.566		

Late Launch

Early Capture

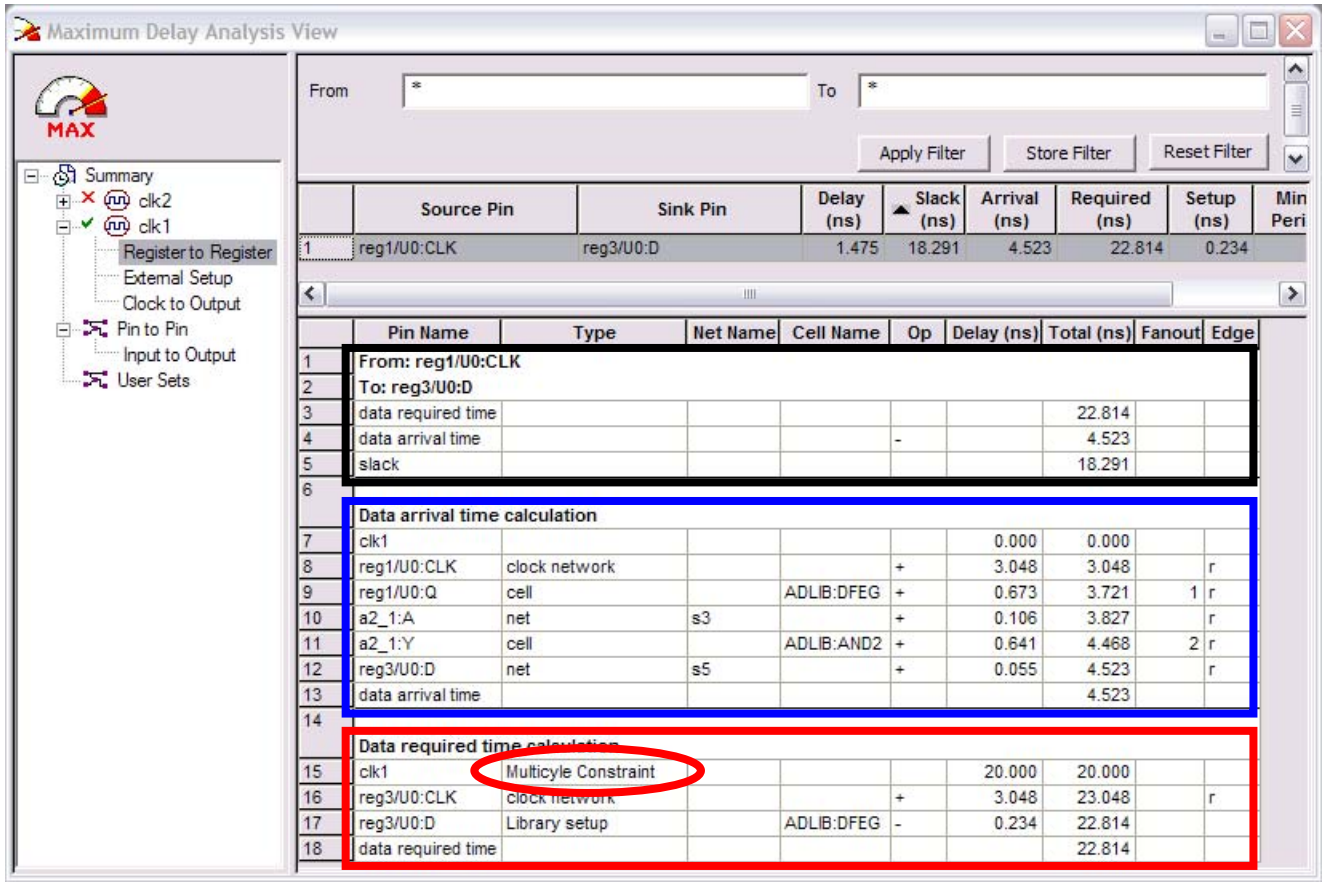
Setup Check w/ Multicycle Path

- SmartTime Uses Multicycle Path Constraint in Setup Check



- Setup Check Calculation
  - Arrival time = Launch edge (0) + max Clock to FF1 + max Data path
  - Required time = Capture edge (2T) + min Clock to FF2 – Setup of FF2
  - Slack = Required – Arrival = Violation if < 0

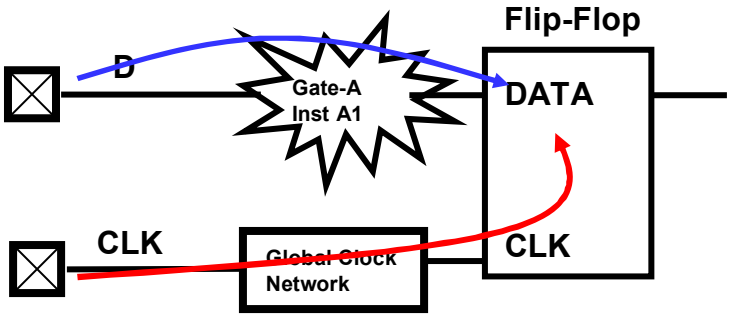
Setup Check w/ Multicycle Path

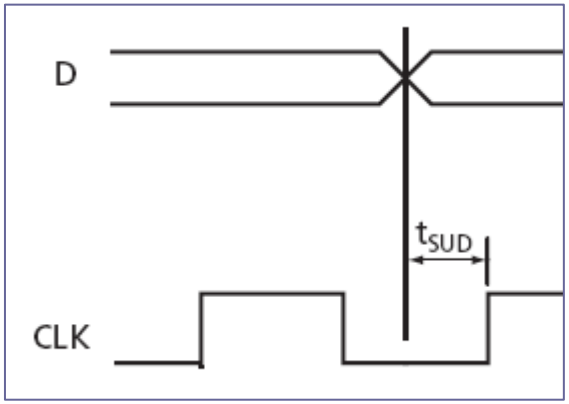


- Setup Check
  - Arrival time = Launch edge (0) + max Clock to FF1 + max Data path
  - Required time = Capture edge (2T) + min Clock to FF2 – Setup of FF2
  - Slack = Required – Arrival = Violation if < 0

SmartTime
External Setup Check

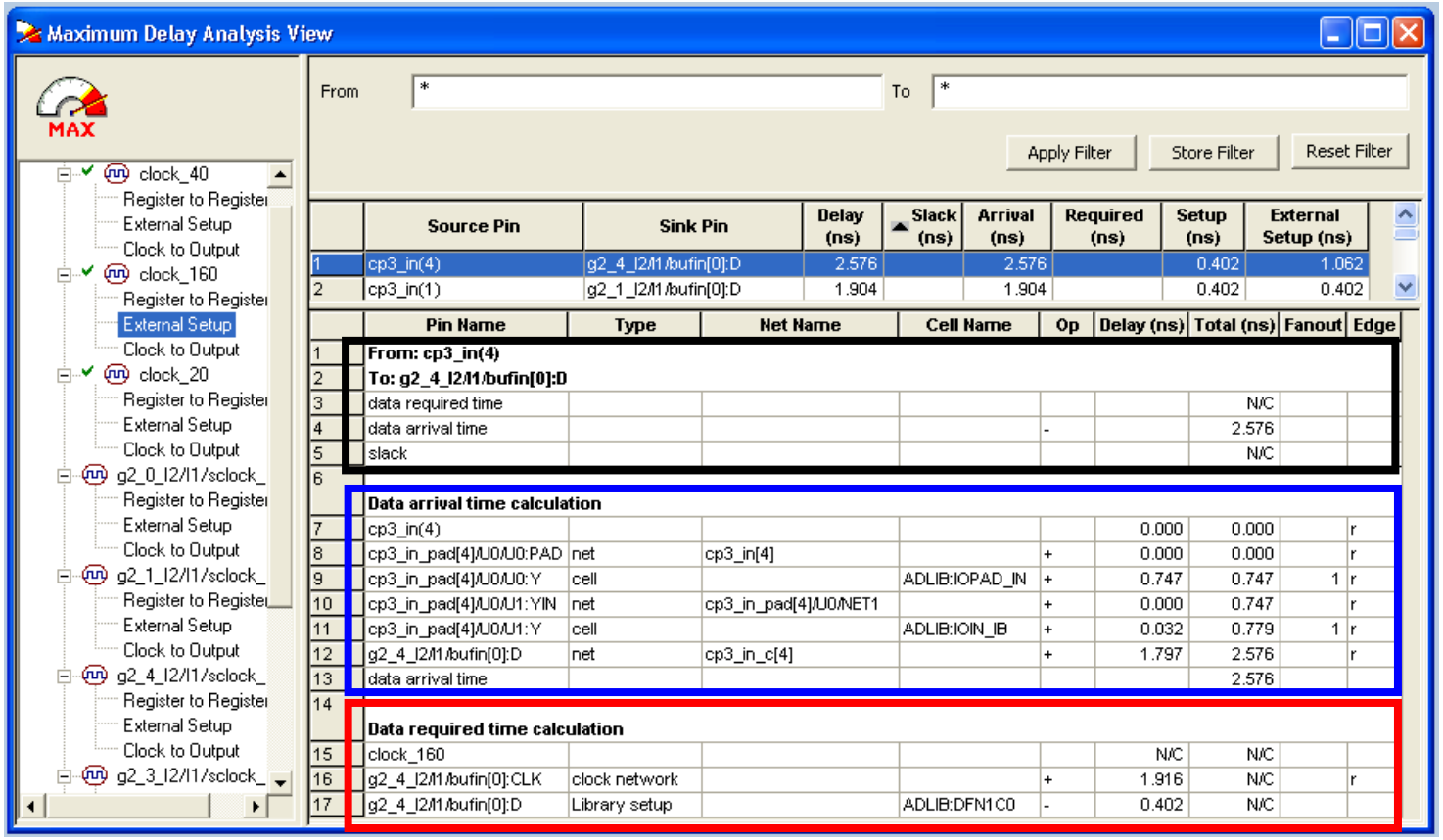
- External Setup Defines the Timing Requirements at the Input Pins
- External Setup Check is Displayed in the Maximum Delay Analysis View for Each Clock Domain





SmartTime External Setup Check

No Input Constraints

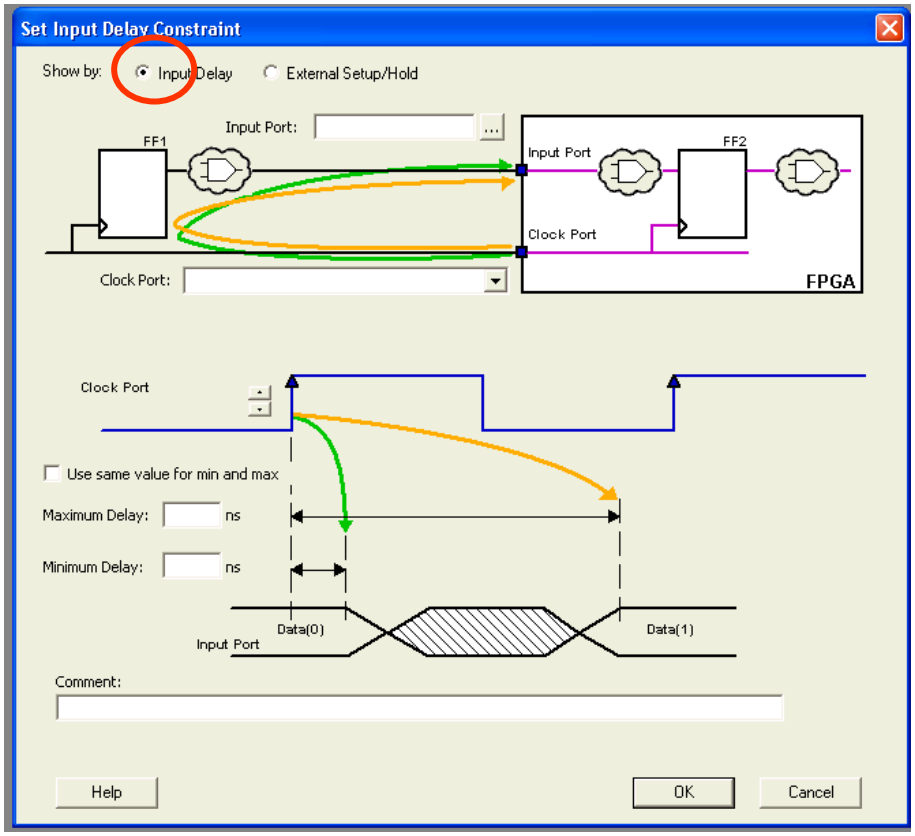


- External Setup Check
  - Arrival time = max Input Pad to FF
  - Required time = Capture edge (T) + min Clock to FF - Setup of FF
  - External Setup = Arrival - Required

SmartTime External Setup Check

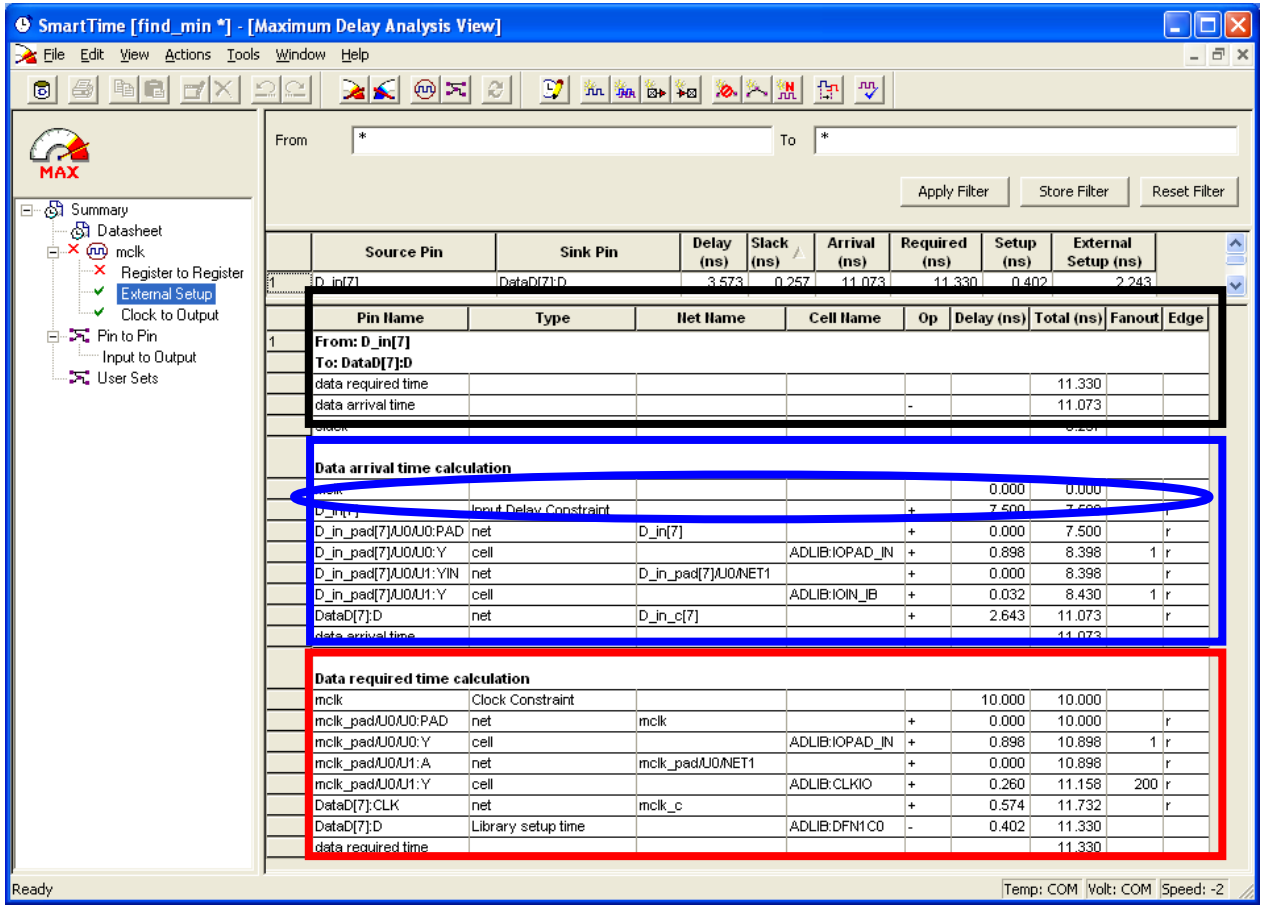
Using Input Delay Constraint

- Enter Constraint as Input Delay
  - Delay is referenced to *launch* edge of clock



SmartTime External Setup Check

Using Input Delay Constraint

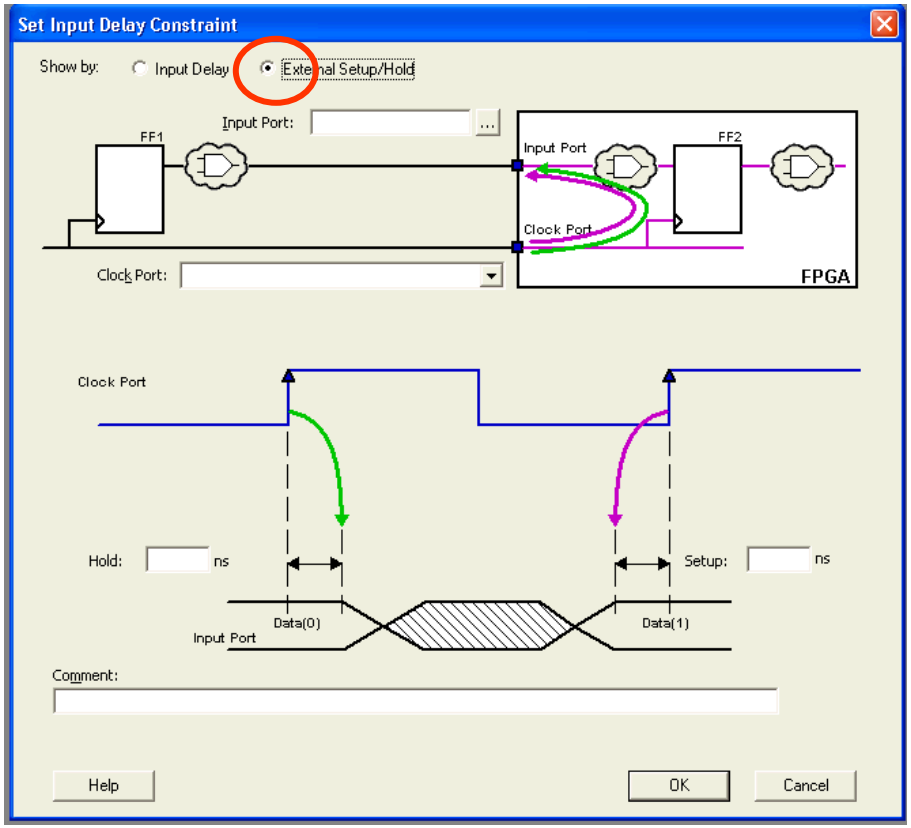


- External Setup Check Calculation
  - Arrival time = Input Delay Constraint + max input delay to FF
  - Required time = Capture edge (T) + min Clock to FF – Setup of FF
  - Slack = Required – Arrival = Violation if < 0

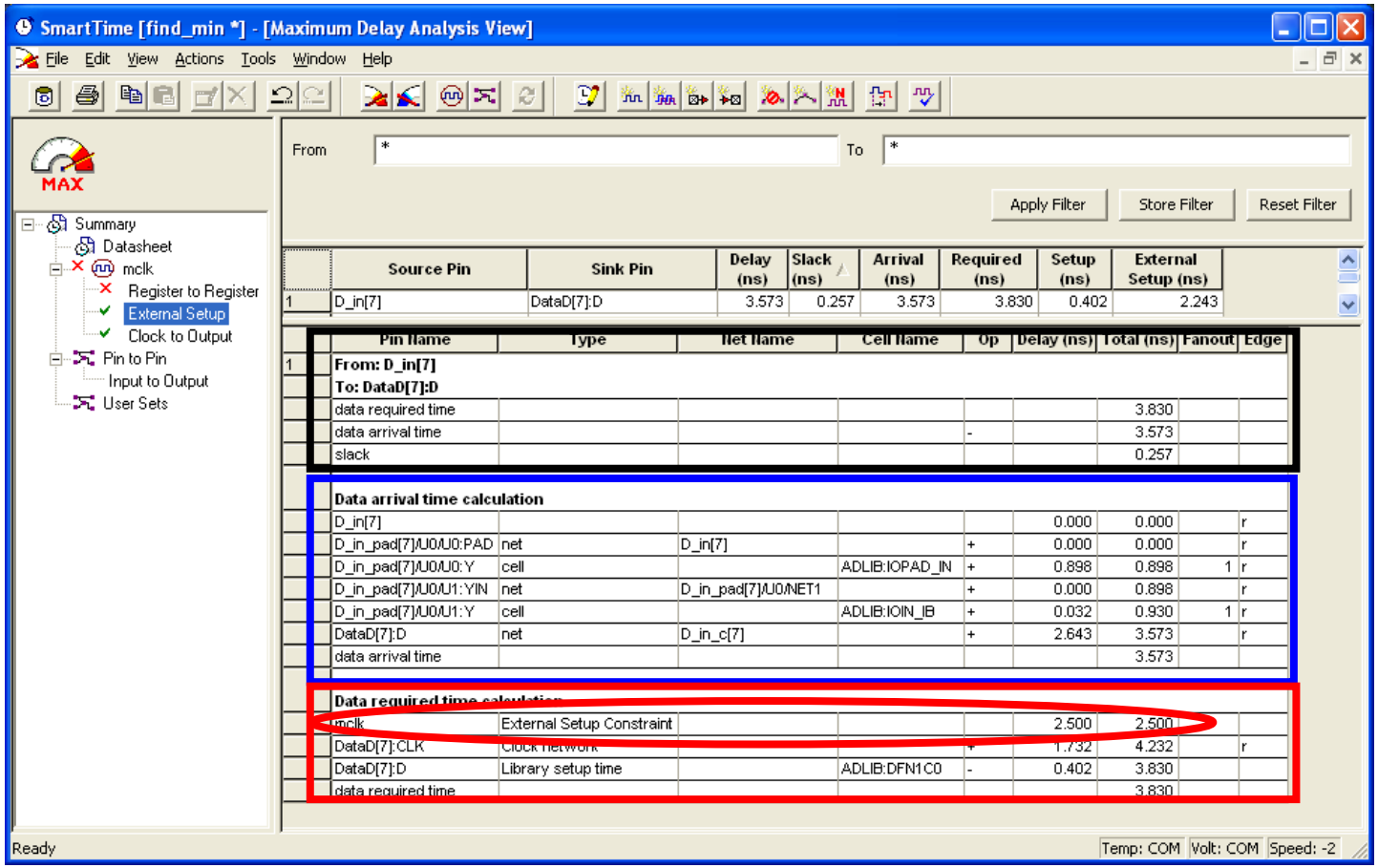
SmartTime External Setup Check

Using External Setup Constraint

- Enter Constraint as External Setup/Hold
  - Delay is referenced to *capture* edge of clock



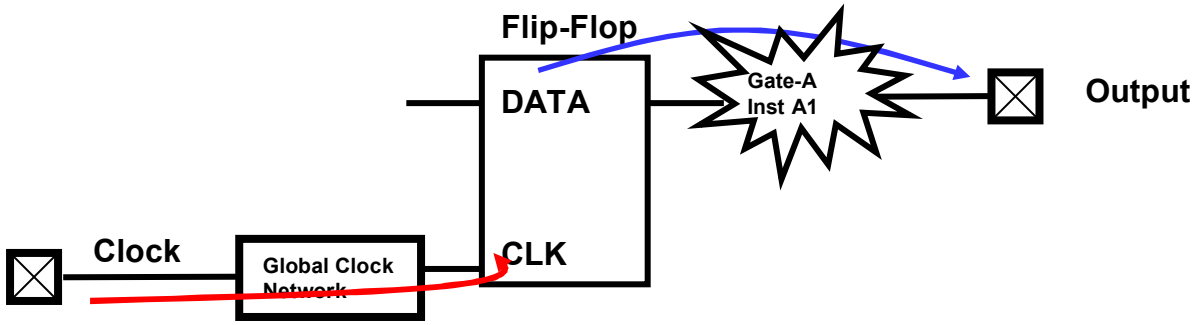
External Setup Check w/ External Setup Constraint



- External Setup Check Calculation
  - Arrival time = Launch edge (0) + max input delay + max Data path
  - Required time = External Setup + min Clock to FF1 – Setup of FF1
  - Slack = Required – Arrival = Violation if < 0

SmartTime
Clock to Output Check

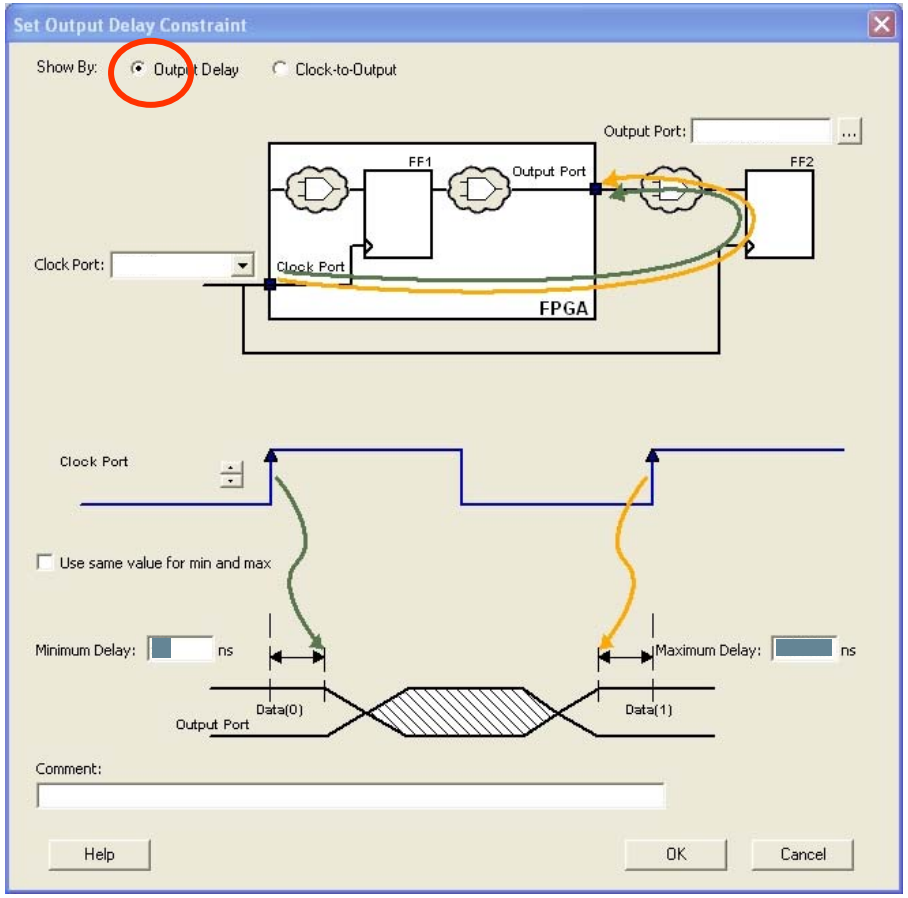
- Clock to Output is Displayed in the Maximum Delay Analysis View for Each Clock Domain



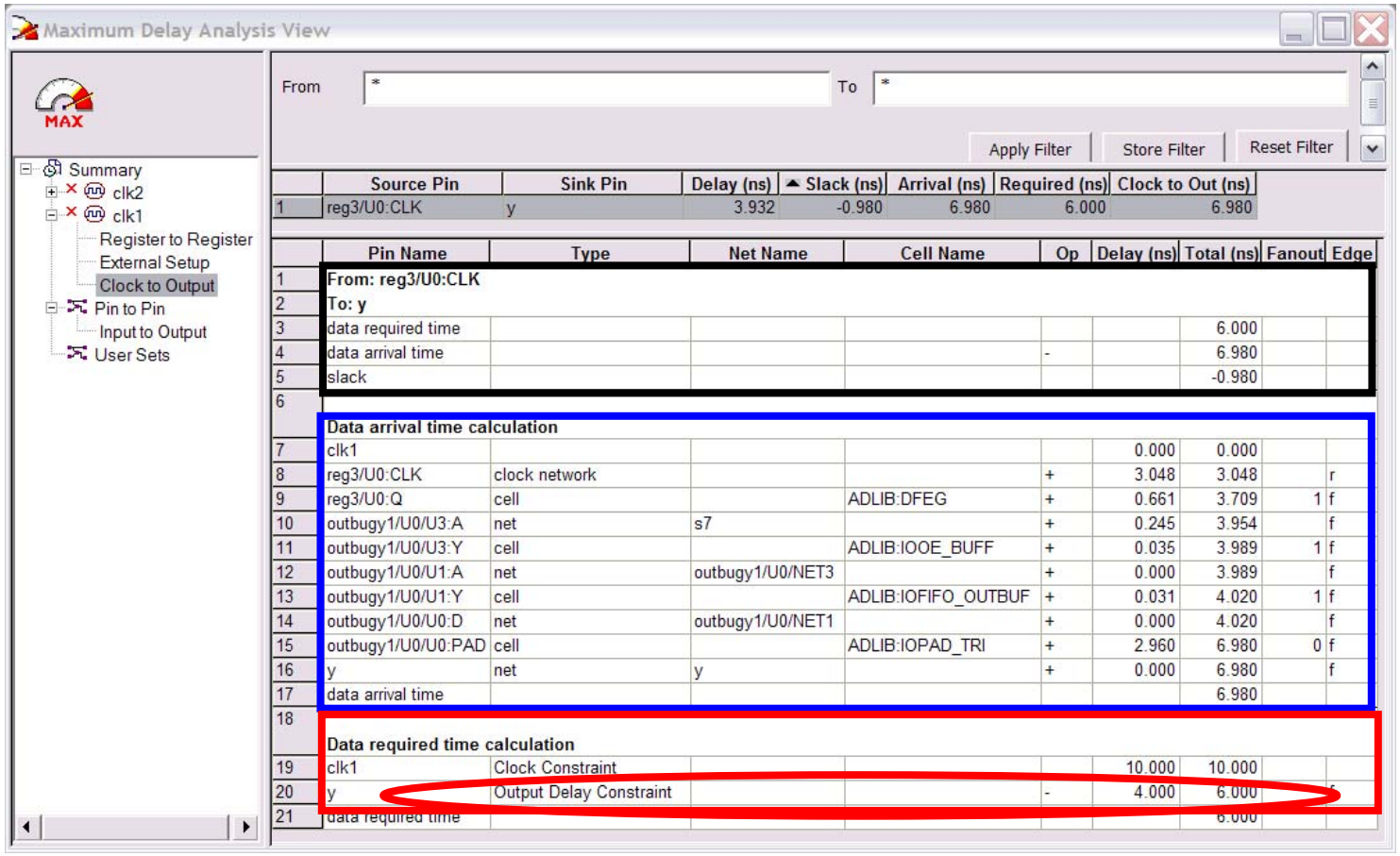
- Clock to Output Calculation
  - Arrival time = Launch edge (0) + max Data path
  - Required time = Capture edge (T)
  - Slack = Required – Arrival = Violation if < 0

Clock to Output Check w/ Output Delay Constraint

- Enter Constraint as Output Delay



Clock to Output w/ Output Delay Constraint



	Source Pin	Sink Pin	Delay (ns)	Slack (ns)	Arrival (ns)	Required (ns)	Clock to Out (ns)
1	reg3/U0:CLK	y	3.932	-0.980	6.980	6.000	6.980

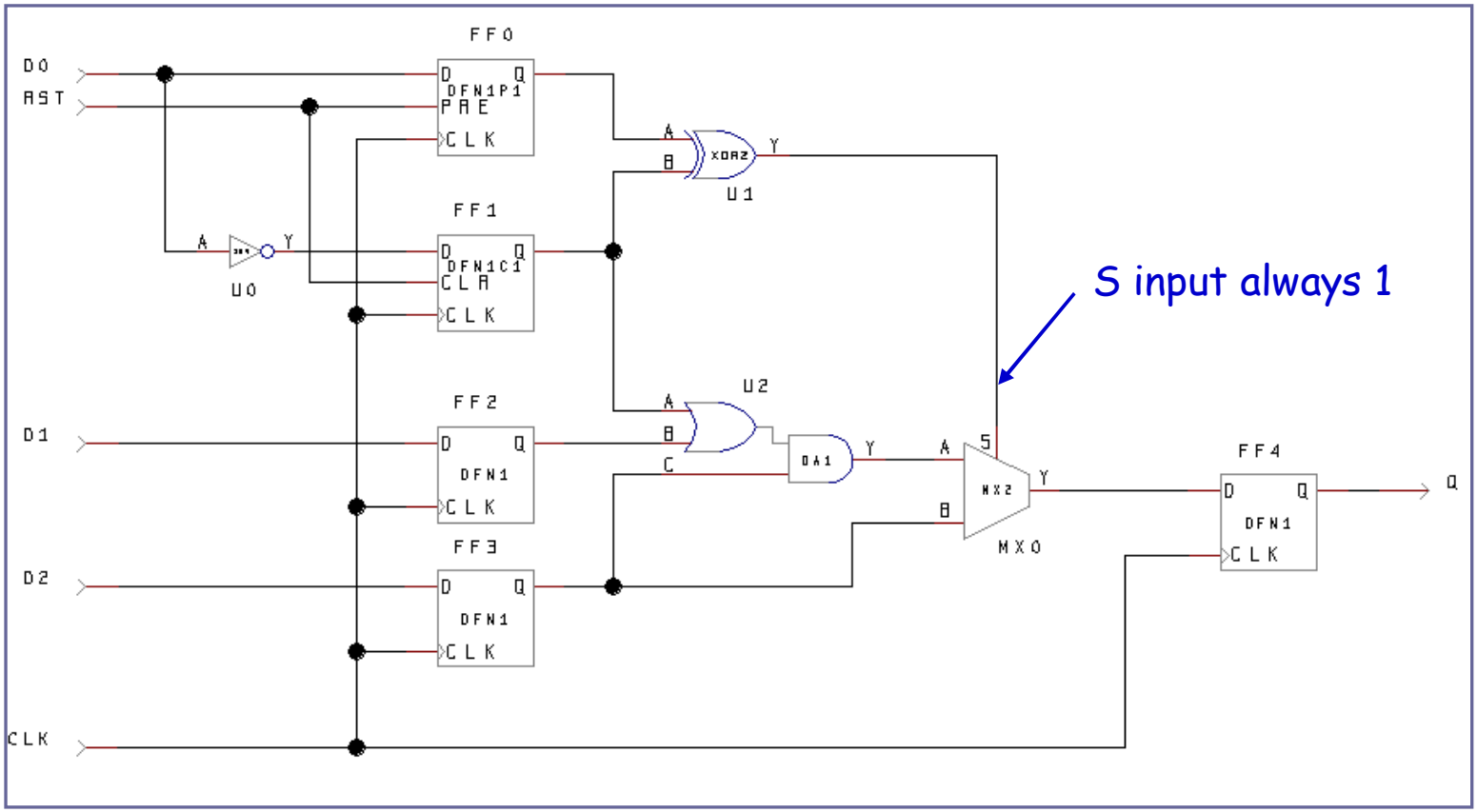
	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	From: reg3/U0:CLK								
2	To: y								
3	data required time						6.000		
4	data arrival time				-		6.980		
5	slack						-0.980		
6									
	Data arrival time calculation								
7	clk1					0.000	0.000		
8	reg3/U0:CLK	clock network			+	3.048	3.048		r
9	reg3/U0:Q	cell		ADLIB:DFEG	+	0.661	3.709	1	f
10	outbugy1/U0/U3:A	net	s7		+	0.245	3.954		f
11	outbugy1/U0/U3:Y	cell		ADLIB:IOOE_BUFF	+	0.035	3.989	1	f
12	outbugy1/U0/U1:A	net	outbugy1/U0/NET3		+	0.000	3.989		f
13	outbugy1/U0/U1:Y	cell		ADLIB:IOFIFO_OUTBUF	+	0.031	4.020	1	f
14	outbugy1/U0/U0:D	net	outbugy1/U0/NET1		+	0.000	4.020		f
15	outbugy1/U0/U0:PAD	cell		ADLIB:IOPAD_TRI	+	2.960	6.980	0	f
16	y	net	y		+	0.000	6.980		f
17	data arrival time						6.980		
18									
	Data required time calculation								
19	clk1	Clock Constraint				10.000	10.000		
20	y	Output Delay Constraint			-	4.000	6.000		f
21	data required time						6.000		

- Setup Check
  - Arrival time = Launch edge (0) + max Data path
  - Required time = Capture edge (T) – Output Delay
  - Slack = Required – Arrival = Violation if < 0

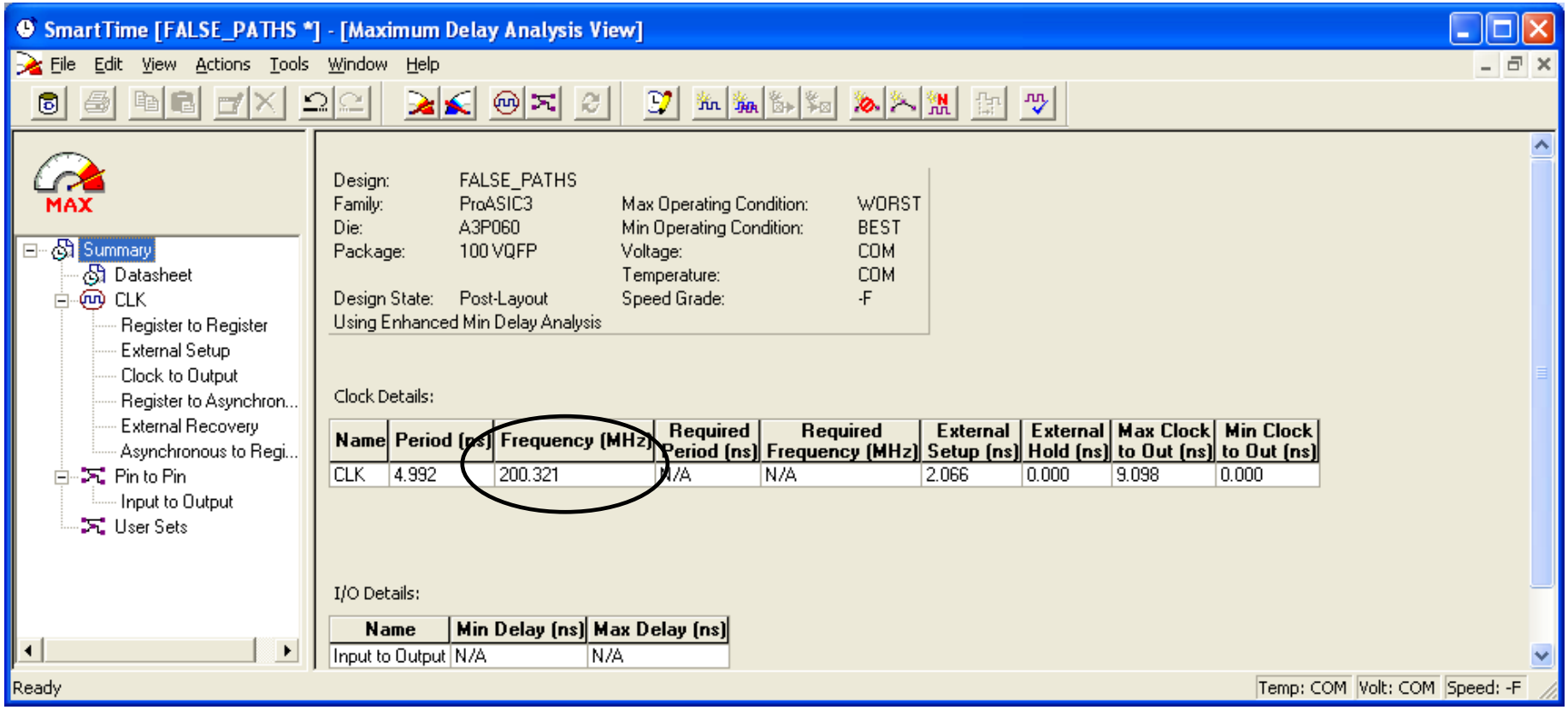
Adding Timing Exceptions

False Path Example

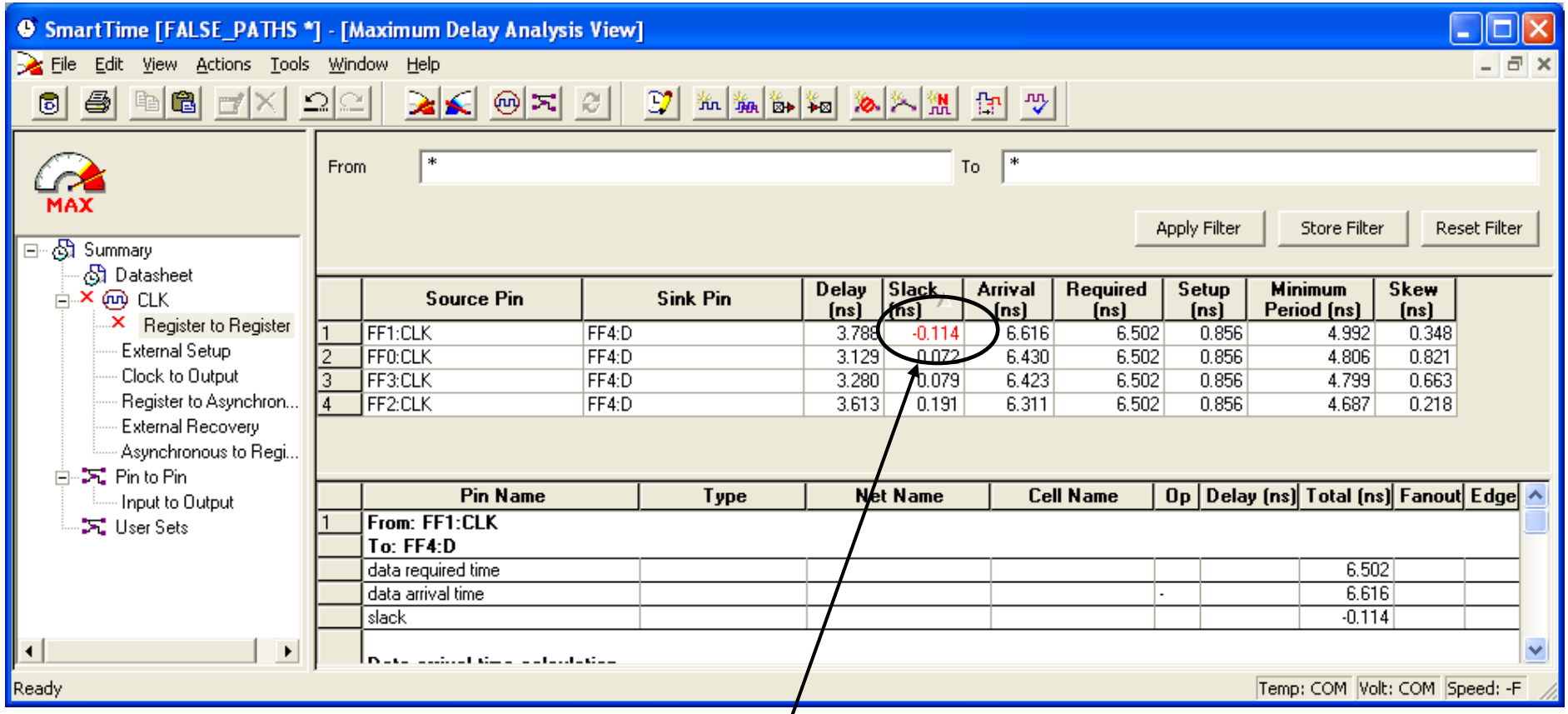
- Multiplexer S Input Never Changes
  - FF1, FF2 and FF3 to FF4 through MX0:A are False Paths



SmartTime
Maximum Frequency



SmartTime
Timing Analysis

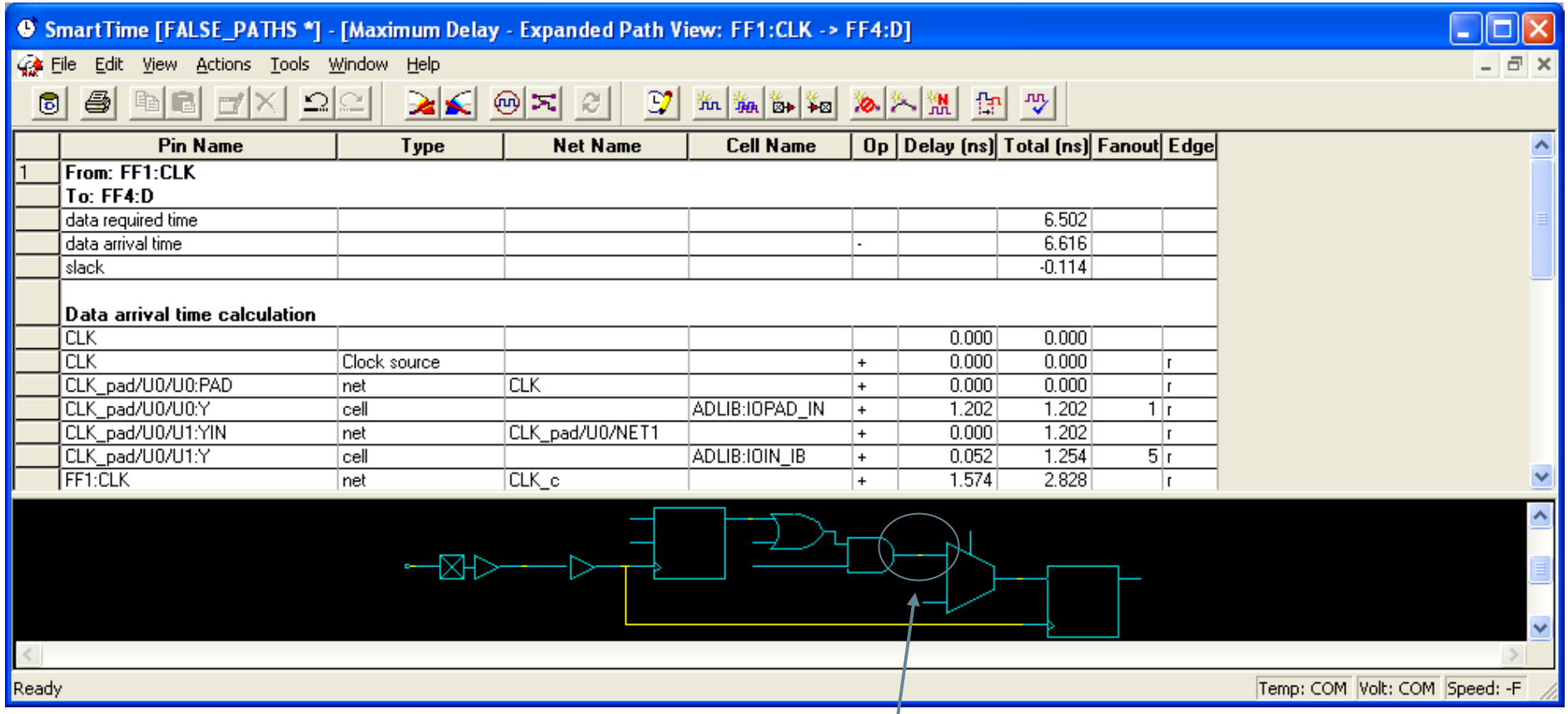


	Source Pin	Sink Pin	Delay (ns)	Slack (ns)	Arrival (ns)	Required (ns)	Setup (ns)	Minimum Period (ns)	Skew (ns)
1	FF1:CLK	FF4:D	3.788	-0.114	6.616	6.502	0.856	4.992	0.348
2	FF0:CLK	FF4:D	3.129	0.072	6.430	6.502	0.856	4.806	0.821
3	FF3:CLK	FF4:D	3.280	0.079	6.423	6.502	0.856	4.799	0.663
4	FF2:CLK	FF4:D	3.613	0.191	6.311	6.502	0.856	4.687	0.218

	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	**From: FF1:CLK**								
	To: FF4:D								
	data required time						6.502		
	data arrival time				-		6.616		
	slack						-0.114		

Timing Violation

Timing Analysis
Expanded Path

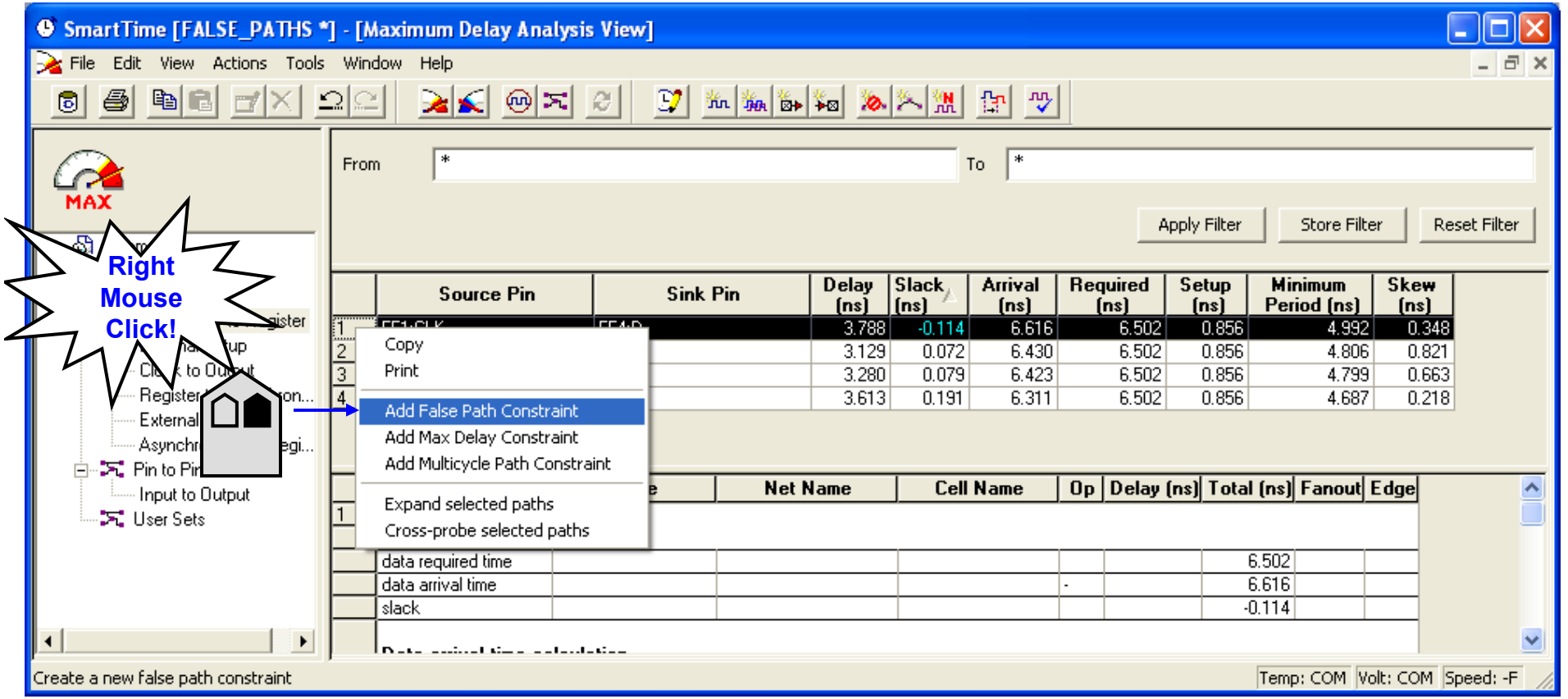


	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	**From: FF1:CLK**								
	To: FF4:D								
	data required time						6.502		
	data arrival time				-		6.616		
	slack						-0.114		
	Data arrival time calculation								
	CLK					0.000	0.000		
	CLK	Clock source			+	0.000	0.000		r
	CLK_pad/U0/U0:PAD	net	CLK		+	0.000	0.000		r
	CLK_pad/U0/U0:Y	cell		ADLIB:IOPAD_IN	+	1.202	1.202	1	r
	CLK_pad/U0/U1:YIN	net	CLK_pad/U0/NET1		+	0.000	1.202		r
	CLK_pad/U0/U1:Y	cell		ADLIB:IOIN_IB	+	0.052	1.254	5	r
	FF1:CLK	net	CLK_c		+	1.574	2.828		r

This is a false path

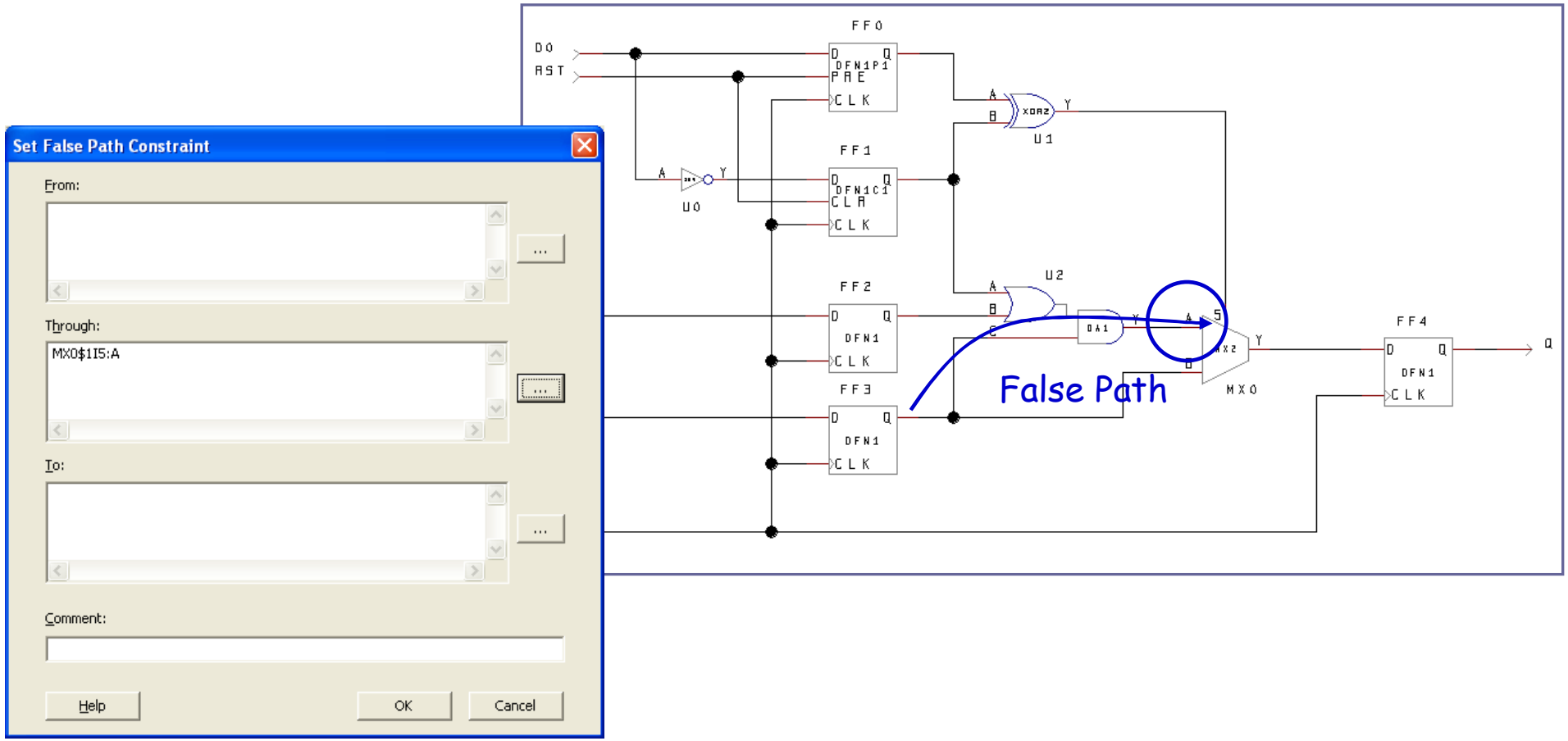
SmartTime

Entering False Path Exceptions



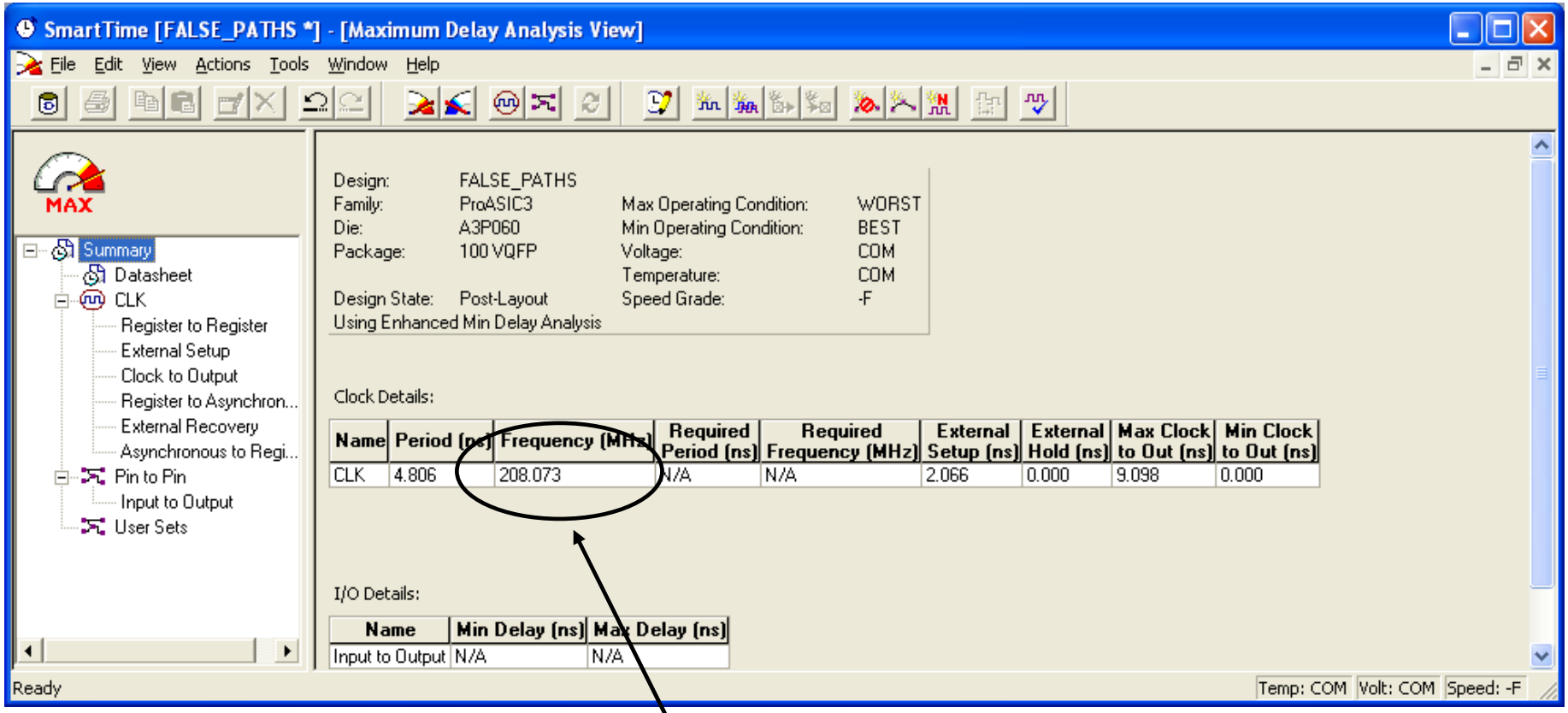
Add False Path Exception

- False Path Constraint Through MX0:A



SmartTime

Maximum Frequency with False Paths

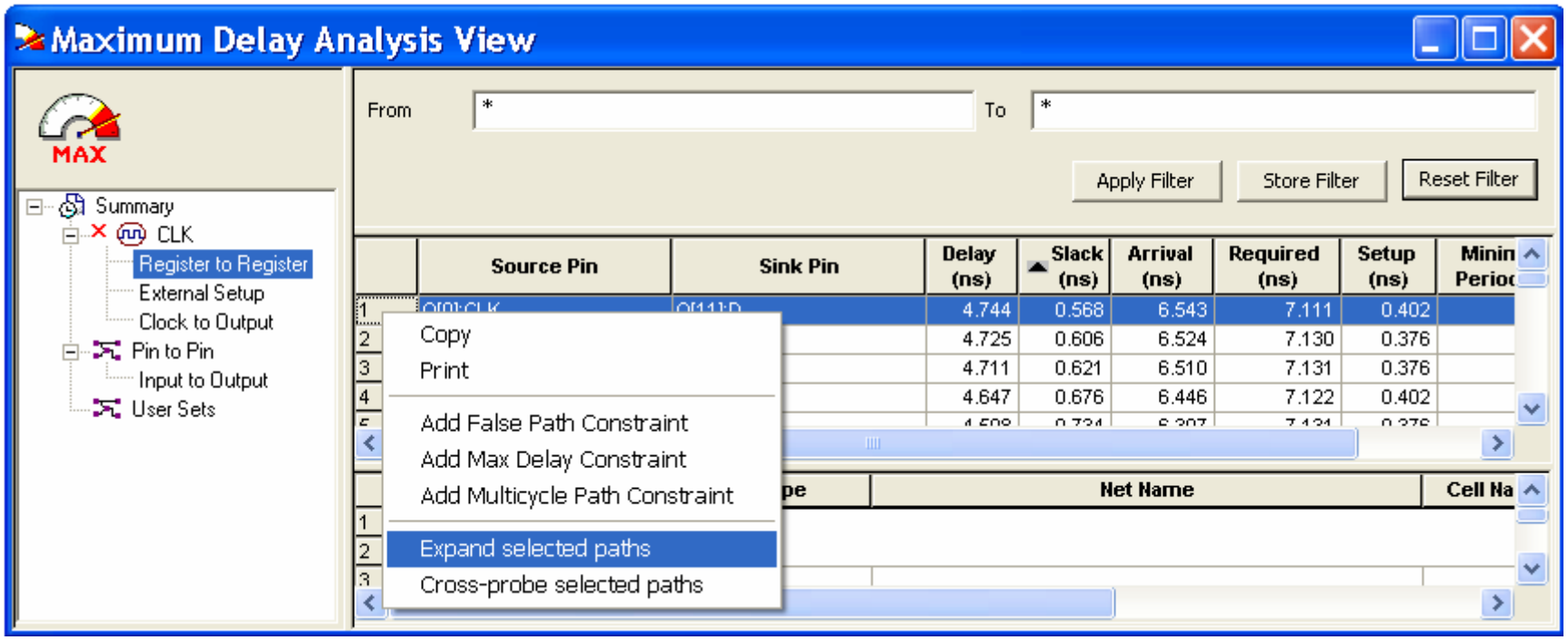


Actual Operating Frequency

SmartTime

Expanded Path Timing Information

- Right-click a Path in the Path List Displays the Expanded Path in a Separate Window

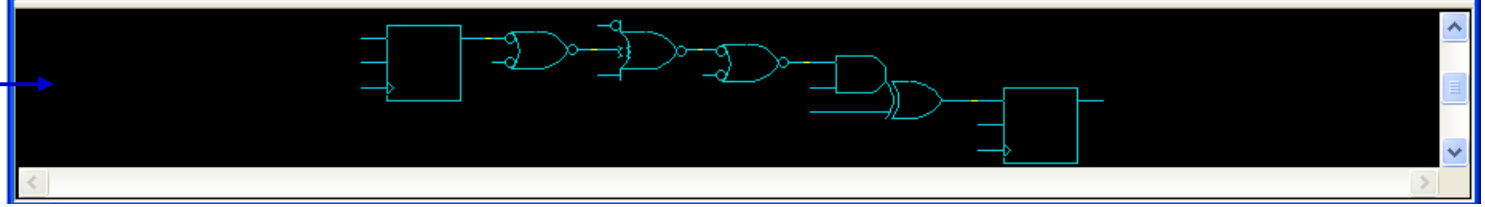


SmartTime
Expanded Path (cont.)

- Expanded Path View Includes a Schematic of the Path

Maximum Delay - Expanded Path View: Q[0]:CLK -> Q[11]:D

	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	**From: Q[0]:CLK**								
2	**To: Q[11]:D**								
3	data required time						7.111		
4	data arrival time				-		6.543		
5	slack						0.568		
6	**Data arrival time calculation**								
7	CLK					0.000	0.000		
8	Q[0]:CLK	clock network			+	1.799	1.799		r
9	Q[0]:Q	cell		ADLIB:DFN1C1	+	0.535	2.334	4	f
10	COUNTING_count_3_G_1_5:A	net	Q_c[0]		+	0.655	2.989		f
11	COUNTING_count_3_G_1_5:Y	cell		ADLIB:NOR2B	+	0.380	3.369	6	f
12	COUNTING_count_3_G_1_3:B	net	COUNTING_count_3_DWACT_ADD_CI_0_tmp[0]		+	0.288	3.657		f
13	COUNTING_count_3_G_1_3:Y	cell		ADLIB:NOR3B	+	0.458	4.115	3	f
14	COUNTING_count_3_I_82:A	net	COUNTING_count_3_DWACT_ADD_CI_0_g_array_3[0]		+	0.364	4.479		f
15	COUNTING_count_3_I_82:Y	cell		ADLIB:NOR2B	+	0.380	4.859	2	f
16	COUNTING_count_3_I_61:A	net	COUNTING_count_3_DWACT_ADD_CI_0_g_array_11_1[0]		+	0.760	5.619		f
17	COUNTING_count_3_I_61:Y	cell		ADLIB:AX1C	+	0.719	6.338	1	f
18	Q[11]:D	net	COUNTING_count_3[11]		+	0.205	6.543		f
19	data arrival time						6.543		
20	**Data required time calculation**								
21	CLK	Clock Constraint				5.714	5.714		
22	Q[11]:CLK	clock network			+	1.799	7.513		r
23	Q[11]:D	Library setup		ADLIB:DFN1C1	-	0.402	7.111		
24	data required time						7.111		



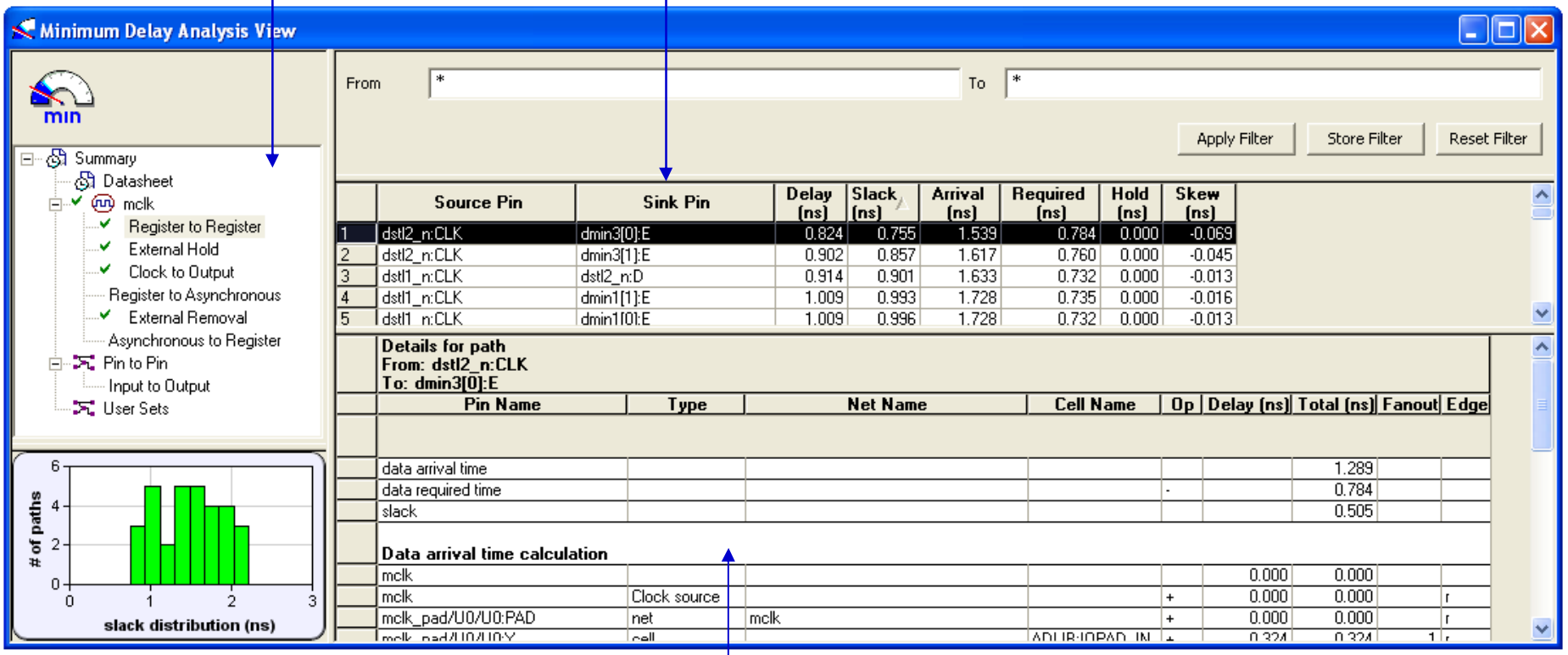
Right click in window to zoom

Timing Analysis with SmartTime

- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

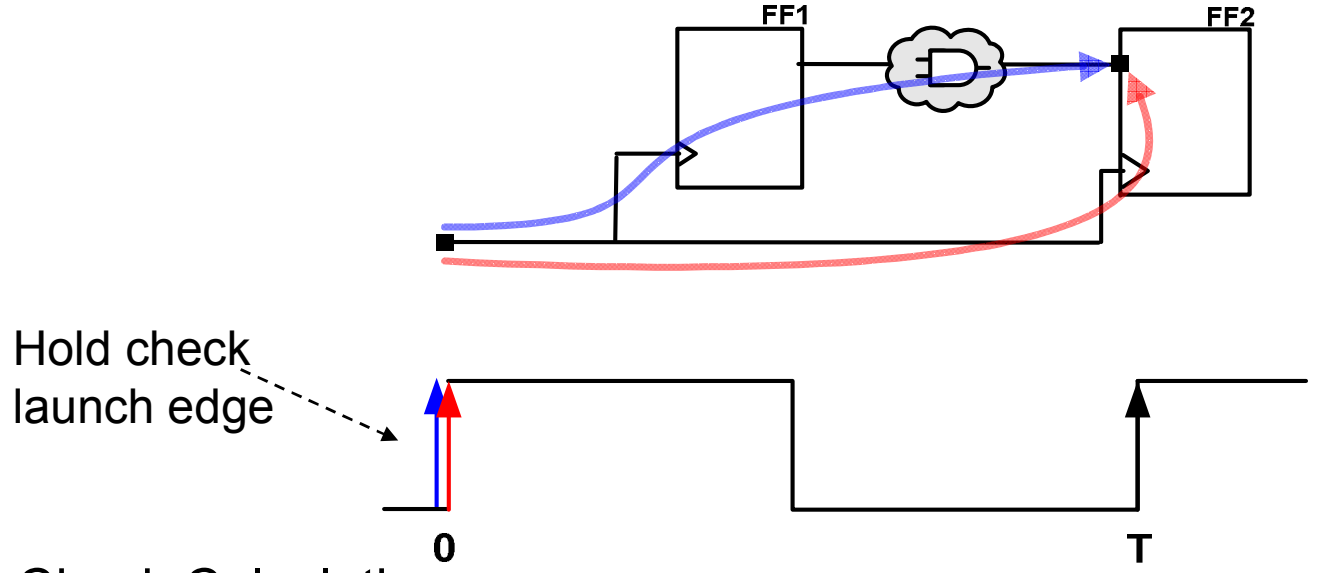
Minimum Delay Analysis View
Register to Register Path Set

- Identify Internal Hold Violations



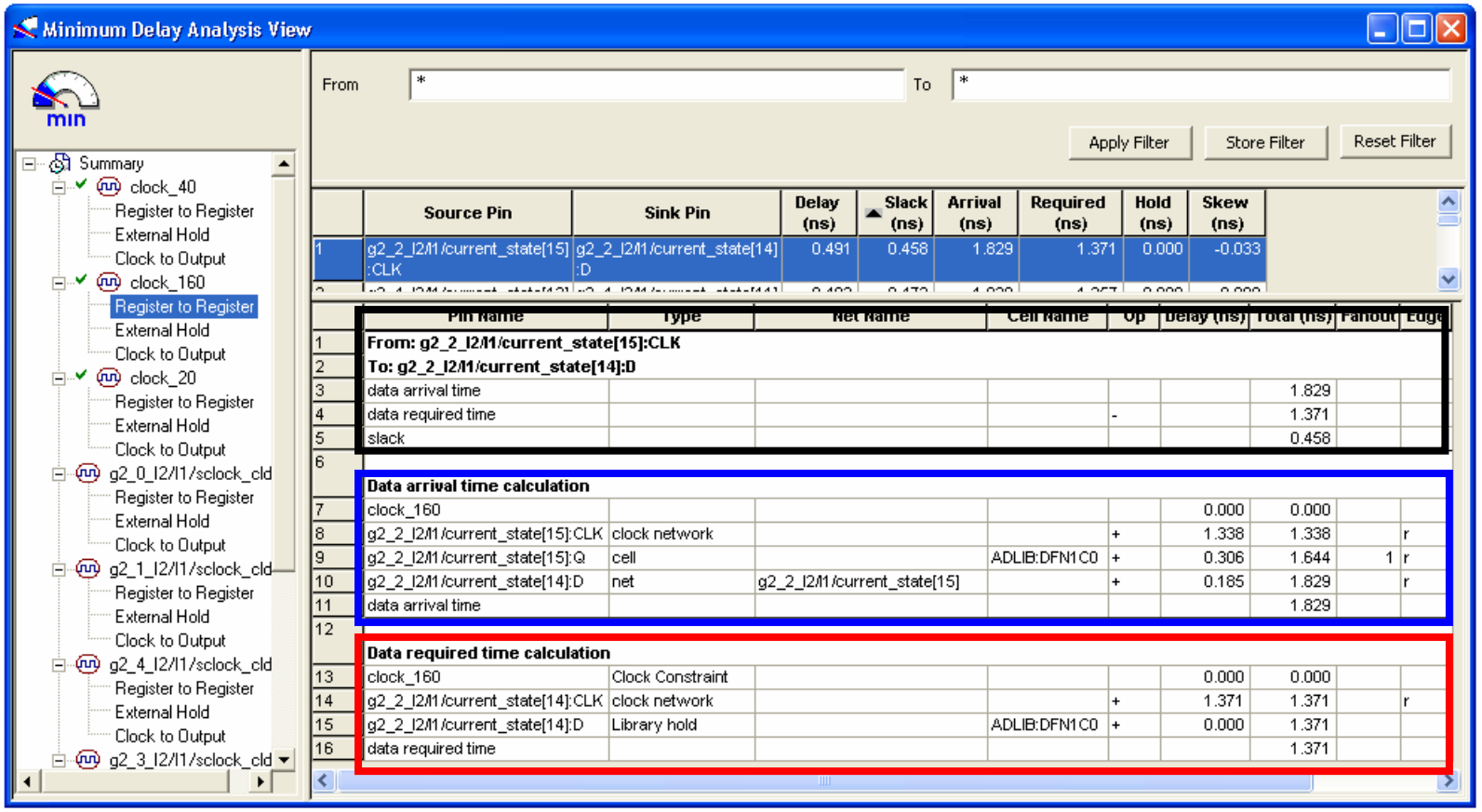
SmartTime
Hold Check

- Hold Check is Displayed in the Minimum Delay Analysis View



- Hold Check Calculation
  - Arrival time = Launch edge (0) + min Clock to FF1 + min Data path
  - Required time = Launch edge (0) + max Clock to FF2 + Hold of FF2
  - Slack = Arrival – Required = Violation if < 0

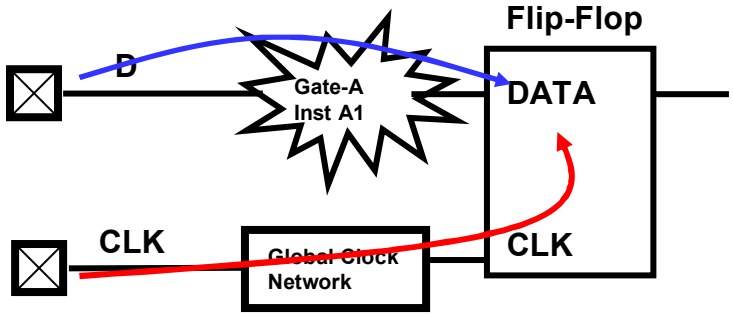
SmartTime
Hold Check (cont.)

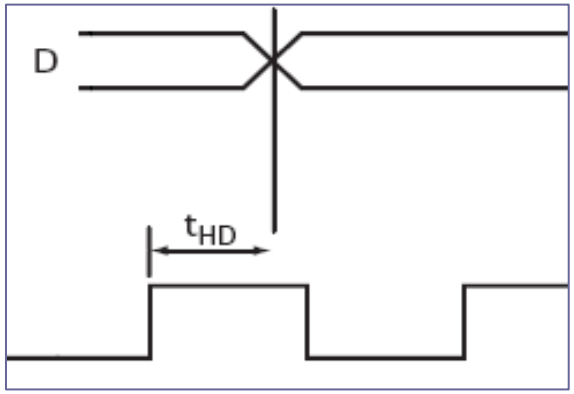


- Hold Check
  - Arrival time = Launch edge (0) + min Clock to FF1 + min Data path
  - Required time = Capture edge (0) + max Clock to FF2 + Hold of FF2
  - Slack = Arrival – Required = Violation if < 0

SmartTime
External Hold Check

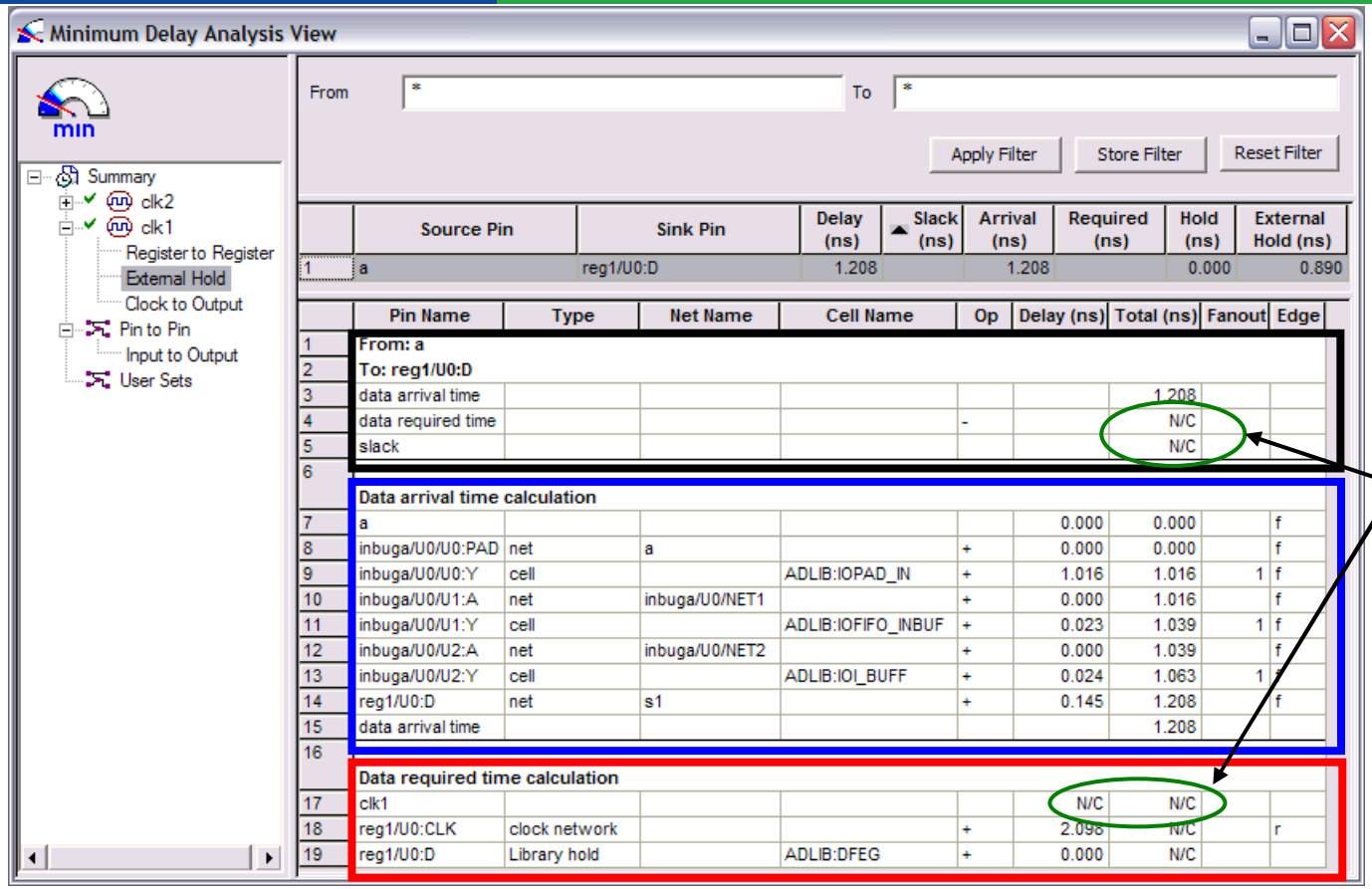
- External Hold Defines the Timing Requirements at the Input Pins
- External Hold Check is Displayed in the Minimum Delay Analysis View for Each Clock Domain





- External Hold Check Calculation
  - Arrival time = min Input Pad to FF1
  - Required time = max Clock to FF1 + Hold of FF1
  - External Hold = Arrival - Required

SmartTime
External Hold Check



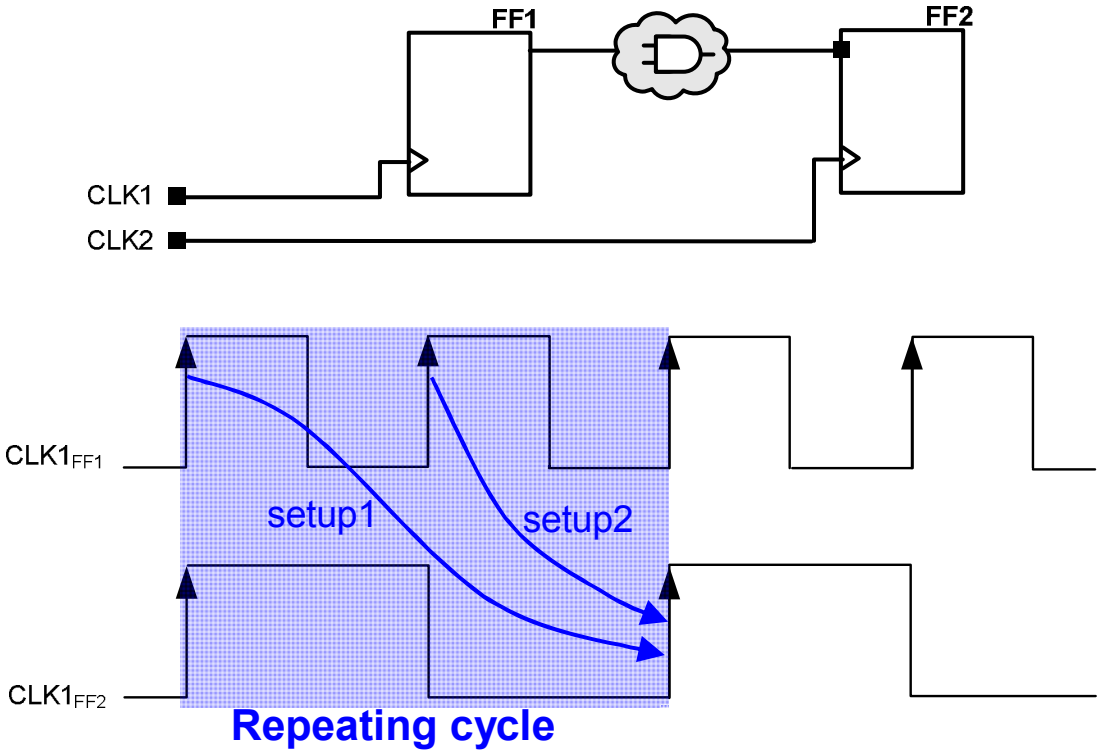
Slack cannot be calculated if clock constraint is not entered

- External Hold Check – Minimum Delay Analysis View
  - Arrival time = min Input Pad to FF1
  - Required time = max Clock to FF1 + Hold of FF1
  - Slack = Arrival – Required = Violation if < 0

Timing Analysis with SmartTime

- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

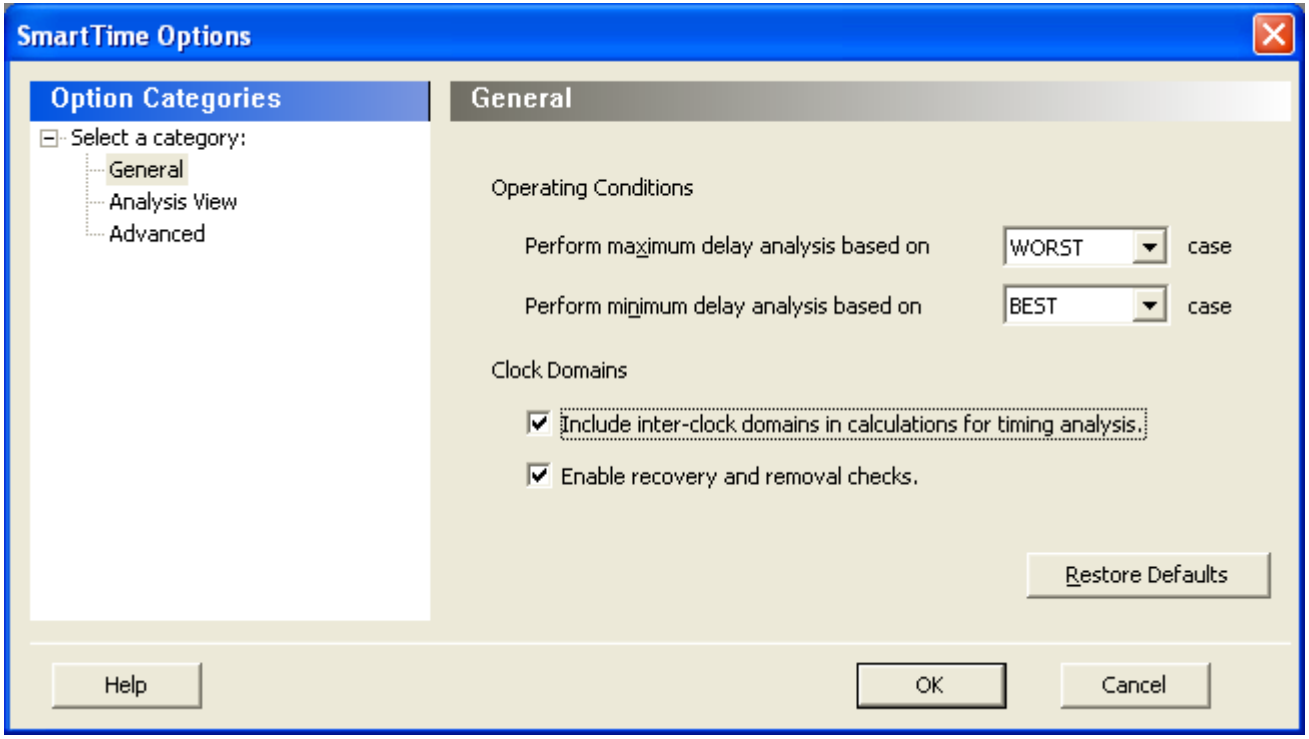
SmartTime
Cross-Clock Domains



- SmartTime Performs Setup Check Across Clock Domains
  - SmartTime Looks at the Relationship Between the Active Clock Edges Over a Full Repeating Cycle, Equal to the Least Common Multiple of the Two Clock Periods
  - For Each Capture Edge at the Destination Flip-flop, SmartTime Assumes That the Corresponding Launch Edge Is the Nearest Source Clock Edge Occurring Before the Capture Edge

SmartTime
Cross-Clock Domain Analysis Activation

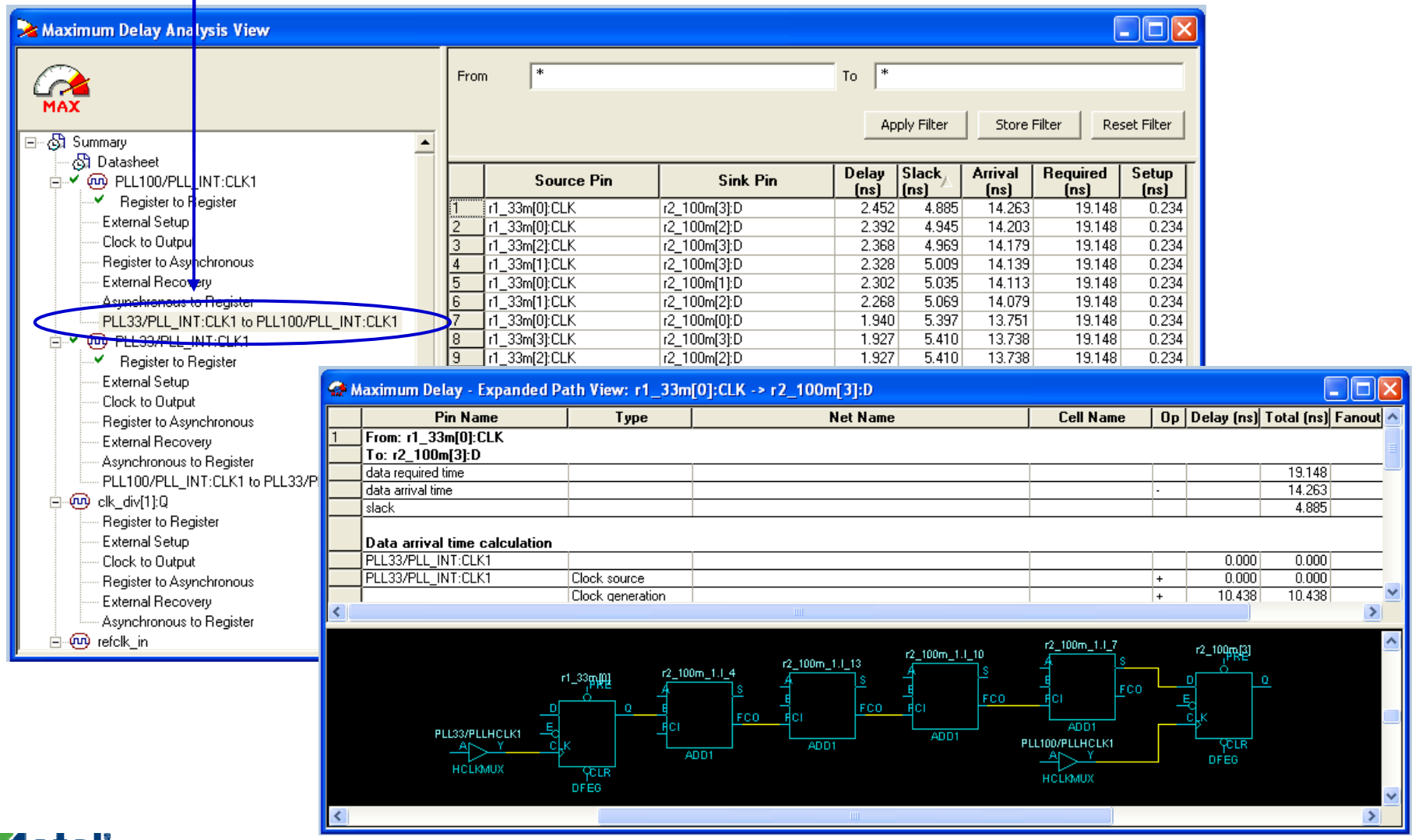
- Activate Cross-clock Domain Analysis for a Particular Clock Domain
  - SmartTime Automatically Detects All Other Clock Domains With Paths Ending at Selected Clock Domain



SmartTime

Cross-Clock Domains in SmartTime

Inter clock domain analysis

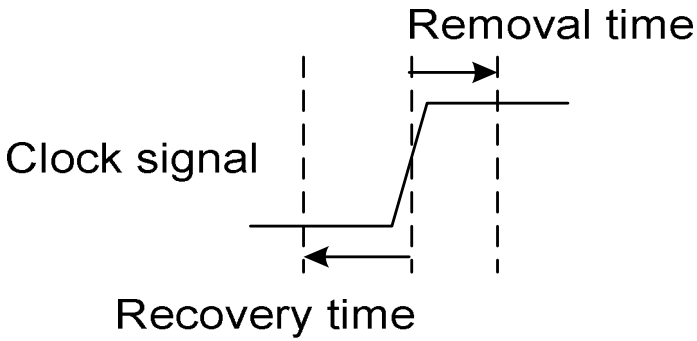


Timing Analysis with SmartTime

- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

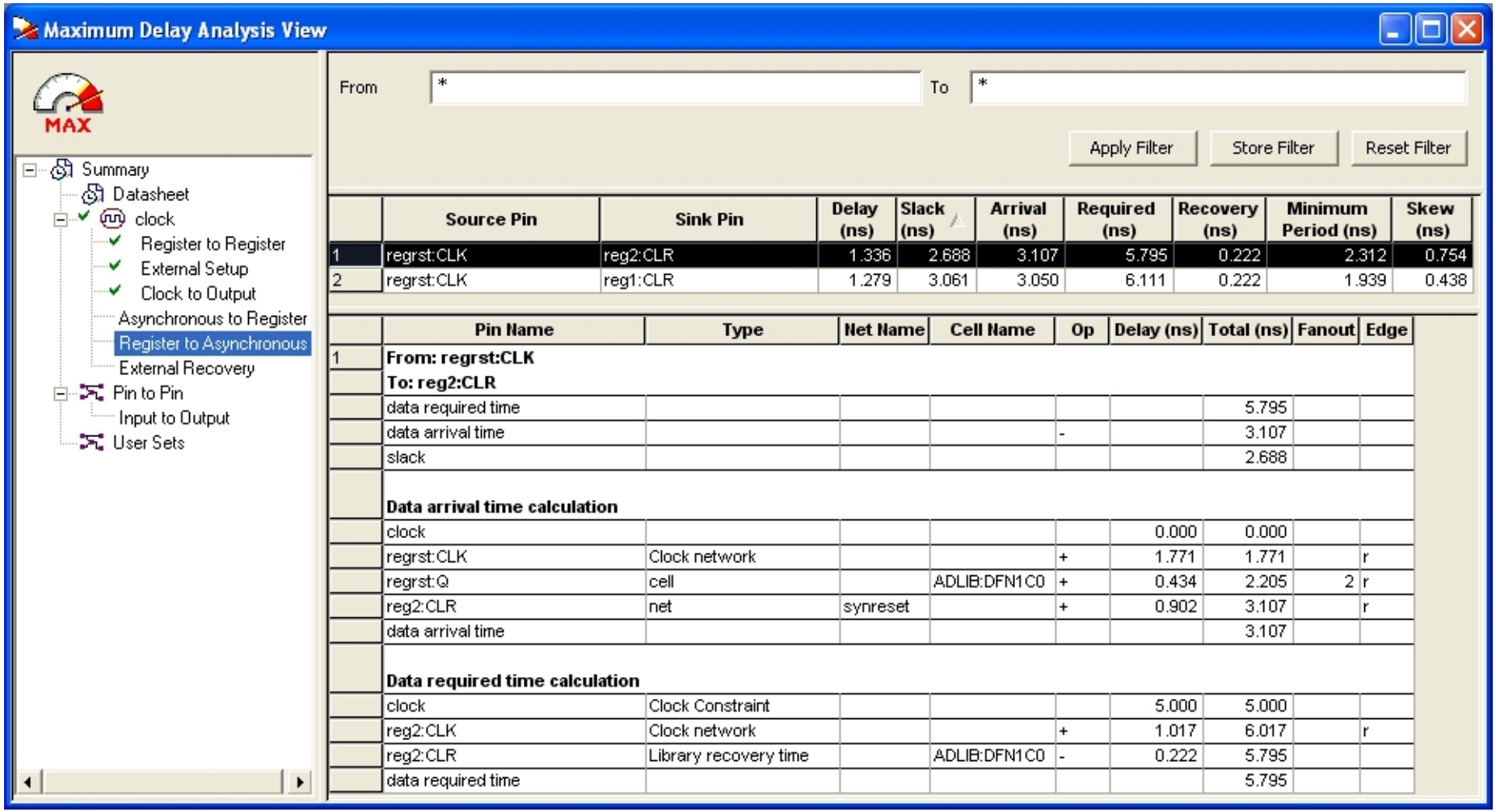
Removal/Recovery Checks

- Definition
  - An asynchronous signal should not be de-activated around the active clock edge



  - Recovery needs to be checked in Max delay analysis
  - Removal needs to be checked in Min delay analysis

Removal/Recovery Checks: Analysis



	Source Pin	Sink Pin	Delay (ns)	Slack (ns)	Arrival (ns)	Required (ns)	Recovery (ns)	Minimum Period (ns)	Skew (ns)
1	regrst:CLK	reg2:CLR	1.336	2.688	3.107	5.795	0.222	2.312	0.754
2	regrst:CLK	reg1:CLR	1.279	3.061	3.050	6.111	0.222	1.939	0.438

	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	**From: regrst:CLK**								
	To: reg2:CLR								
	data required time						5.795		
	data arrival time				-		3.107		
	slack						2.688		
	Data arrival time calculation								
	clock					0.000	0.000		
	regrst:CLK	Clock network			+	1.771	1.771		r
	regrst:Q	cell		ADLIB:DFN1C0	+	0.434	2.205	2	r
	reg2:CLR	net	synreset		+	0.902	3.107		r
	data arrival time						3.107		
	Data required time calculation								
	clock	Clock Constraint				5.000	5.000		
	reg2:CLK	Clock network			+	1.017	6.017		r
	reg2:CLR	Library recovery time		ADLIB:DFN1C0	-	0.222	5.795		
	data required time						5.795		

Timing Analysis with SmartTime

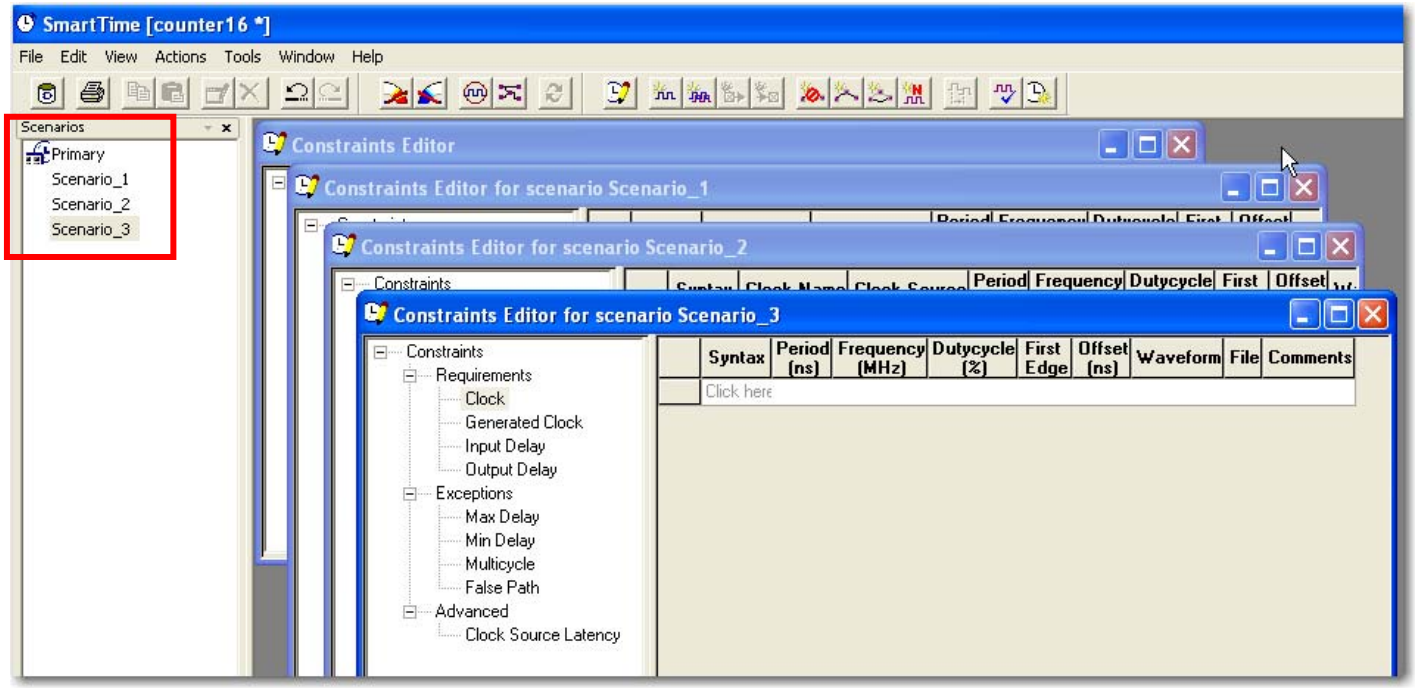
- Design Summary
- Maximum Delay Analysis View
- Minimum Delay Analysis View
- Cross Clock Domain Analysis
- Removal and Recovery Checks
- SmartTime Scenarios

SmartTime Scenarios

- Create constraint options for testing
  - Use for pre-layout timing
  - Use to drive TDPR
- Launch from Tools => Constraints Editor => New Scenario

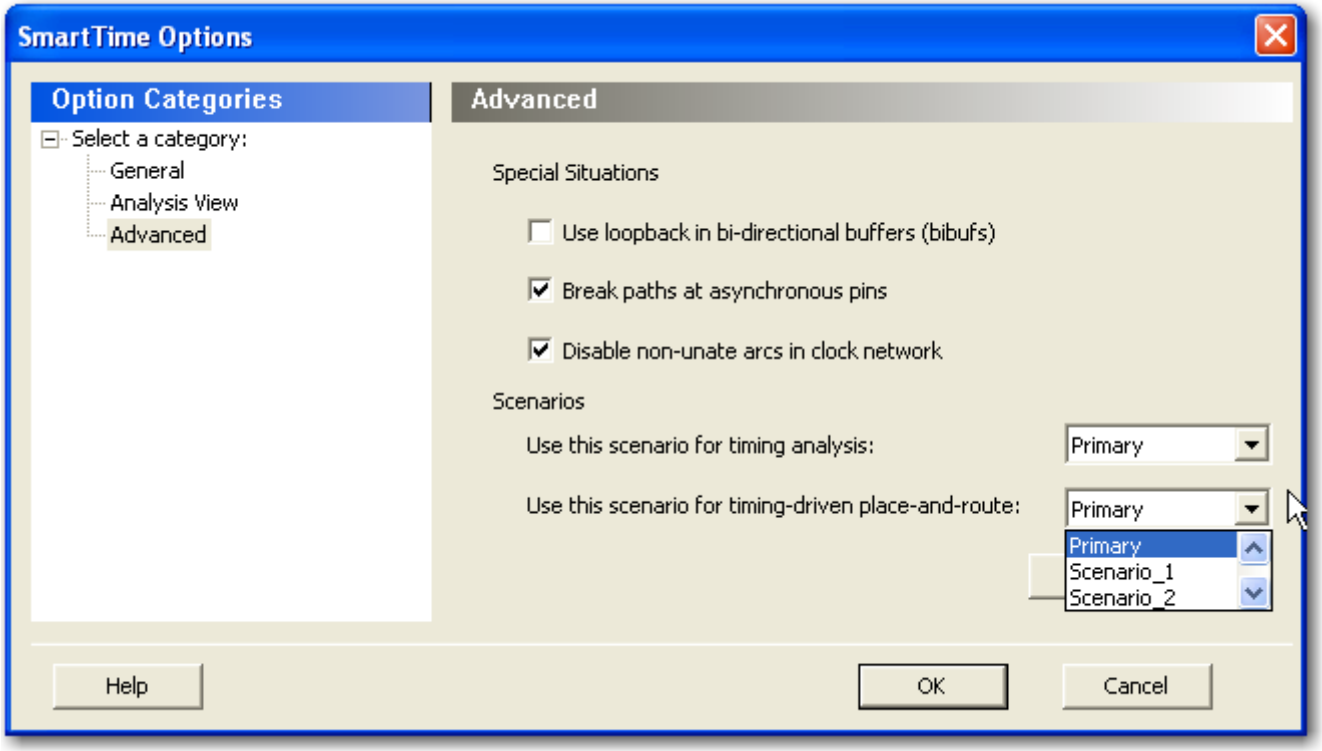
SmartTime Scenarios

- Brings up Timing Constraints Dialog = “Scenario_1”
  - Additional Scenarios can be created
- Use Constraints Wizard to Enter Constraints
  - Wizard adds constraints only to the active scenario
- Import SDC or Export Scenario with SDC
- Scenarios can be Added, Edited, Deleted, Copied



SmartTime Scenarios

- Use Options => Advanced => Scenarios
  - Use specific scenario for timing analysis
  - Use specific scenario for TDPR



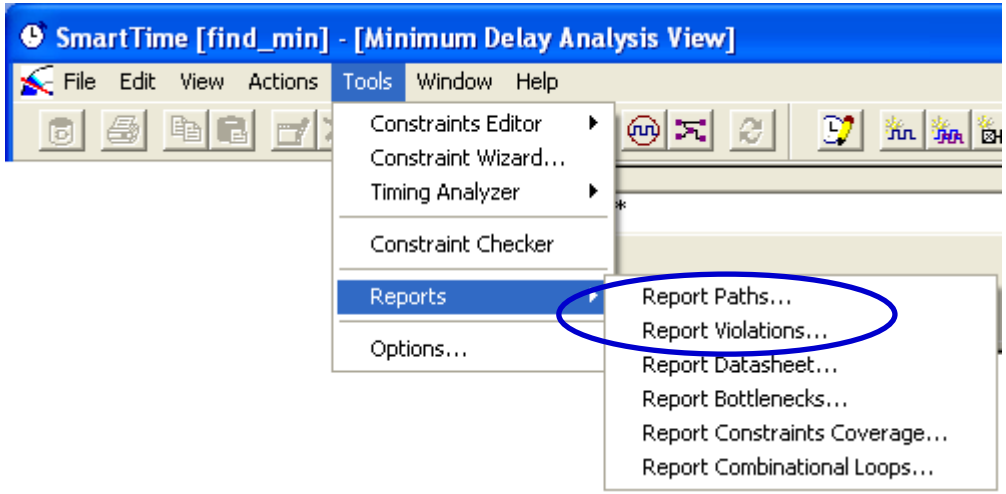
SmartTime Timing Reports

SmartTime
Timing Reports

- SmartTime Can Generate Five Types of Reports:
  - Timing Report
  - Timing Violation Report
  - Data Sheet Report
  - Bottle Neck Report
  - Constraints Coverage
  - Combinational Loops

SmartTime

Timing and Timing Violation Reports

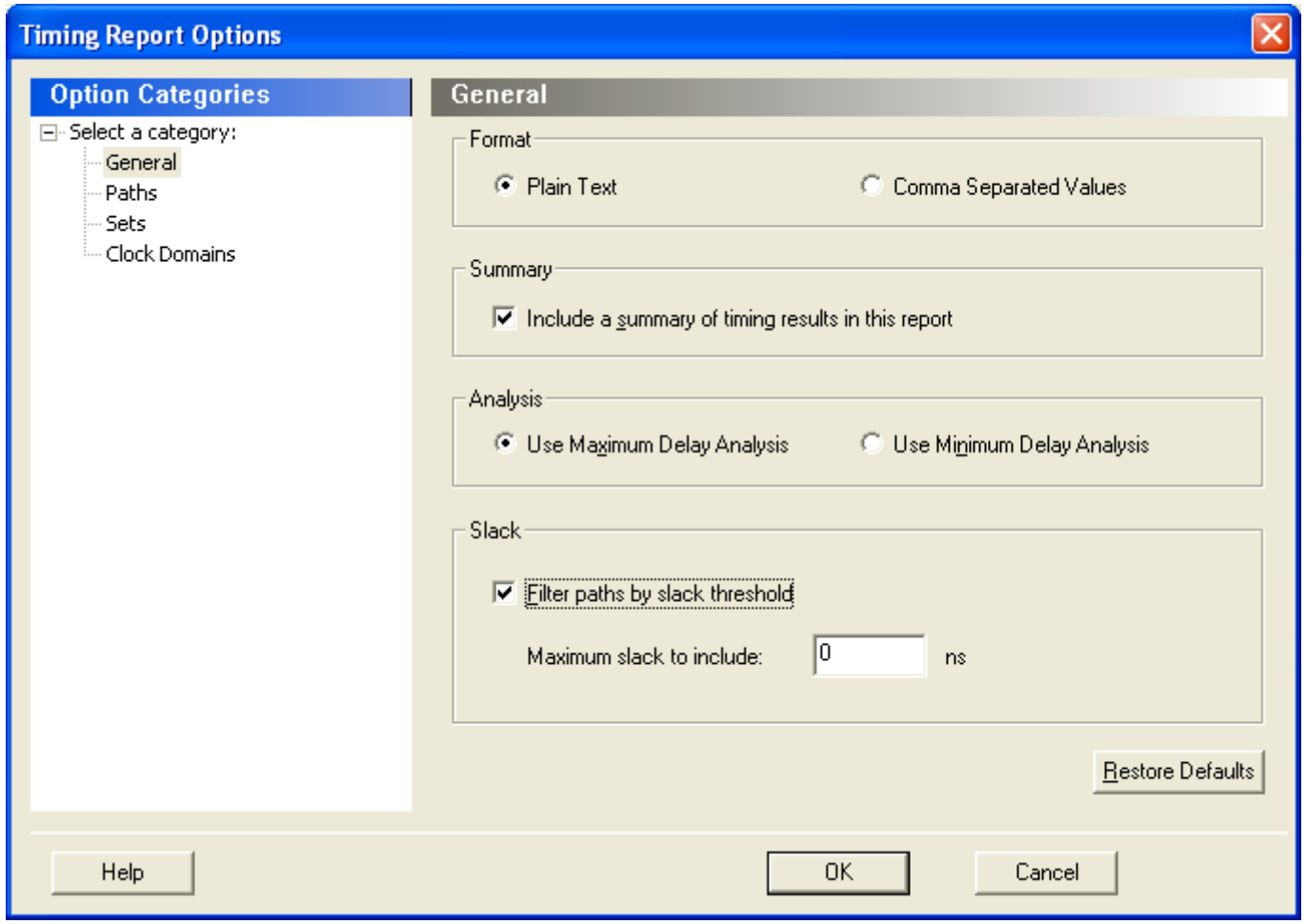


Timing Report Contains Timing Information of the Design in a Text Format

- Information Can Be Customized

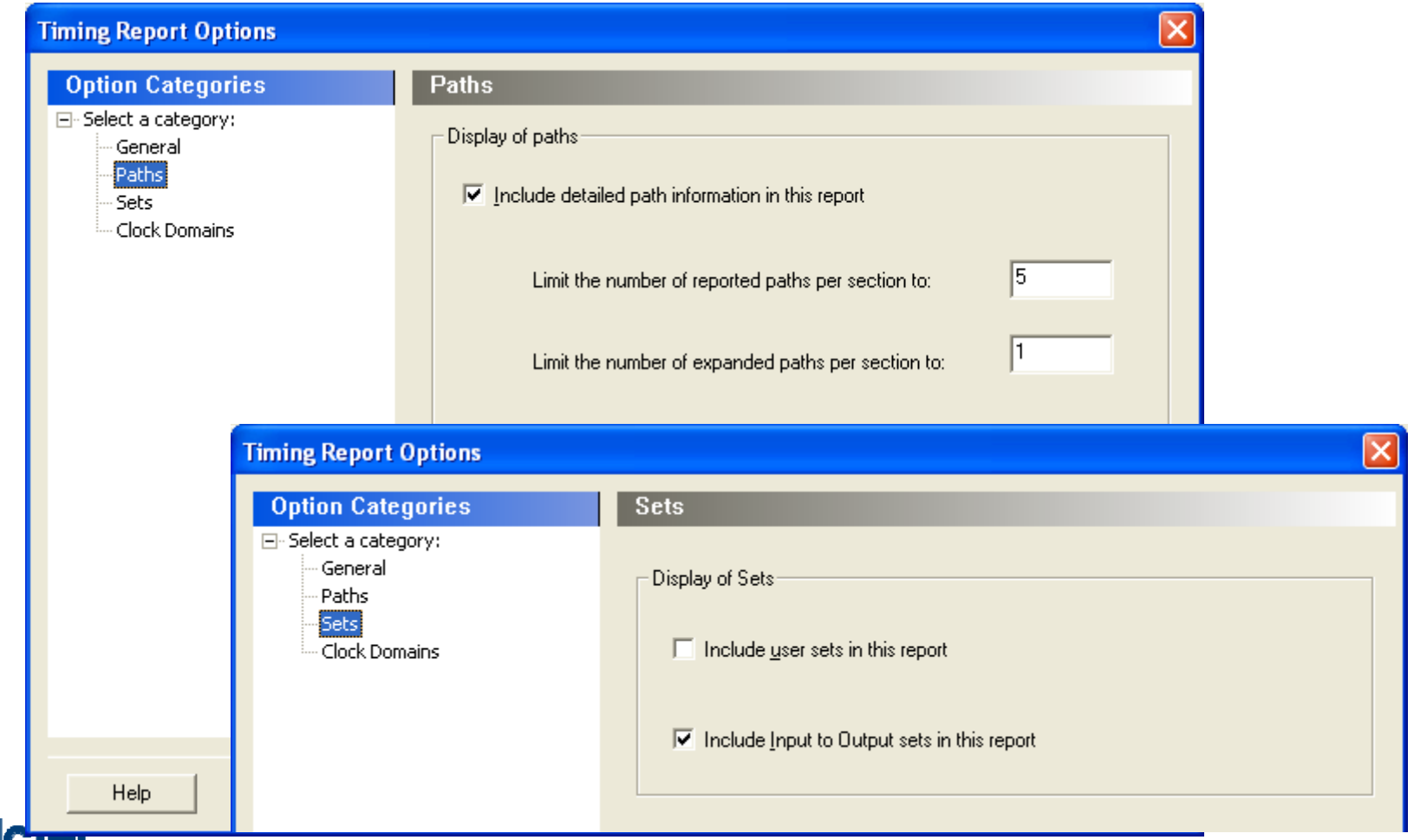
- Timing Violation Report Contains a Flat List of Paths With Timing Violations (No Breakdown by Clock Domain).

SmartTime
Timing Report Options



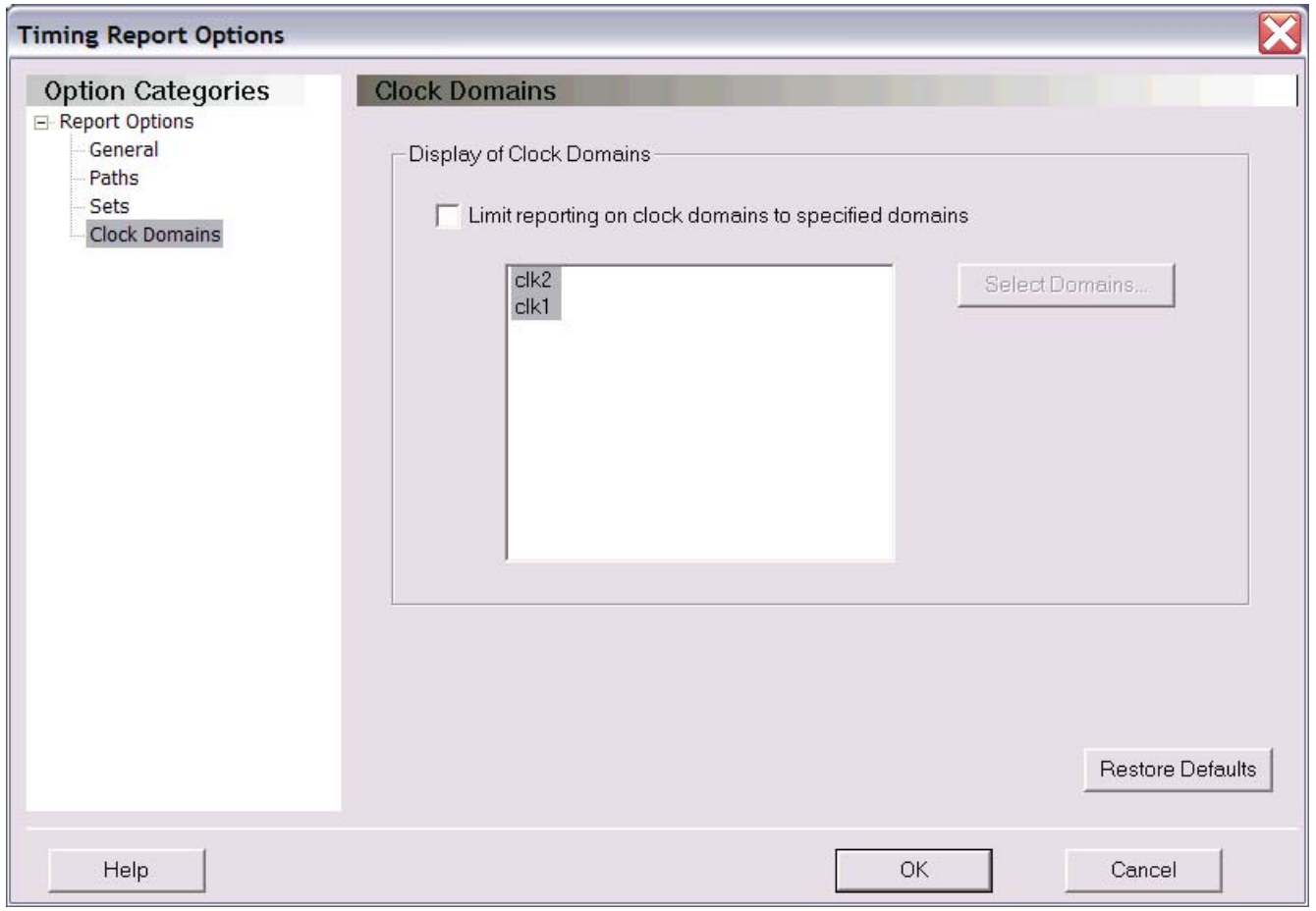
SmartTime
Timing Report Options

- Paths & Sets Options
  - Include User Defined Sets in Timing Report



SmartTime
Timing Report Options

- Clock Domains
  - Specify Clock Domains to be Included in the Timing Report



SmartTime
Timing Report Structure

- Header
  - Design Information, Operating Conditions…
- Summary
  - Data-sheet (Freq, ext setup/hold, min/max clock-to-out…)
- Clock Domain Details
  - Path Sets (reg2reg, in2reg, reg2out, Custom…)
    - Path Details
    - Expanded Paths
- Inter-clock Domain Details
- Pin-to-pin Path
  - Path Sets (in2out, Custom…)

SmartTime
Timing Report

COUNT16 - Timing Report

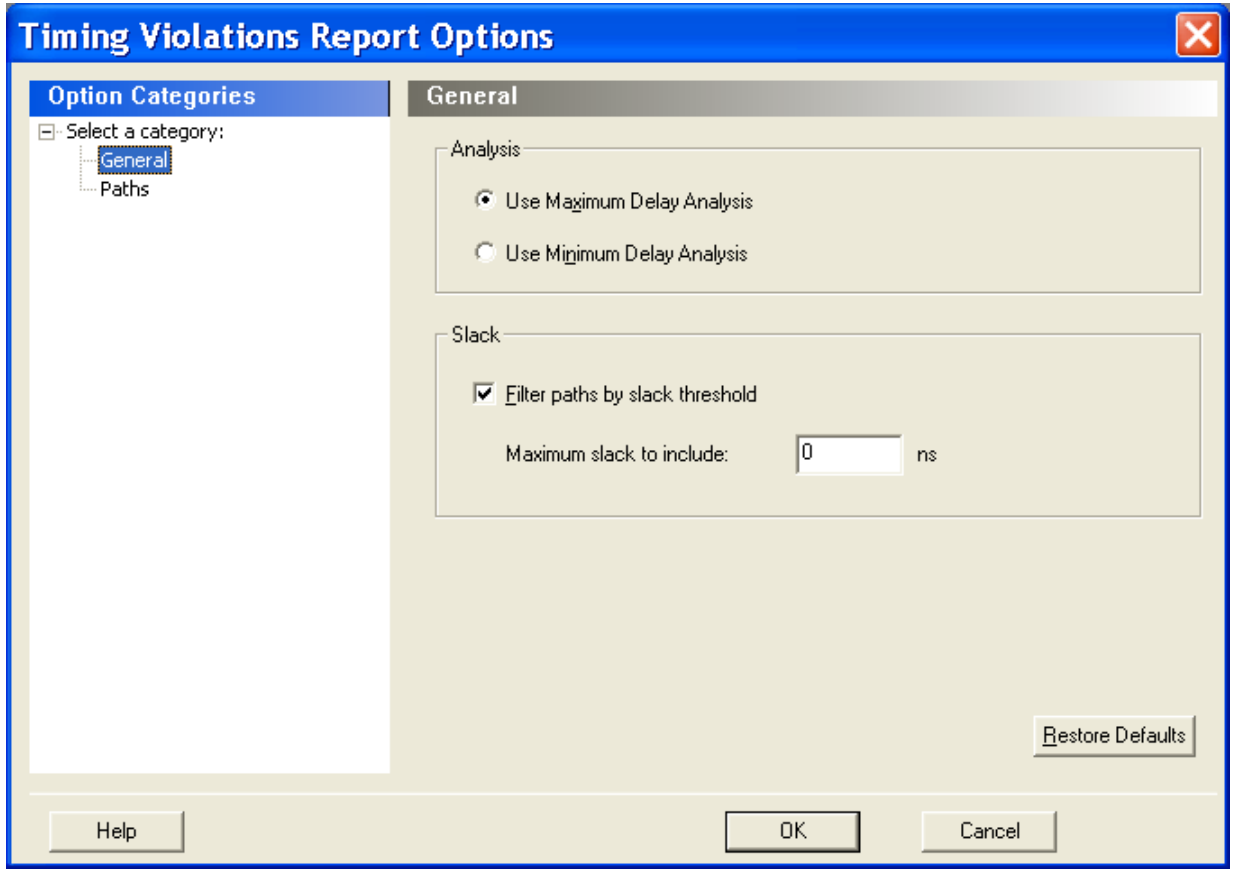
File Actions Help

```
Timing Report Max Delay Analysis

SmartTime Version 3.0
Actel Corporation - Actel Designer Software Release v7.3 SP1 (Version 7.3.1.9)
Copyright (c) 1989-2007
Date: Wed Mar 14 22:07:32 2007


Design: COUNT16
Family: ProASIC3
Die: A3P060
Package: 100 VQFP
Temperature: COM
Voltage: COM
Speed Grade: -2
Design State: Pre-Layout
Min Operating Condition: BEST
Max Operating Condition: WORST
Using Enhanced Min Delay Analysis


--------------------------------------------------
SUMMARY

Clock Domain:               CLK
Period (ns):                5.884
Frequency (MHz):            169.952
Required Period (ns):       5.714
Required Frequency (MHz):   175.009
External Setup (ns):        4.234
External Hold (ns):         0.010
Min Clock-To-Out (ns):      2.158
Max Clock-To-Out (ns):      6.320

Clock Domain:               CLKA
Period (ns):                N/A
Frequency (MHz):            N/A
Required Period (ns):       5.000
Required Frequency (MHz):   200.000
External Setup (ns):        N/A
External Hold (ns):         N/A
Min Clock-To-Out (ns):      N/A
```

SmartTime

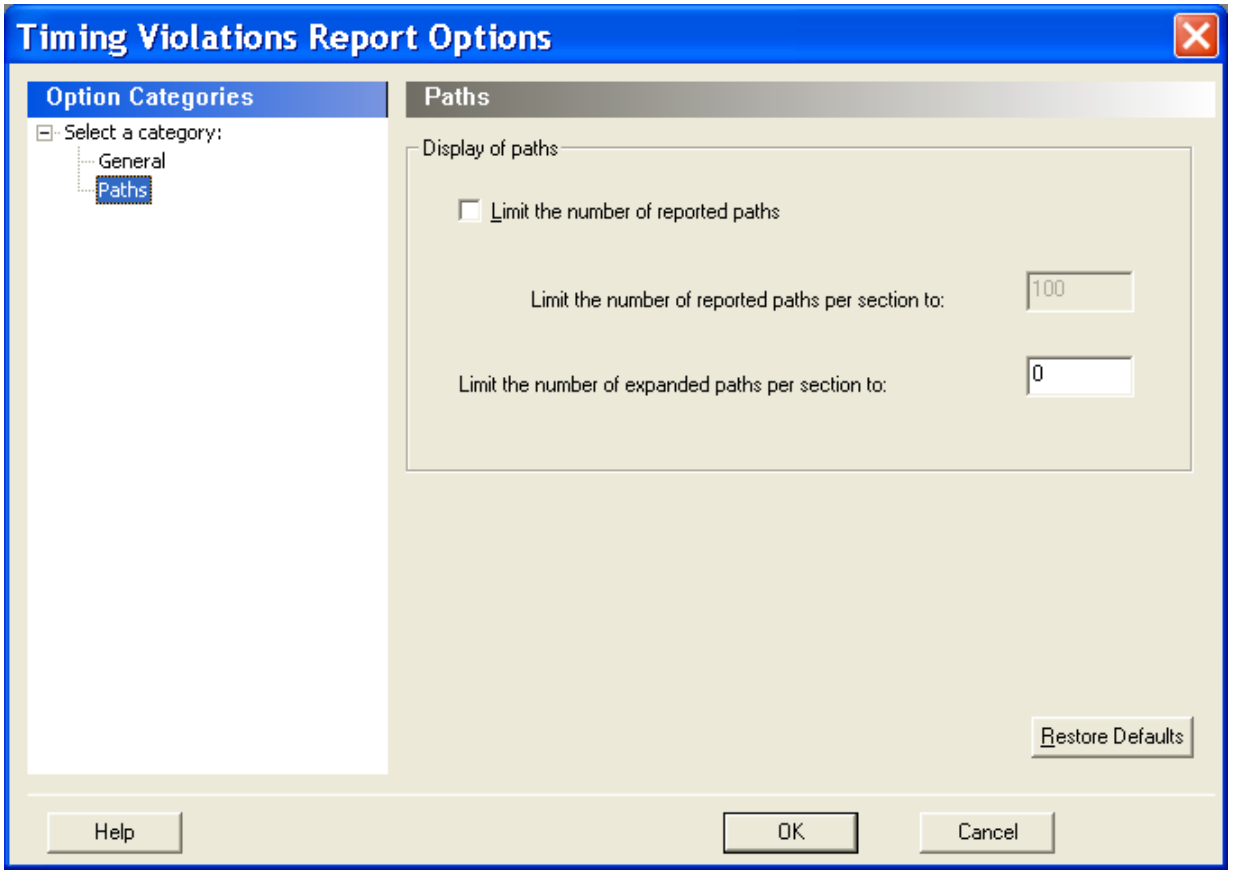
Timing Violations Report Options

- General Options
  - Specify Maximum or Minimum Delay Analysis
  - Enter Slack Threshold



SmartTime
Timing Violations Report Options

- Path Options
  - Limit Number of Reported Paths



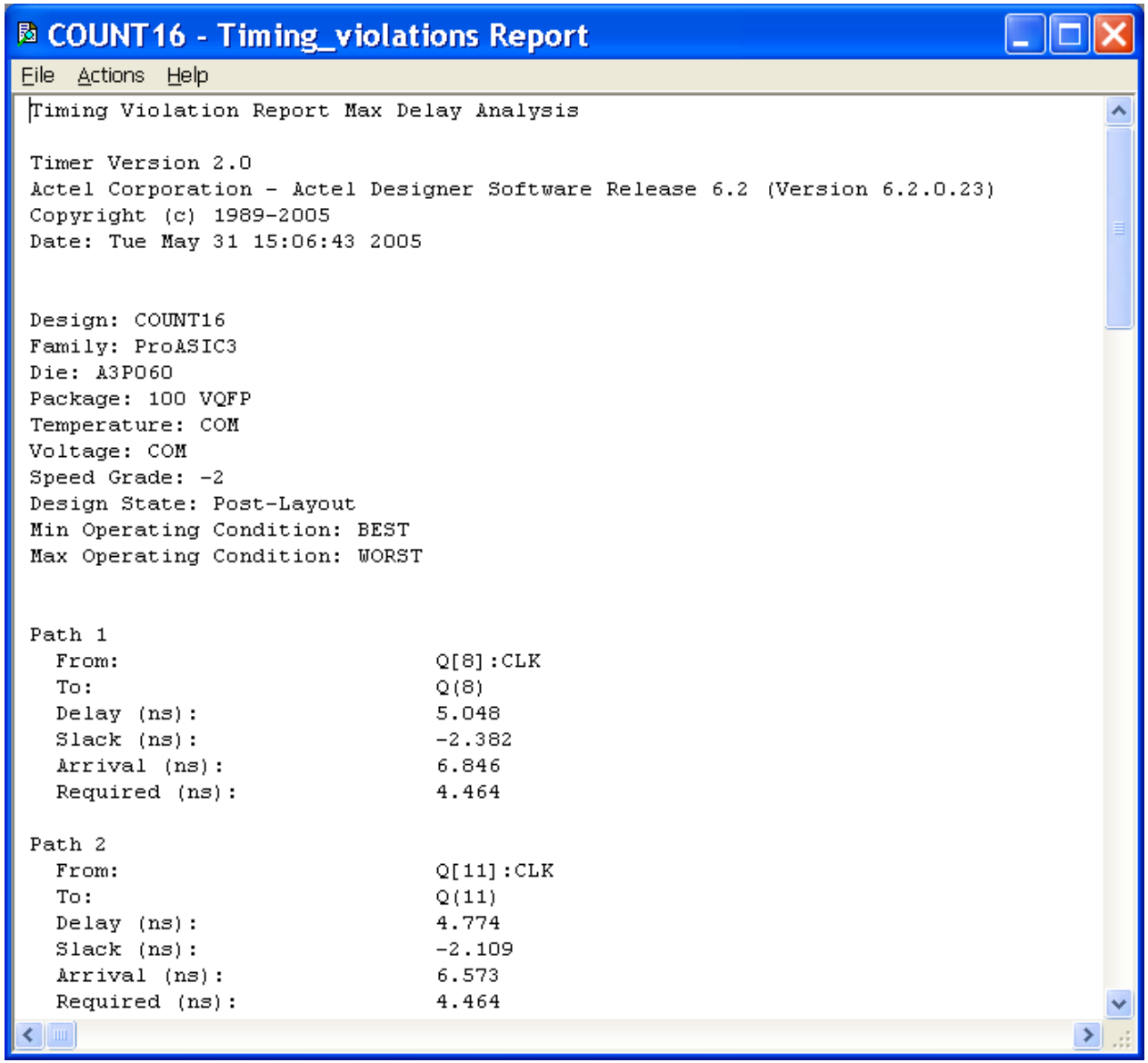
SmartTime

Timing Violation Report Structure

- Header
  - Design Information, Operating Conditions…
- Paths
  - List of Paths With Timing Violations

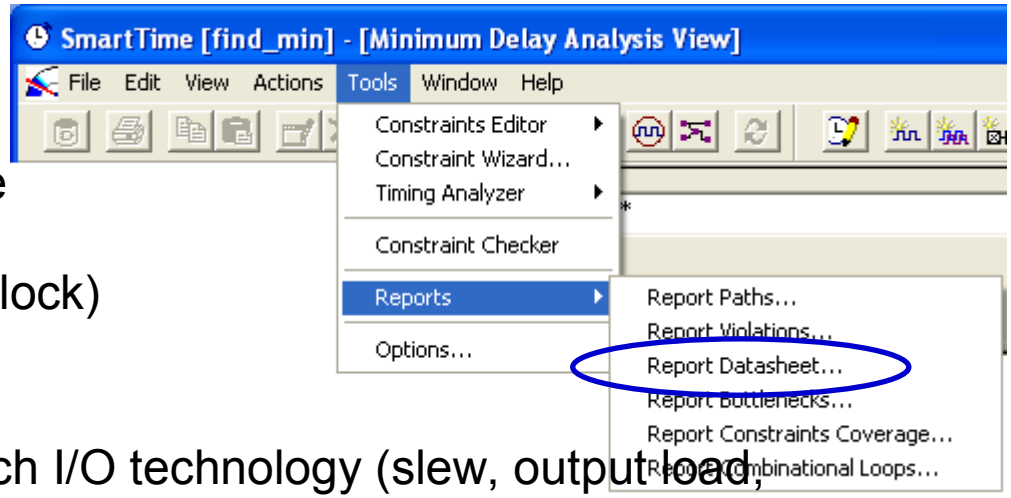
SmartTime

Timing Violation Report



SmartTime
Datasheet Report

- Content
  - Pin Description
    - Pin location on the package
    - Port name
    - Type (input, output, inout, clock)
    - I/O Technology used
  - DC Timing Characteristics
    - Electrical parameters of each I/O technology (slew, output load, voltages,)
  - AC Timing Characteristics
    - External Timing Requirements
- New analysis view in the SmartTime Analyzer
- Report Format
  - Text format or CSV (Comma Separated Value)



Datasheet Report
AC Timing Characteristics

- Maximum Clock Frequency for all External Clocks
- External Setup/Recovery/Hold/Removal for Each Input Port with Respect to External Clocks
- Clock-to-out for Output Ports
- Input-to-Output for Combinational Paths

Datasheet Report

Pin Description

Name	Location	Type	Techno
a[0]	J14	Input	LVCMOS15
b[0]	H15	Input	LVCMOS25
reset	K1	Input	SSTL2I
ci	B8	Input	LVTTL (1)
clk	R7	Clock	GTLP25
y[0]	C9	Output	LVTTL (2)
co	B7	Output	LVTTL (3)
en	L2	Input	PCI

DC Electrical Characteristics

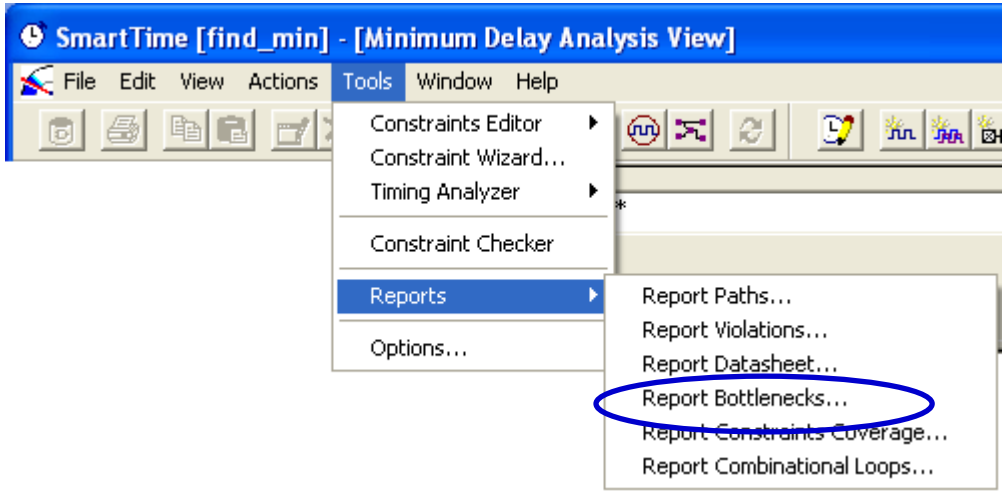
Name	Vcci (V)	Resistor Pull	Input Delay	Hot Swappable	Vccr (V)	Output Drive (mA)	Slew	Output Load
GTLP25	2.5		no	yes	1			
LVCMOS15	1.5	Up	no	yes				
LVCMOS25	2.5	None	no	yes				
LVTTL (1)	3.3	None	no	yes				
LVTTL (2)	3.3	None		yes		24	Low	35
LVTTL (3)	3.3	None		yes		12	High	40
PCI	3.3		no	no				
SSTL2I	2.5		no	yes	1.25			

Datasheet Report (cont.)

AC Electrical Characteristics

Description				Min	Max	Unit
Clock frequency	clk				189.143	MHz
Clock period	clk			5.287		ns
Setup time	a[0]	before	clk (rise)	-0.946		ns
Setup time	b[0]	before	clk (rise)	-2.642		ns
Setup time	ci	before	clk (rise)	-3.100		ns
Setup time	reset	before	clk (rise)	-0.096		ns
Hold time	a[0]	after	clk (rise)	-1.278		ns
Hold time	b[0]	after	clk (rise)	-2.062		ns
Hold time	ci	after	clk (rise)	-2.241		ns
Hold time	reset	after	clk (rise)	-0.429		ns
Recovery time	reset	before	clk (rise)	-0.266		ns
Removal time	reset	after	clk (rise)	-0.429		ns
Propagation delay	clk (rise)	to	co	3.545	7.420	ns
Propagation delay	clk (rise)	to	y[0]	6.720	14.309	ns

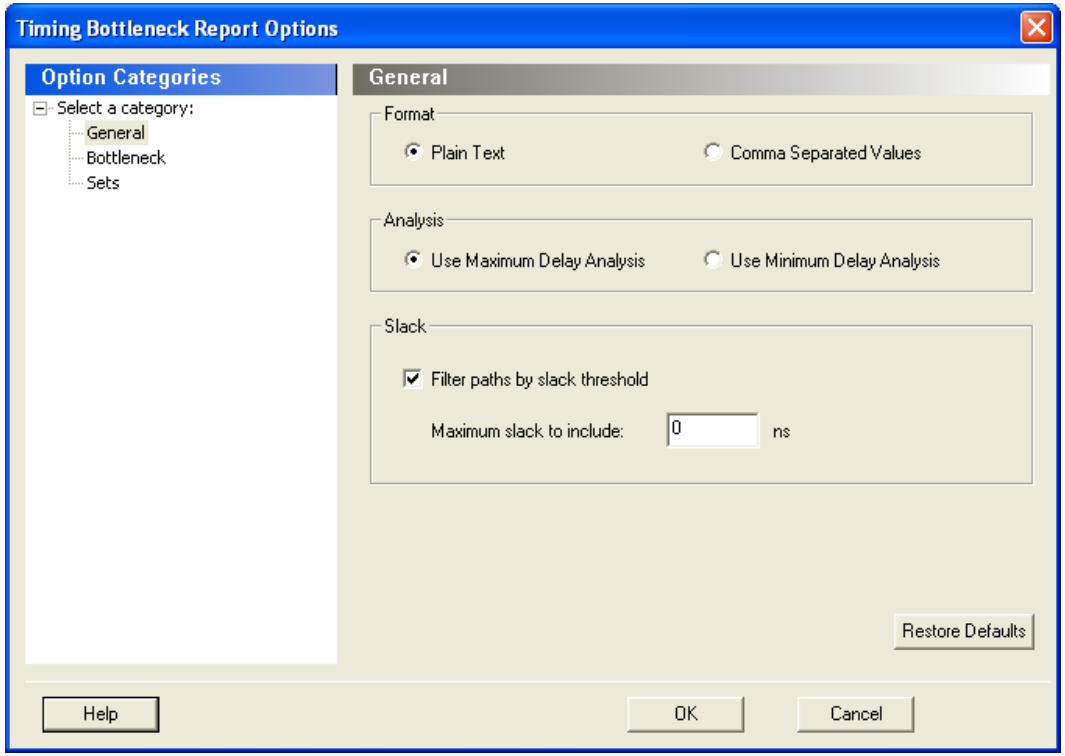
SmartTime
Bottleneck Report



- Bottleneck Analysis
  - A bottleneck is a point in the design that contributes to multiple failing paths
  - SmartTime finds and lists all bottlenecks according to severity
- SmartTime “Bottleneck Analysis Report” Shows
  - Severity: “Path_Costs”
    - Lists instances causing the greatest amount of delay
      - Reports total max delay caused by the instance
  - # of Occurrences: “Path_Count”
    - Lists instances causing greatest number of path violations
      - Reports number of paths affected by the instance delay

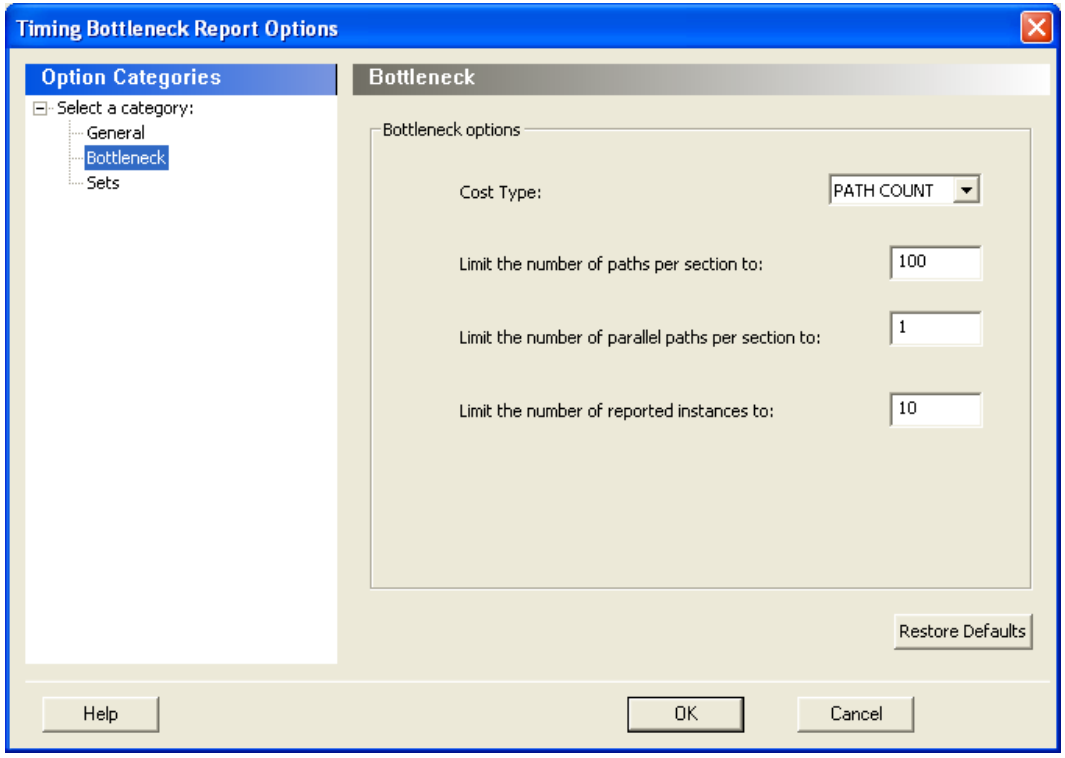
Bottleneck Analysis Report
General Options

- Format
  - Text or CSV
- Maximum or Minimum Delays
  - Default uses Max Delay
- Filter by Slack Value
  - Violations below the specified value are not reported
  - Default = 0 ns



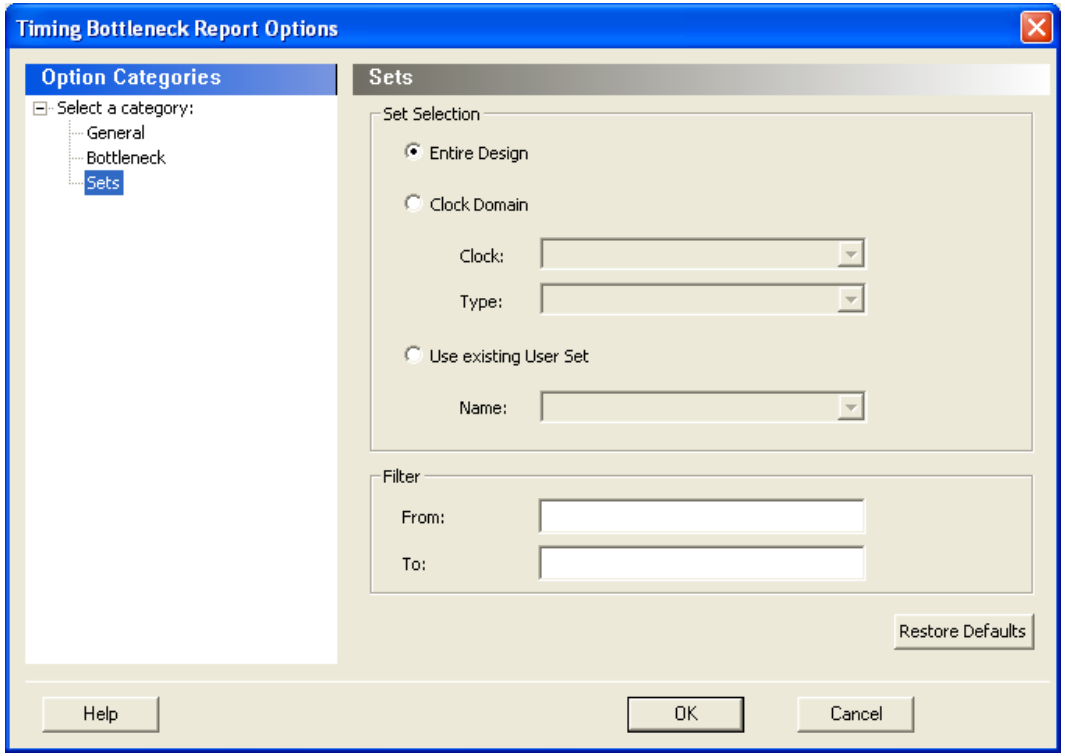
Bottleneck Analysis Report
Bottleneck Options

- Cost Type
  - Path Count
  - Combined Delay
- Specify
  - Number of Paths to Analyze
  - Number of Parallel Paths per Section
  - Number of Instances



Bottleneck Analysis Report
Set Options

- Define Sets to be Analyzed:
  - Entire Design
  - Specific Clock Domain
  - User Defined Set
- Report Instances on Violating Paths
  - From starting from location
  - To a destination location



Bottleneck Analysis
Path Count Report

- Path Count Report Shows:
  - Number of paths where an instance is causing delay
  - The corresponding instance name

Bottleneck Analysis

Instance Name	Bottleneck Cost
COUNTING_count_3_G_1_3:Y	31
COUNTING_count_3_I_98:Y	31
COUNTING_count_3_I_95:Y	24
COUNTING_count_3_G_1:Y	24
COUNTING_count_3_I_82:Y	20
COUNTING_count_3_I_93:Y	16
COUNTING_count_3_I_97:Y	16
COUNTING_count_3_G_1_4:Y	14
COUNTING_count_3_G_1_1_0:Y	12
COUNTING_count_3_I_59:Y	12

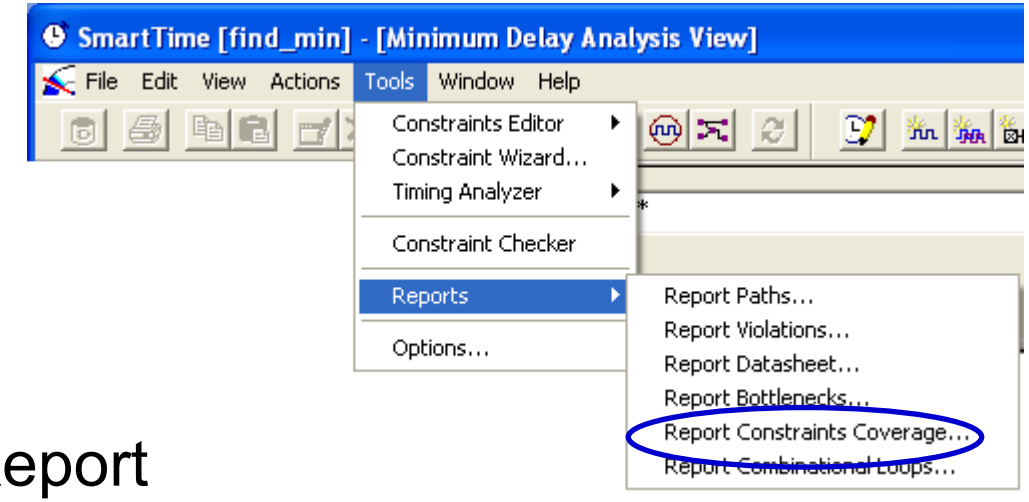
Bottleneck Analysis
Path Cost Report

- Lists total combined Max Delay caused by an instance

Bottleneck Analysis

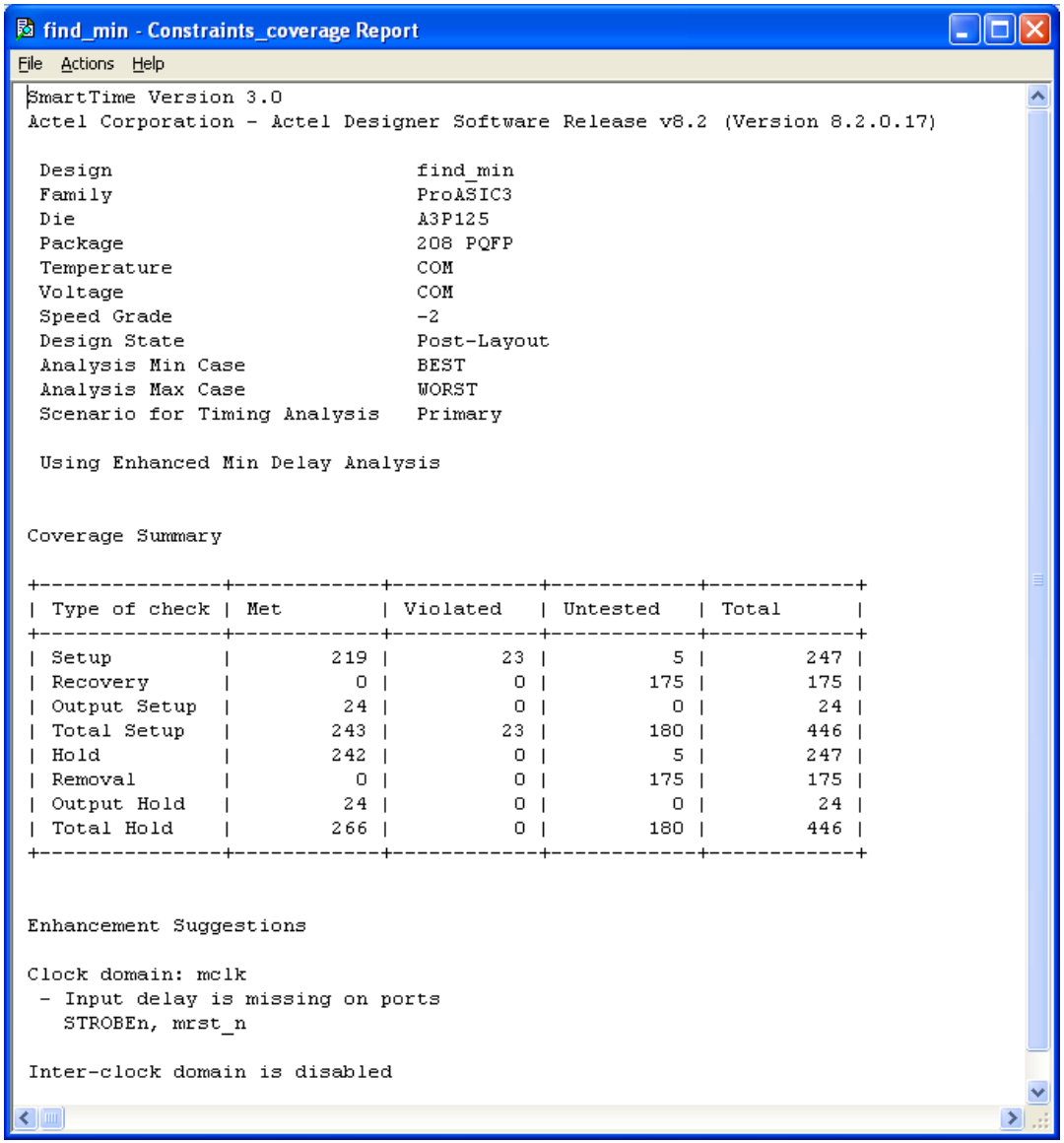
Instance Name	Bottleneck Cost
COUNTING_count_3_G_1_3:Y	-94.730
COUNTING_count_3_I_98:Y	-90.718
COUNTING_count_3_I_95:Y	-76.488
COUNTING_count_3_G_1:Y	-66.400
COUNTING_count_3_I_82:Y	-61.206
COUNTING_count_3_I_97:Y	-53.968
COUNTING_count_3_I_93:Y	-47.400
COUNTING_count_3_G_1_3_0:Y	-41.203
COUNTING_count_3_I_99:Y	-38.910
Q[3]:Q	-38.436

SmartTime
Constraint Coverage Report

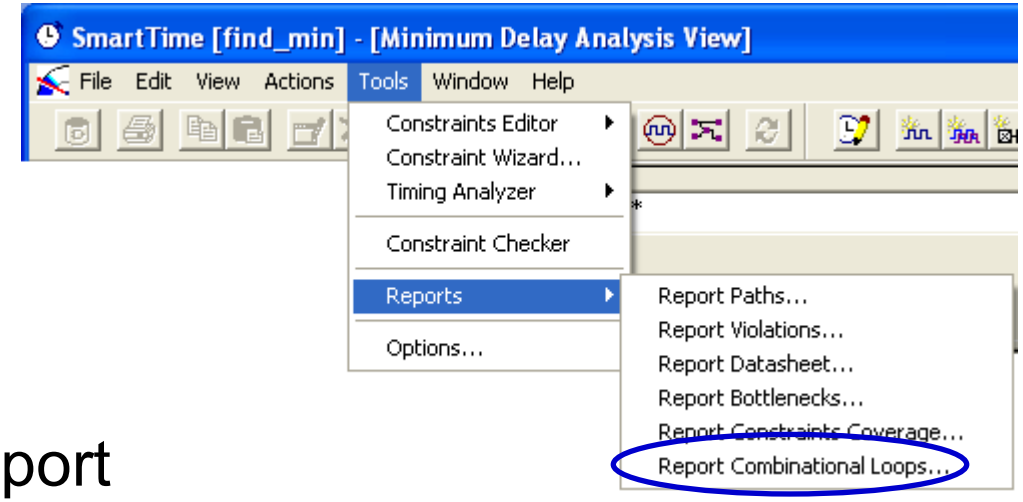


- Constraint Coverage Report
  - Reports statistics of constraints
    - Met
    - Not Met
    - Not Tested

Constraint Coverage Report

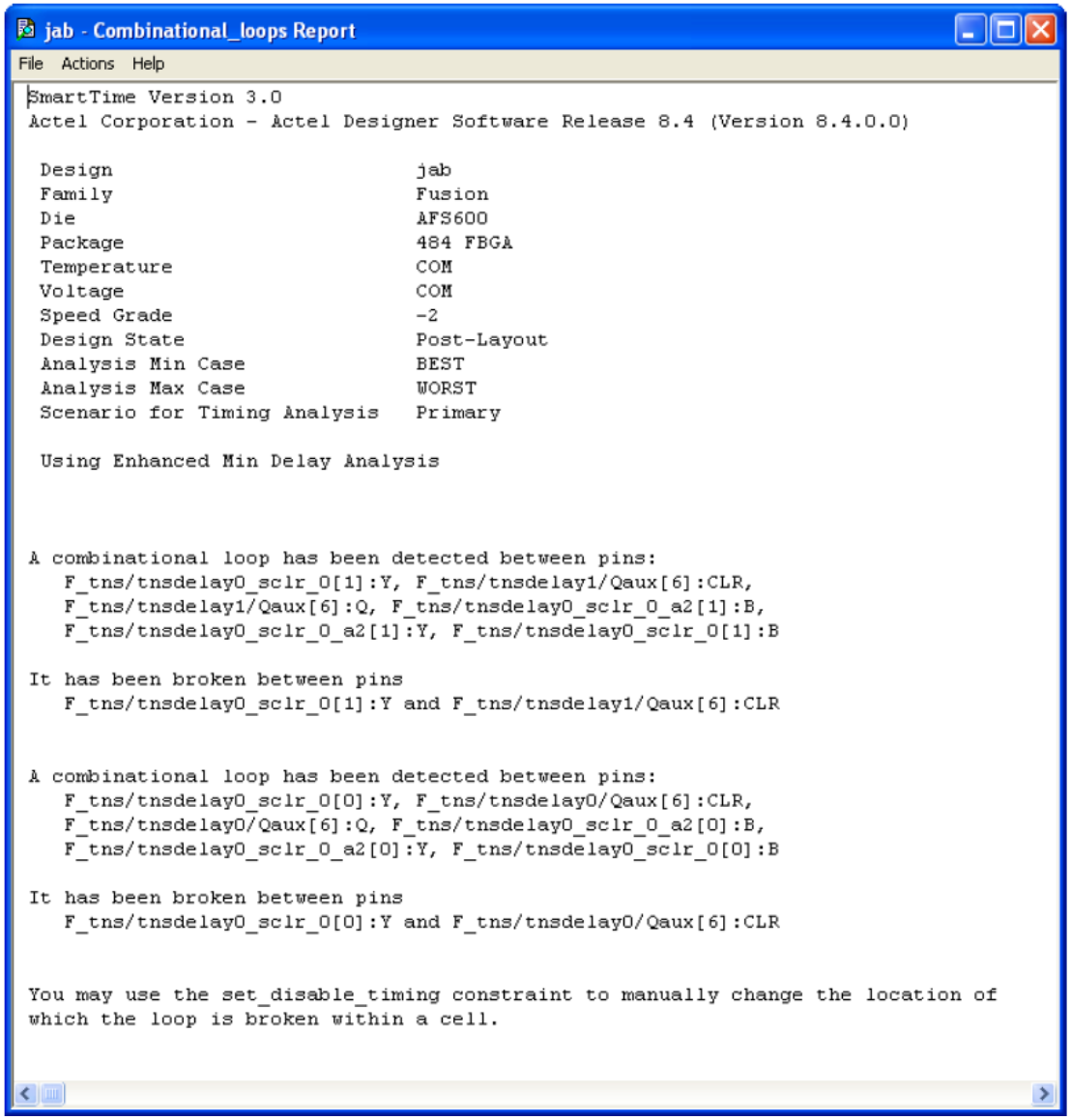


SmartTime
Combinatorial Loops Report

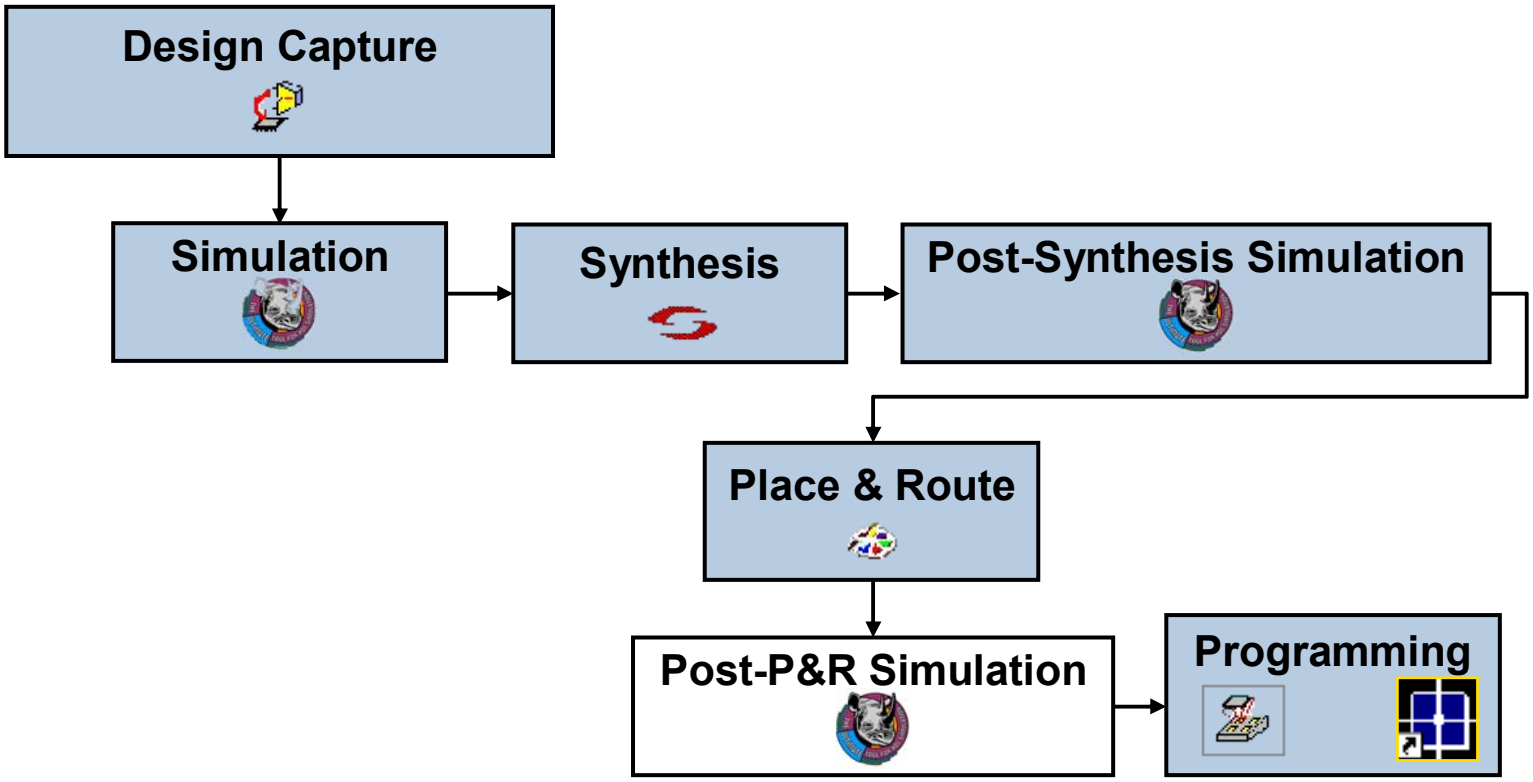


- Combinatorial Loop Report
  - **Displays all loops found during initialization**
  - **Reports pins associated with the loop(s), location where the loop is broken for timing analysis**

Combinatorial Loop Report

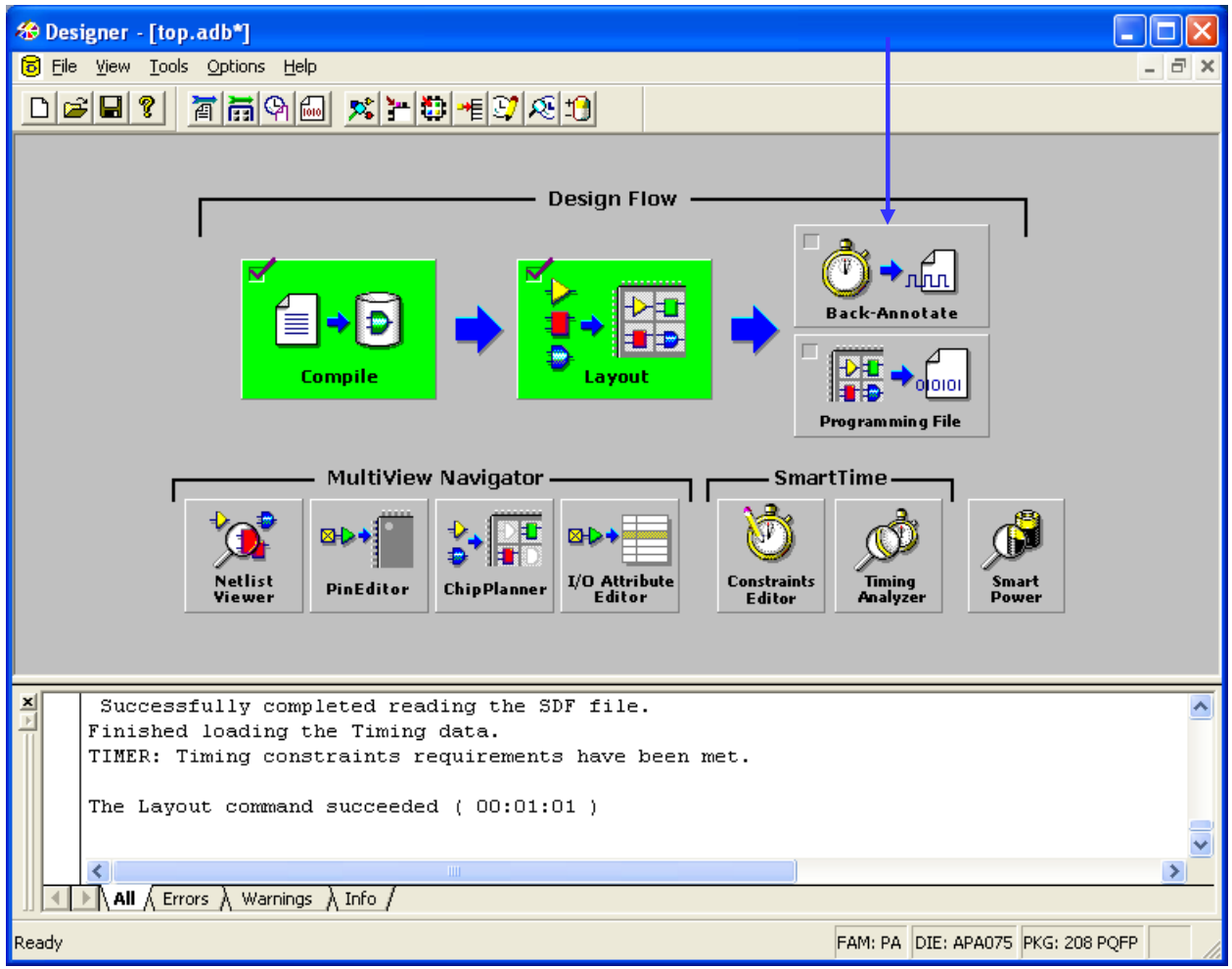


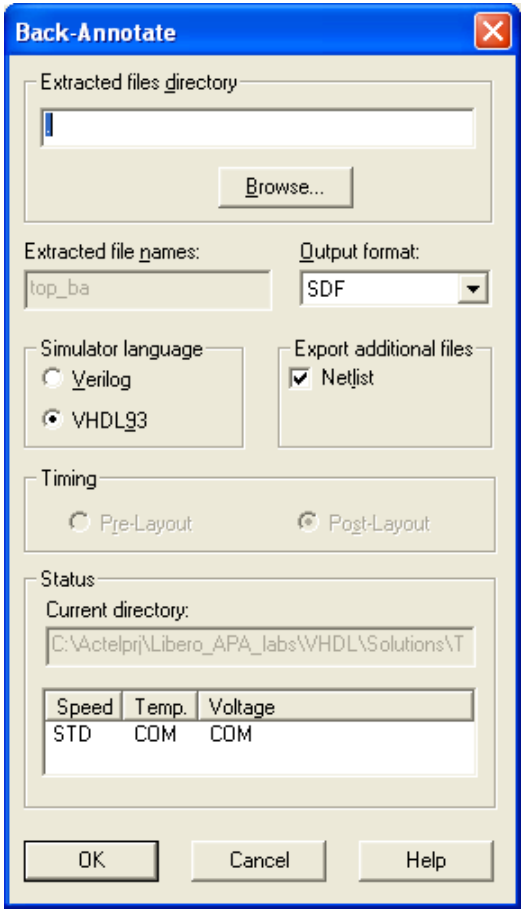
Post-Layout Simulation



Exporting SDF File

Extract Timing delays
for post-layout simulation





Post-Layout Simulation

- Steps
  - Route design
  - Export .sdf file (Back-annotate)
  - Run Post-layout Simulation
- SDF File Contains Delays for Min, Typ and Max

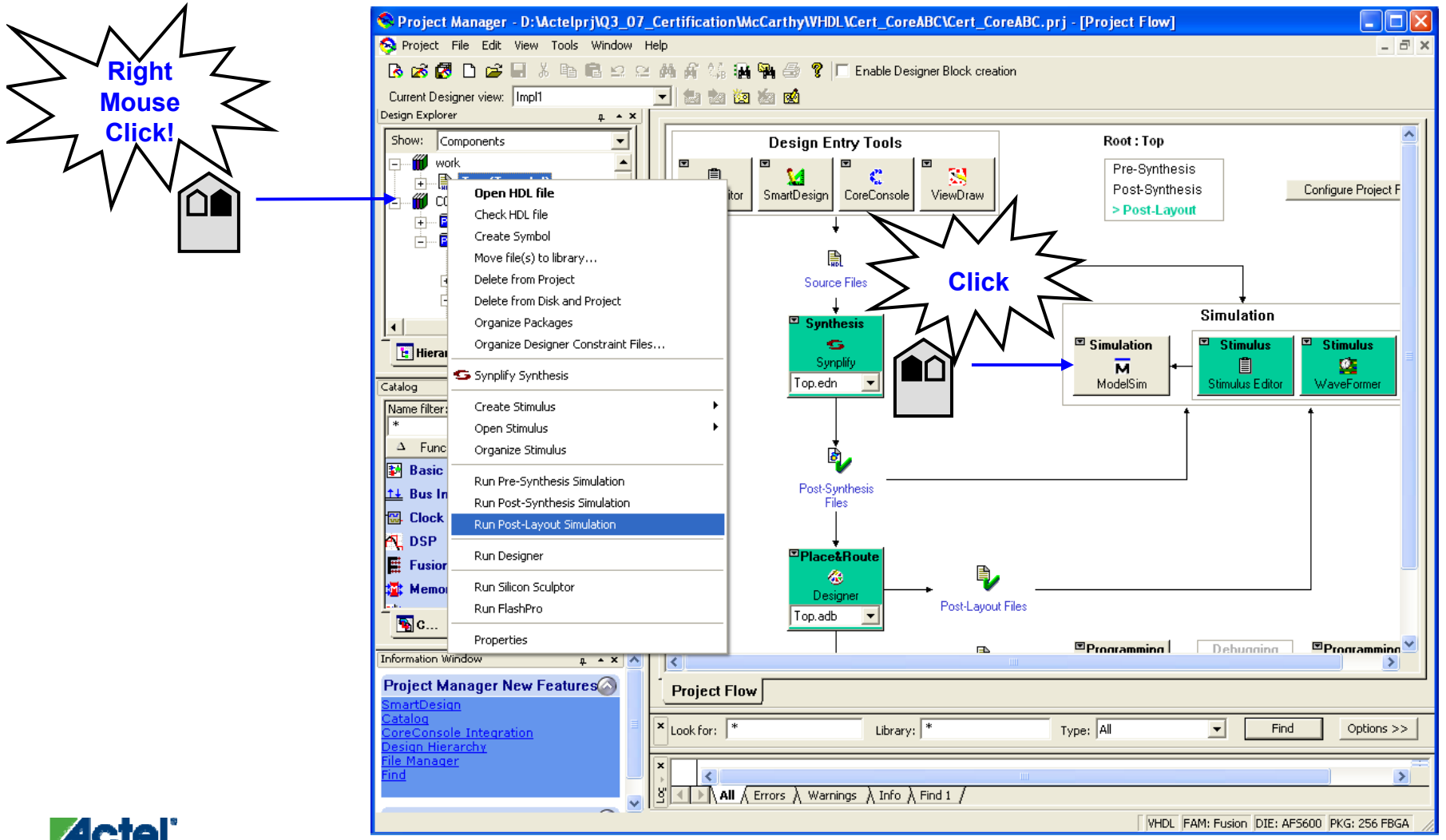
```
(DELAYFILE
 (SDFVERSION "2.1")
 (DESIGN "counter")
  (VOLTAGE 2.70:2.50:2.30)
 (PROCESS "WORST")
 (TEMPERATURE 0:25:70)
 (TIMESCALE 100ps)

 (CELL
 (CELLTYPE "OUTBUF")
 (INSTANCE COUNT_pad_12)
 (DELAY
  (ABSOLUTE
     (PORT D (1.65:2.55:3.52) (2.21:3.40:4.62))
     (IOPATH D PAD (19.19:28.90:39.88) (17.49:26.35:38.85))
  )
 )
 )
```

rising min:typ:max

falling min:typ:max

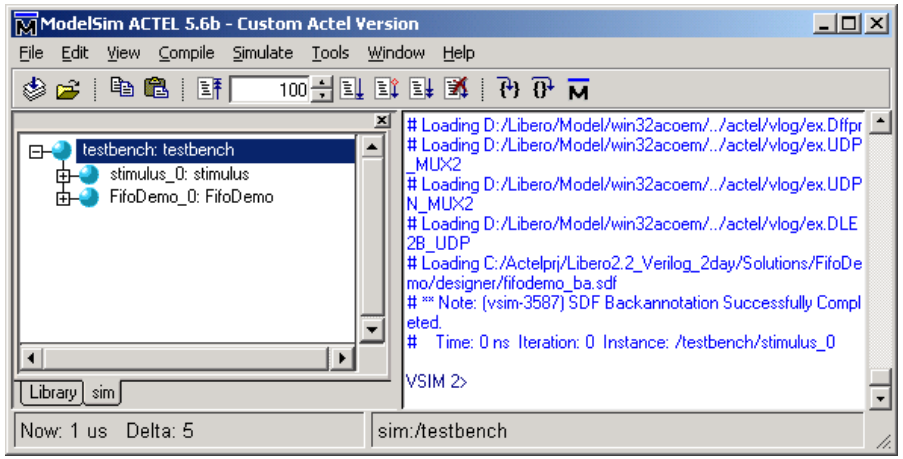
Post-Layout Simulation

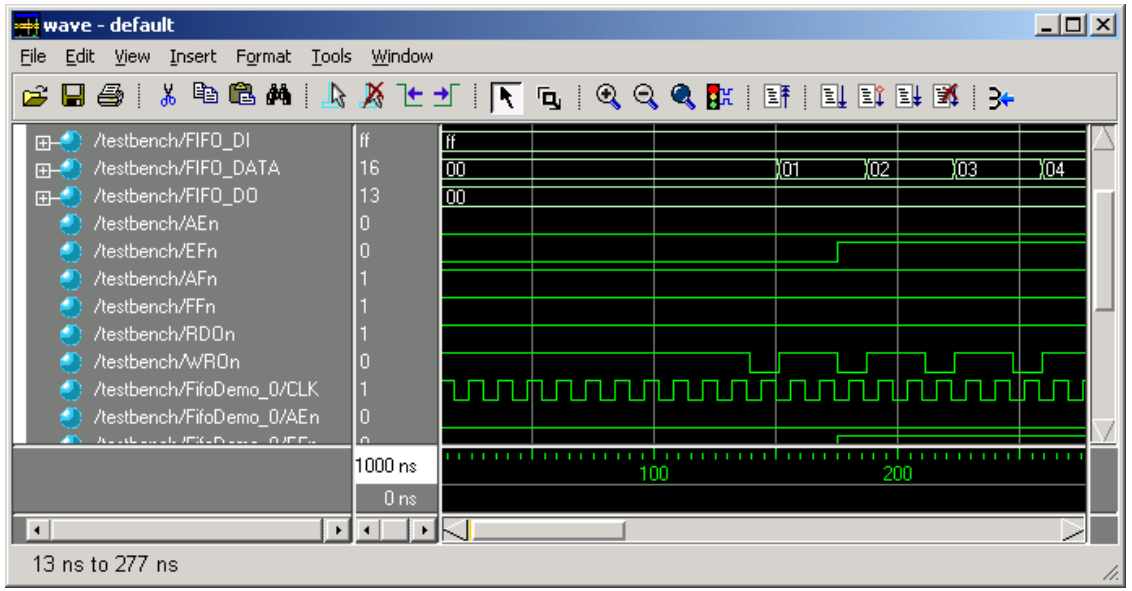
- Click “Simulation” in Project Flow Window or…



Post-Layout Simulation

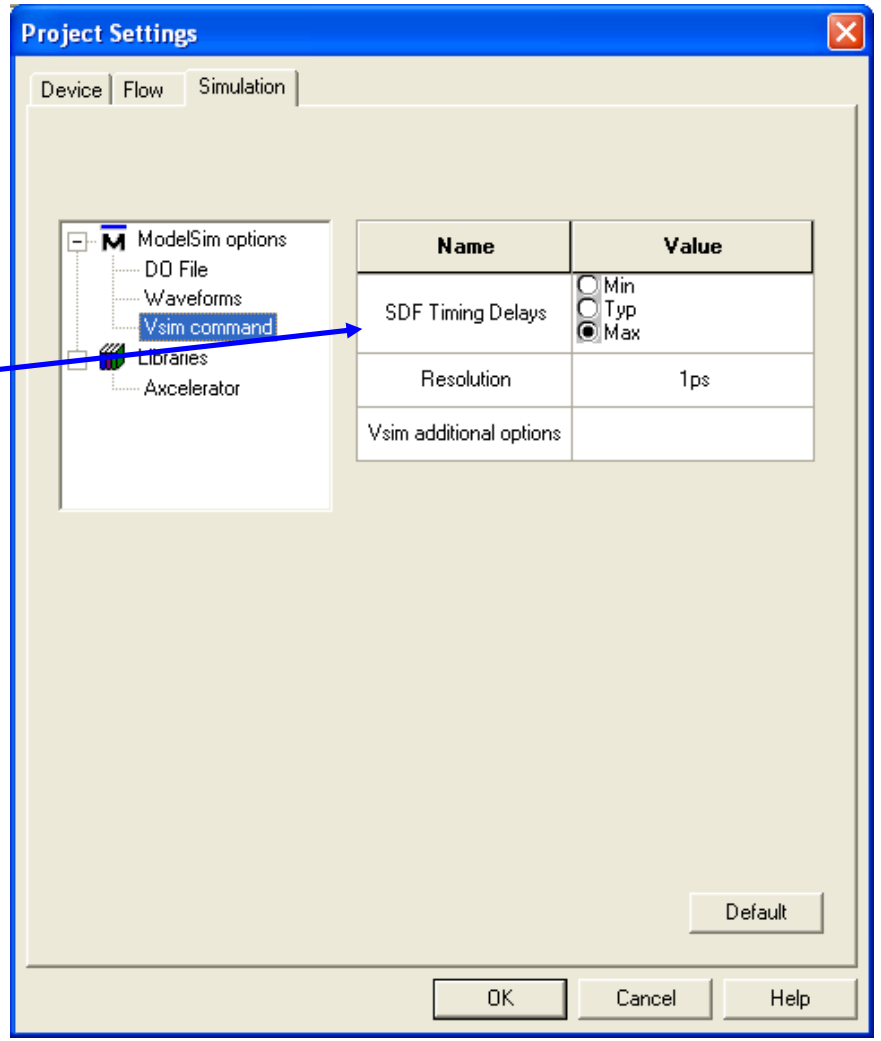
- Structural Netlist and .sdf File Used for Simulation
  - Simulator runs for 1 uS as Default
  - Max Operating Conditions Default

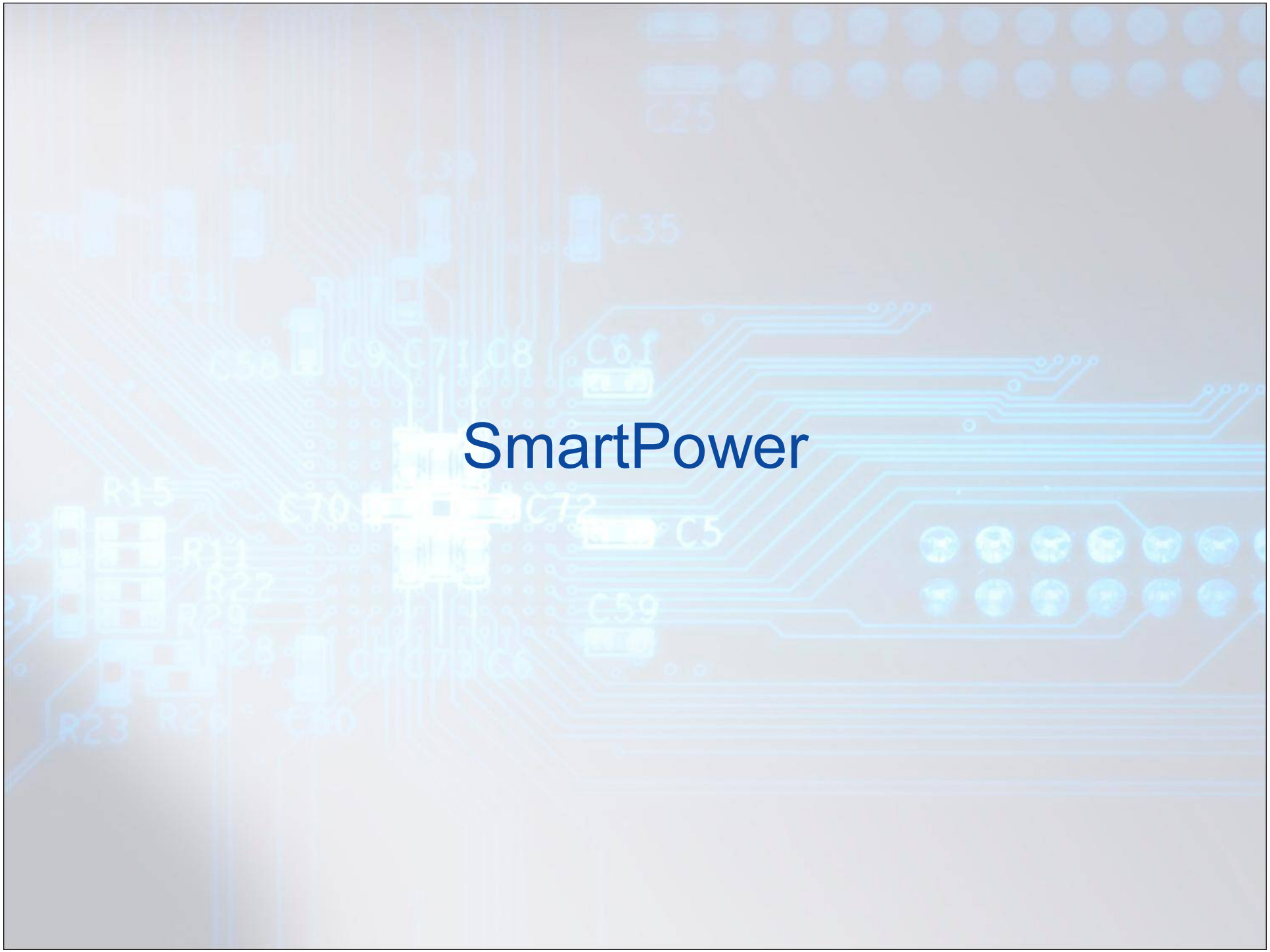




Post-Layout Simulation
Selecting Operating Conditions

- Post-layout Operating Conditions Can Be Specified within Libero
  - Tools > Options from Libero Main Window
  - Select Simulation Tab in Options Window
  - Choose Min/Typ/Max

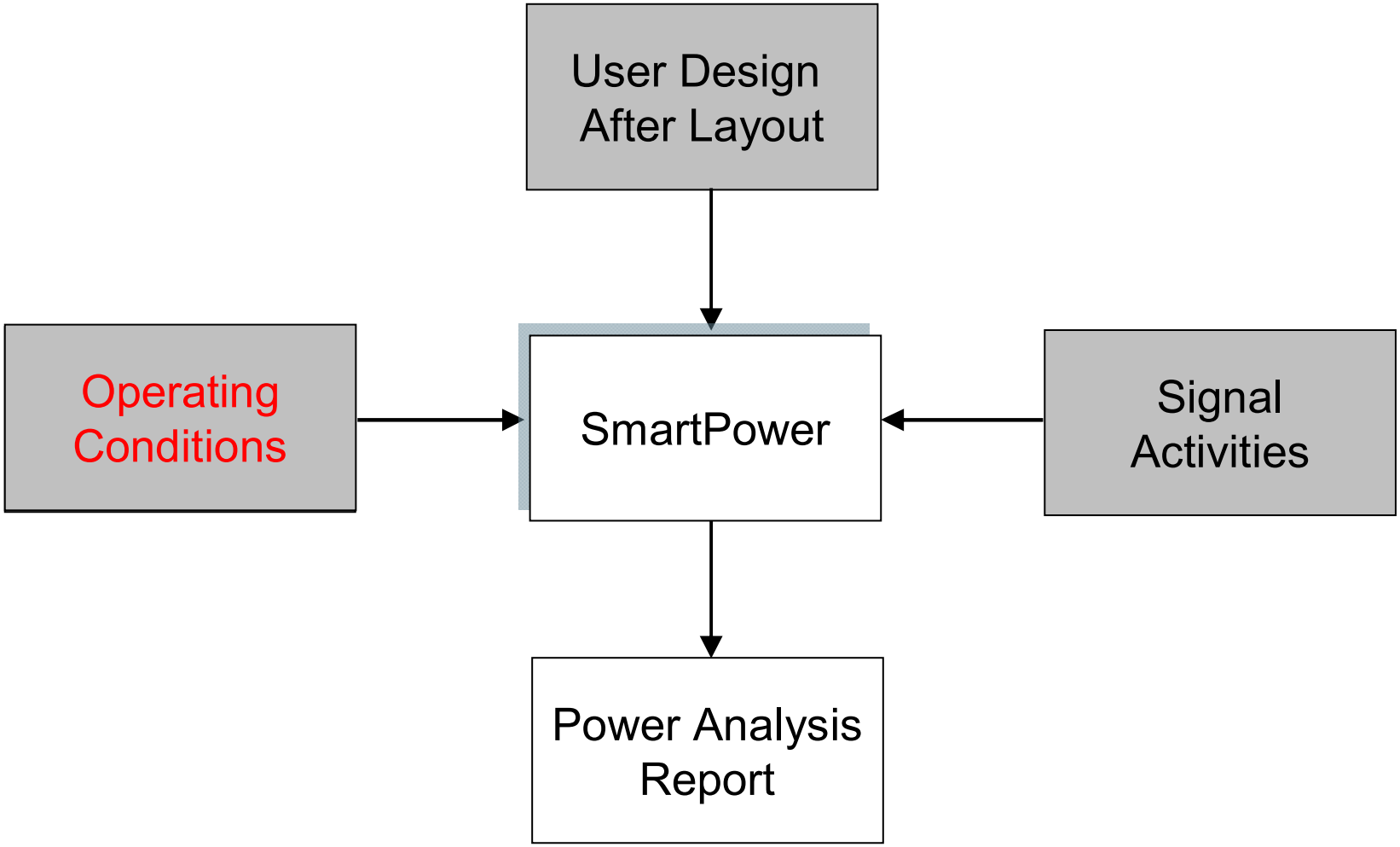



SmartPower

SmartPower Overview

- SmartPower Supports Axcelerator, RTAX-S, Fusion, IGLOO, ProASIC3, ProASIC^{PLUS} and ProASIC
- SmartPower Features:
  - Power Estimate for Individual Components in Design
  - Hierarchical Reports of Dynamic Power Consumption
  - Linked to SmartTime
    - Uses Clock Constraints
  - Support for Simulation Files:
    - VCD (Value-Change Dump)
    - SAIF (Synopsys Switching Activity Interchange Format)

SmartPower Overview



SmartPower Operating Conditions (1)

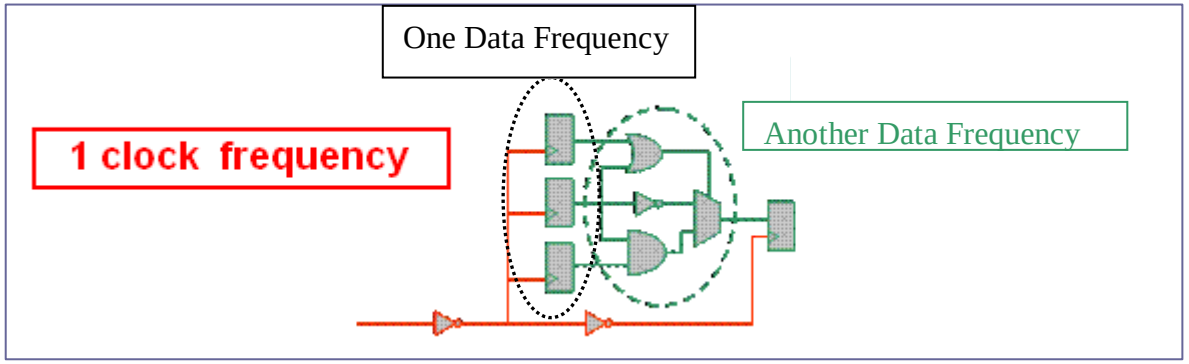
- Operating Conditions are Temperature, Voltage, Process
- Impact of Temperature:
  - High on Static Power,
  - Limited on Dynamic Power
- Impact of Process:
  - High on Static Power,
  - Limited on Dynamic Power
- Impact of Voltage:
  - High on Static Power,
  - High on Dynamic Power
- Impact of Radiation:
  - For RTAX-S, the TID reports show a very small rise in ICC at 100Krad. This rise in ICC is so small that it is in the range of error for experimental measurement, and therefore is insignificant
  - For RT-A3P, the TID reports show that this is also true for TID < 40Krad

SmartPower Operating Conditions (2)

- Reminder about Best and Worst Corners Power
  - Power Corner definition differs from Timing
    - Best Power Corner:
      - Low Temperature, Low Voltage
    - Worst Power Corner:
      - High Temperature, High Voltage
  - For ProAsic3 COM for example
    - Best Power Corner: Temp= 0c, VCCA=1.425 V, VCCI = 3.0V
    - Typical Power Corner: Temp=25c, VCCA=1.5 V, VCCI = 3.3V
    - Worst Power Corner: Temp=70c, VCCA=1.575 V, VCCI = 3.6V

SmartPower Signal Activities

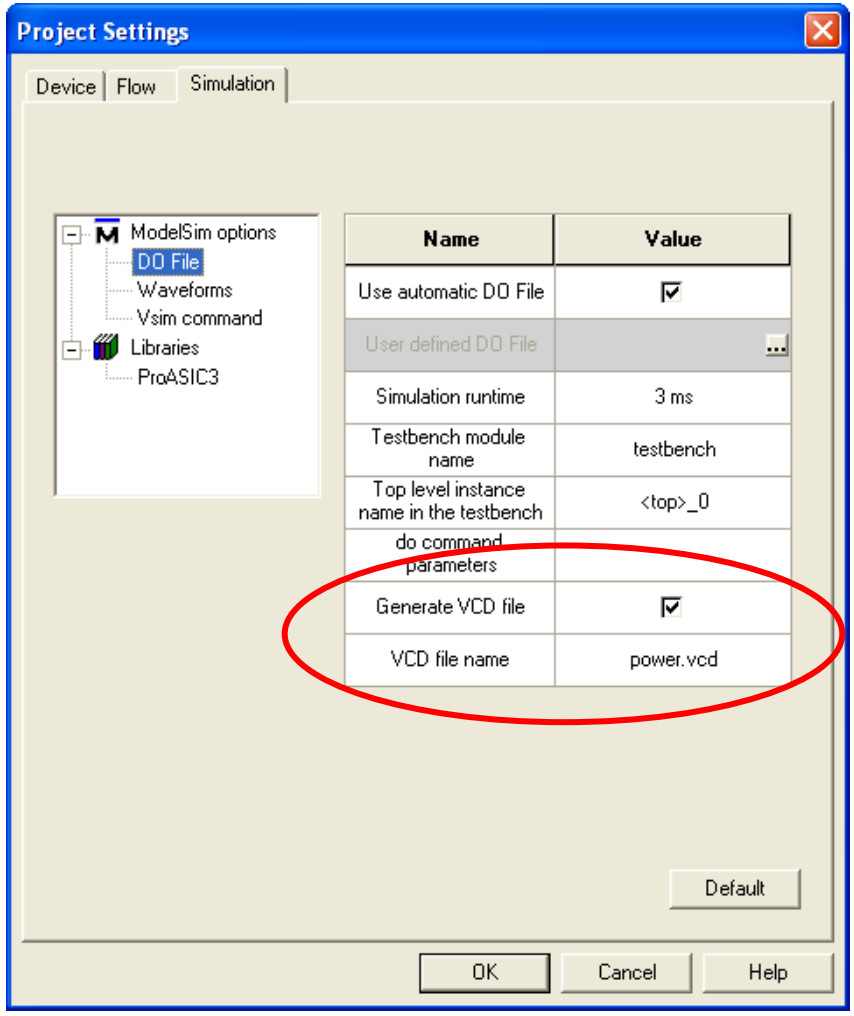
SmartPower Switching Activity Estimation is Based on the Concept of Clock Domains



- From Simple Model
  - Automatic Partitioning in Clock Domains by SmartPower
  - Only a few frequencies to set per Clock Domain
- To Accurate Model
  - Manual Partitioning
  - Pin by Pin annotation or Import Simulation file

Simulation Based Flow (1)

- Open Libero Simulation Options
- Check the “Generate VCD file” Checkbox Under “Do File”
- VCD File is Created Automatically During Simulation
  - Actel recommends performing post-layout simulation using the back-annotated netlist when generating the VCD files

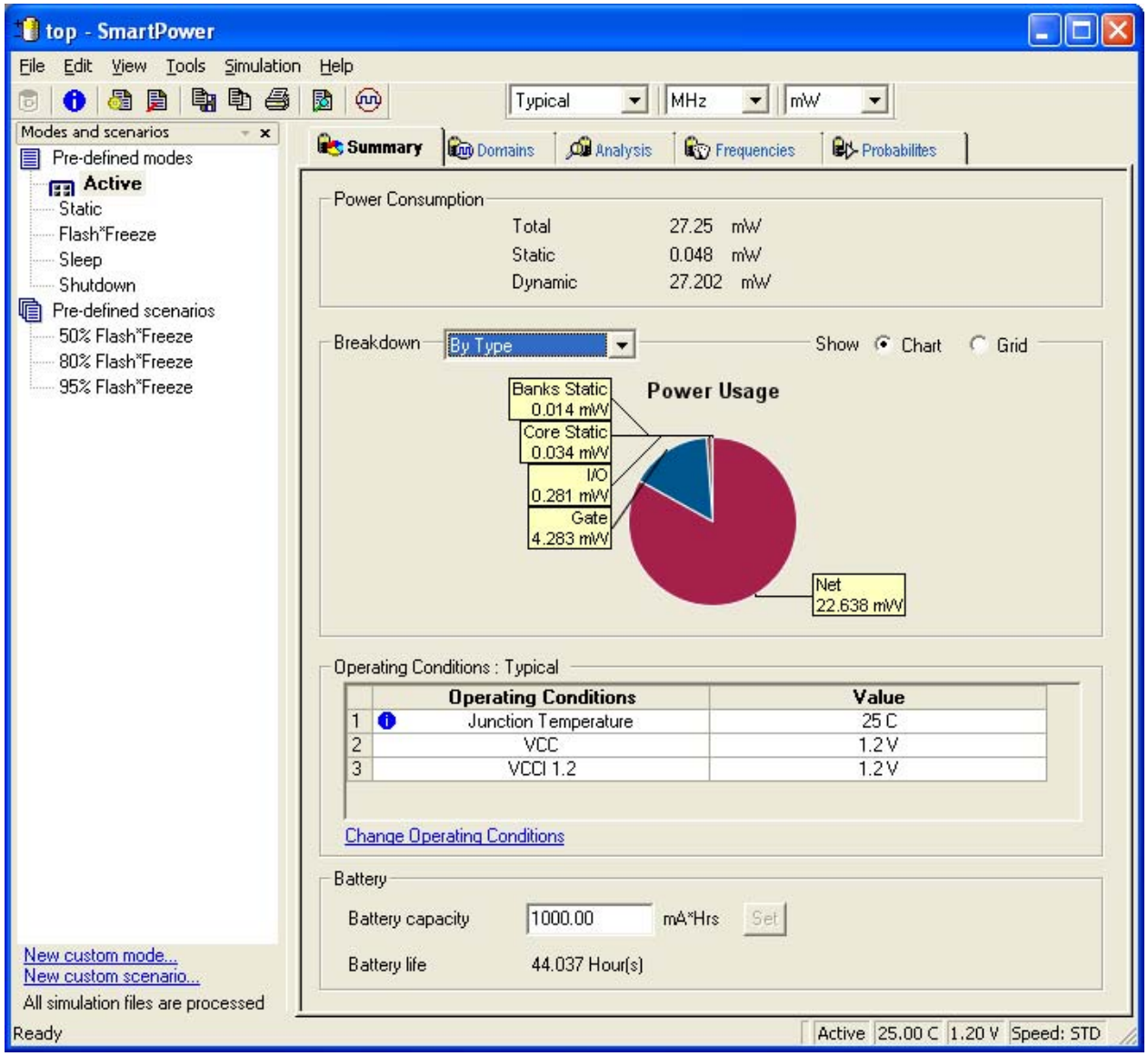


Simulation Based Flow (2)

- Simulation based estimation, User may still affect results
  - Quality of Simulation Trace
  - Post-Layout simulation versus RTL level simulation
  - Simulation for design coverage versus simulation to emulate true patterns and timings relationships, etc
  - Impact of glitches
- How do you know if your simulation is OK for power analysis?

SmartPower GUI

SmartPower GUI



SmartPower Summary Tab

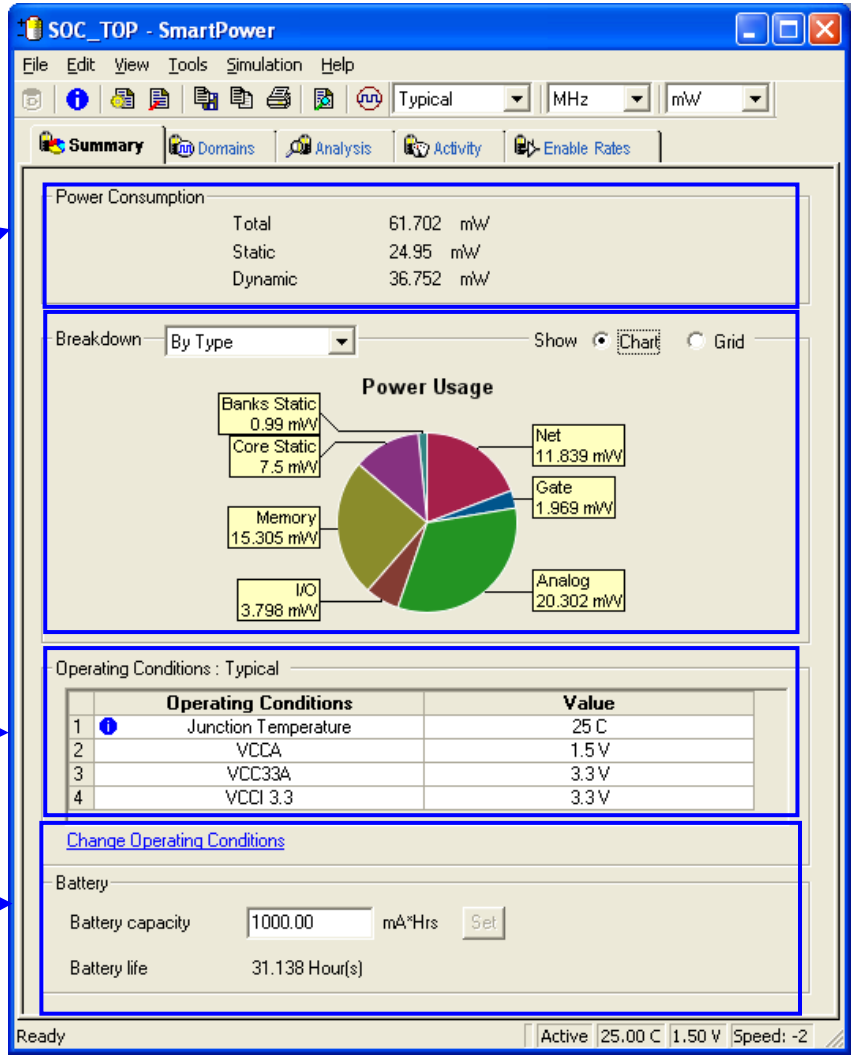
- Summary Tab Displays:
  - Static and Dynamic Power
  - Junction Temperature
  - Battery Life

Power Consumption section displays total static and dynamic power

Breakdown section displays the power breakdown by type of component or by voltage rail or by clock domain
Data displayed as pie chart or in grid

Temperature section displays the thermal resistance θ_{JA} and the impact of the power consumption on the junction temperature for a given cooling scenario

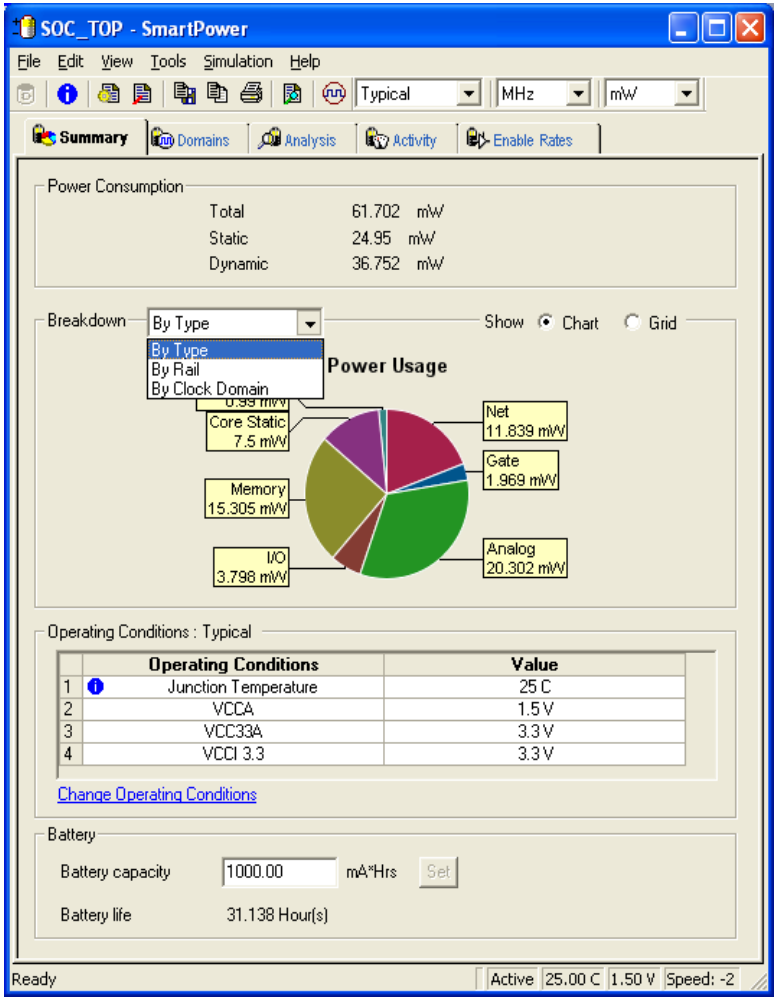
Battery Life Estimate for given battery capacity



SmartPower Summary Tab

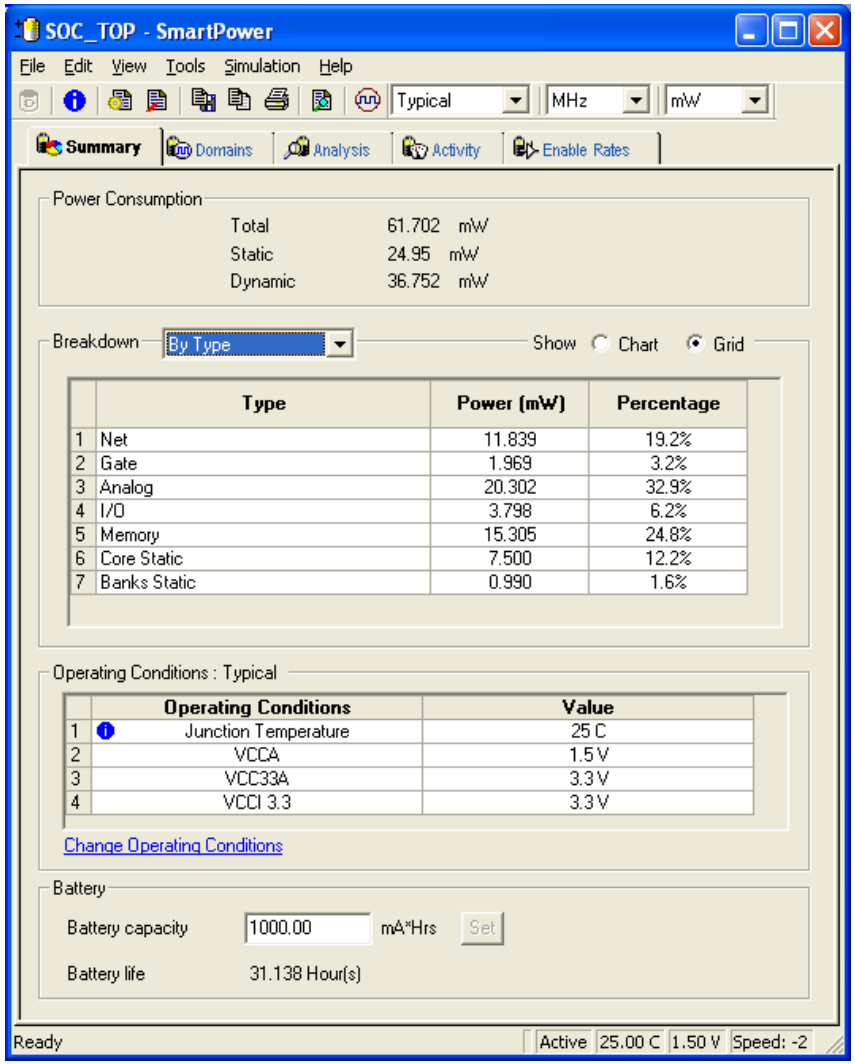
Power Breakdown

- Display Power Consumption by Type, Rail or Clock Domain
  - Pie chart or grid display
- Type:
  - Displays a Breakdown of the Design's Power Usage by:
    - Net
    - Gate
    - I/O
    - Memory
    - Clock
    - Core Static
    - Banks Static
- Rail:
  - Displays a Breakdown of the Power Usage by Voltage rail
    - Power, Voltage, and Current Drawn are Displayed for each rail
- Clock Domain



SmartPower Summary Tab

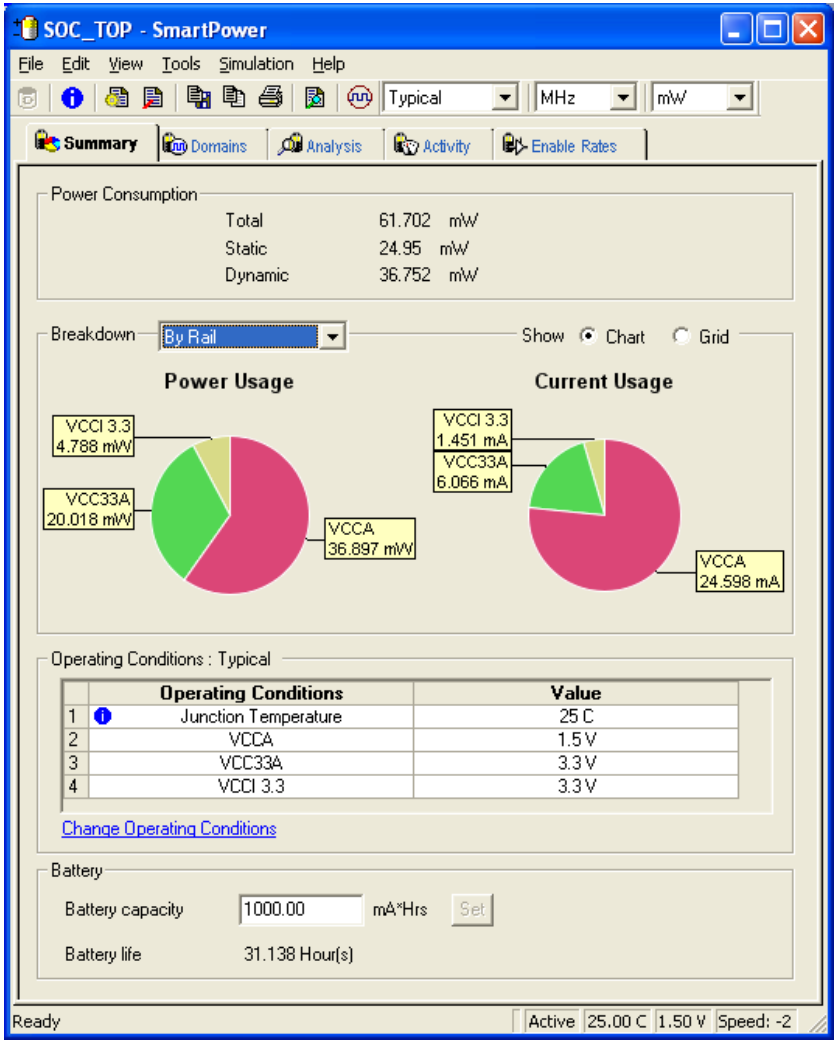
Power Breakdown – Grid View

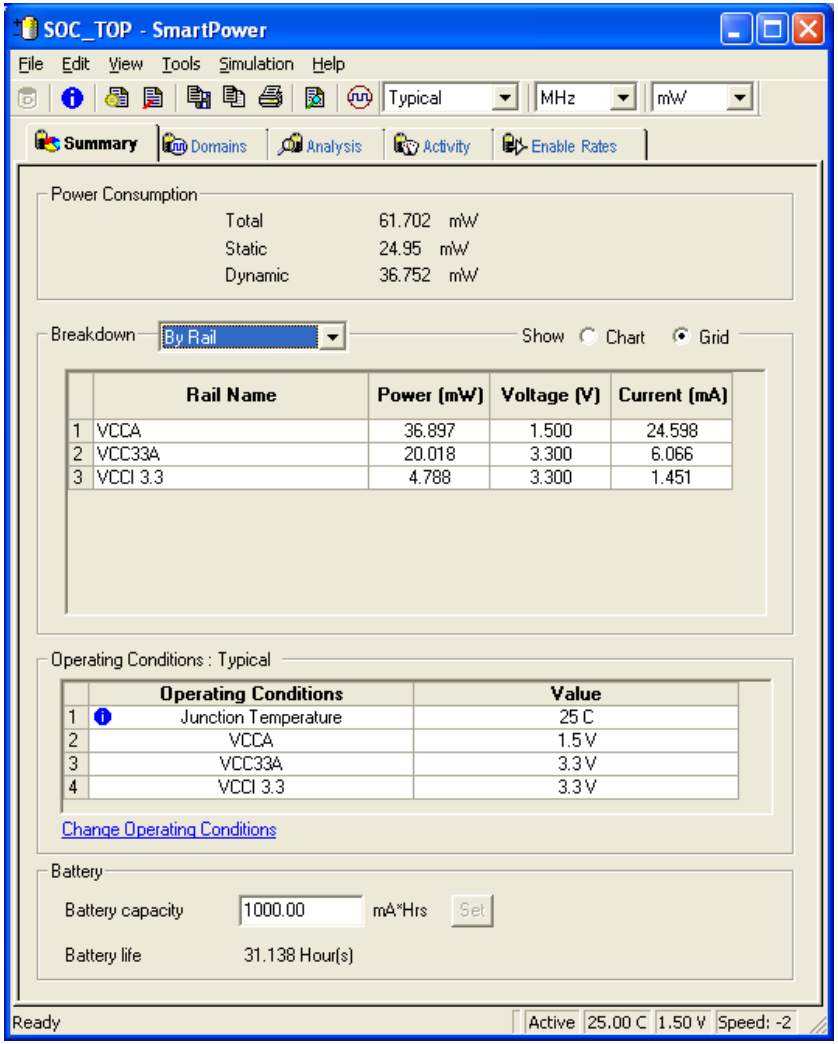


SmartPower Summary Tab

Power Breakdown by Power Rail

- Displays Power Consumption of all Power Rails

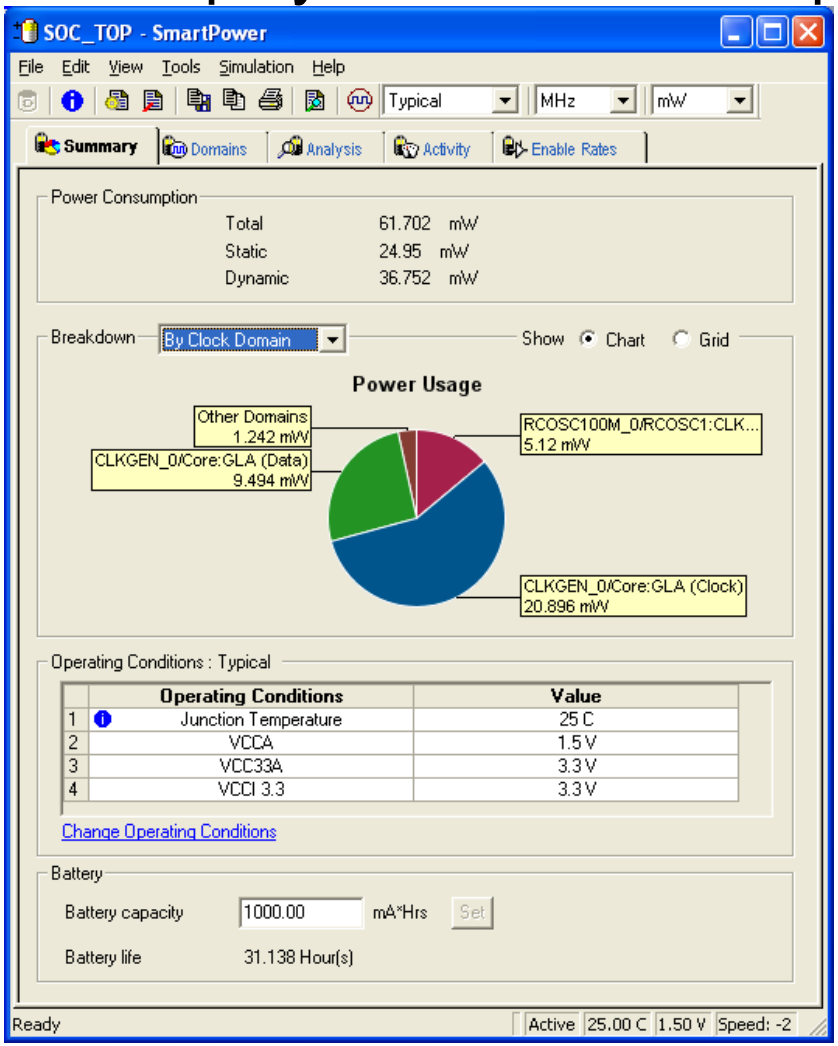


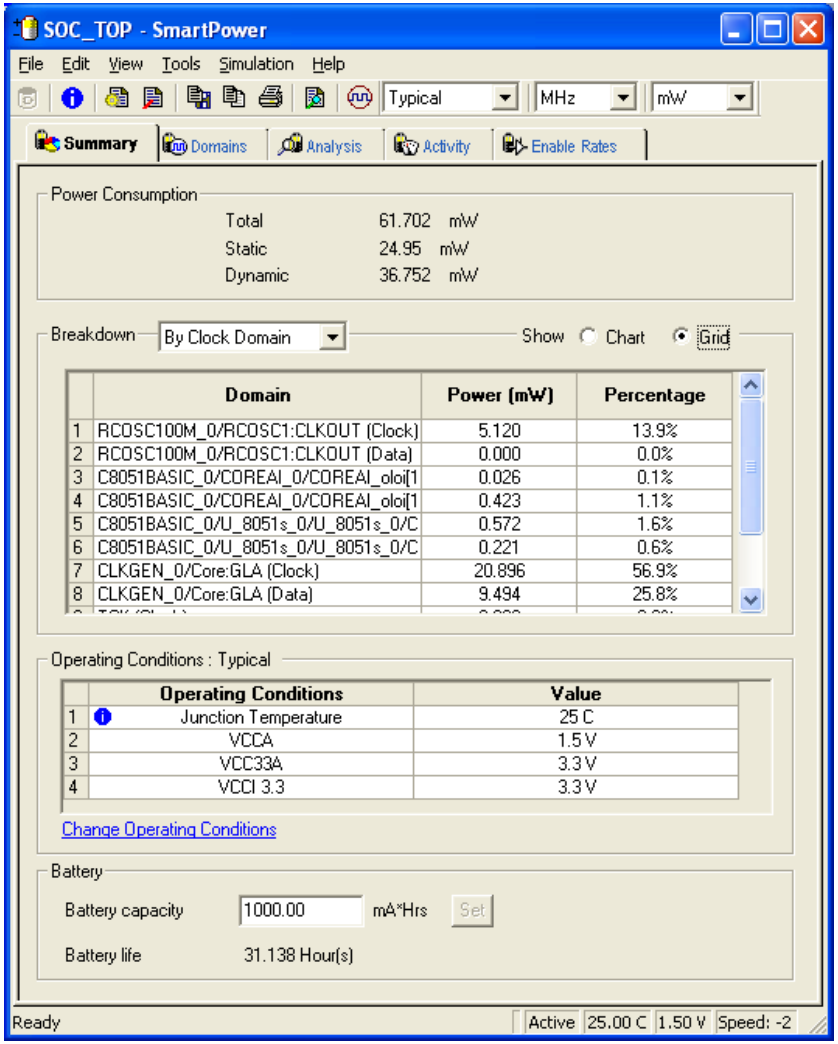


SmartPower Summary Tab

Power Breakdown by Clock Domain

- Displays Power Consumption of all Clock Domains





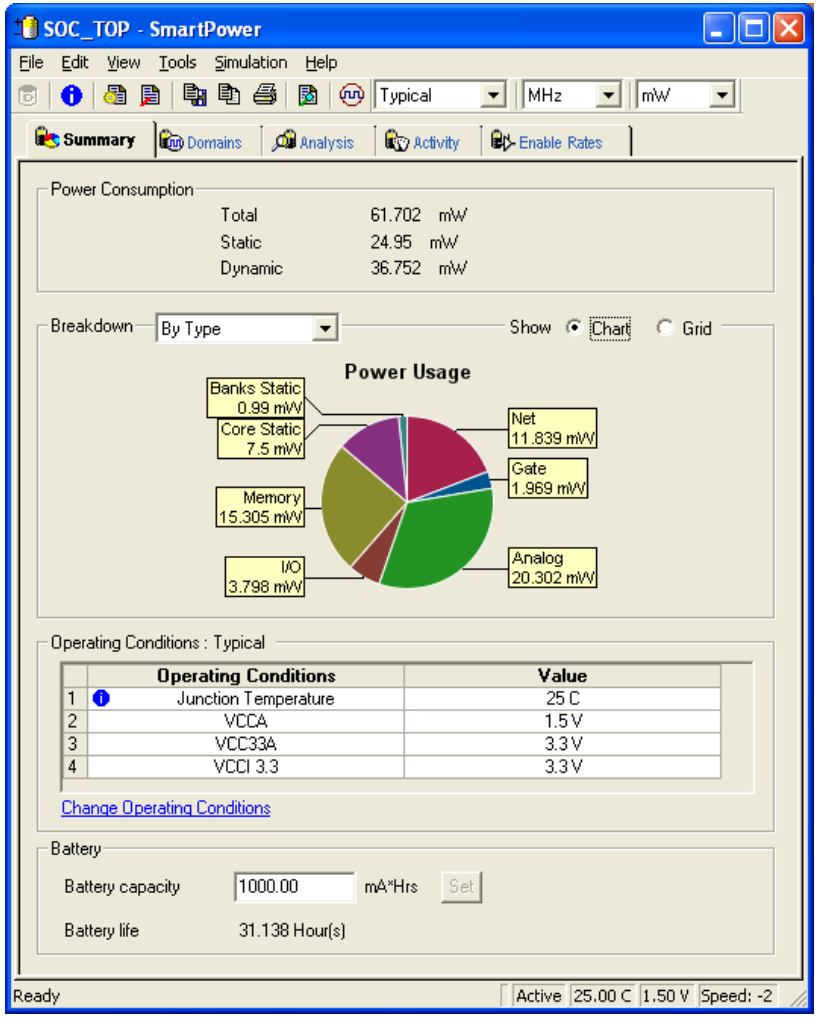
SmartPower Summary Tab

Junction Temperature

- SmartPower Estimates Junction Temperature (T_J)
- Junction Temperature can be Calculated from Operating Conditions or Ambient Temperature

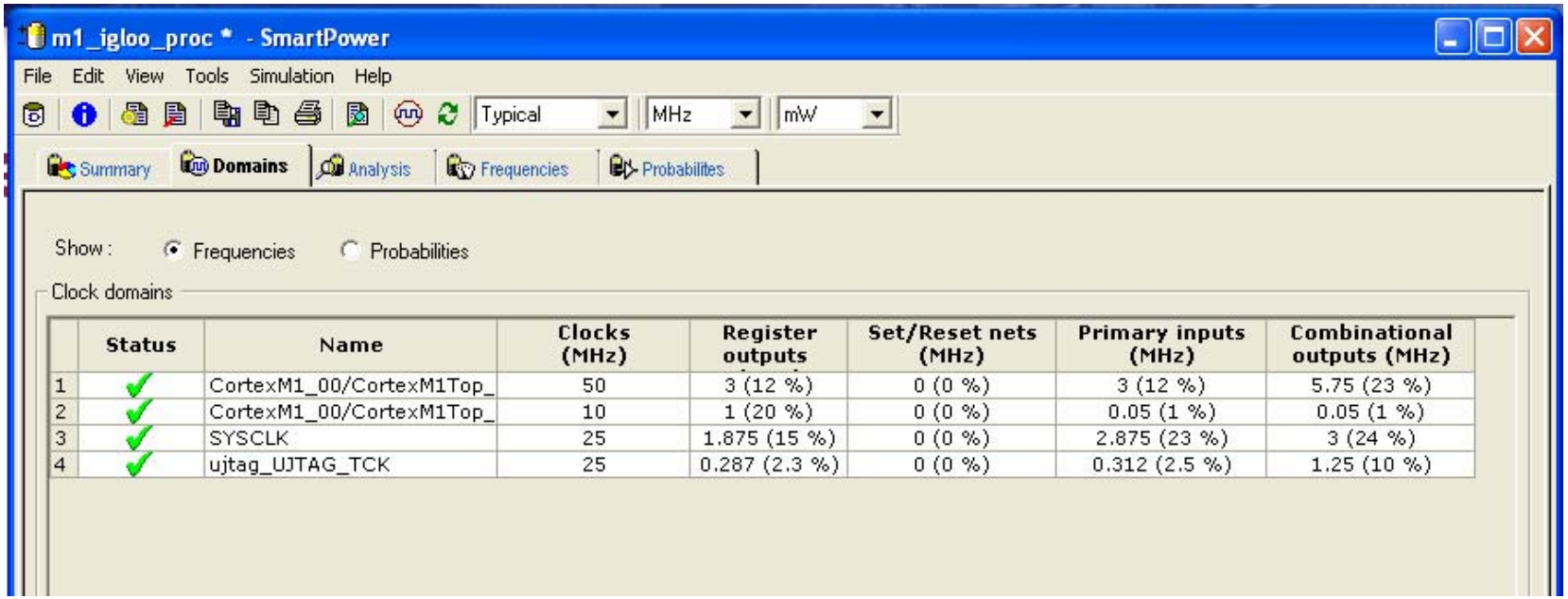
  $T_J = T_A + \theta_{JA} * (P_{dynamic} + P_{static})$

  θ_{JA} (Package Thermal Resistance)
- Cooling Scenarios:
  - Still Air (default)
  - 300 ft / min
  - Case Cooling
  - Custom
- SmartPower Reports the Thermal Resistance θ_{JA}



SmartPower Domain Tab (1)

- Actel Vector-less Approach
  - Clock Constraints extracted form SmartTime
  - A distinct toggle rate can be set per clock domain & per type of net

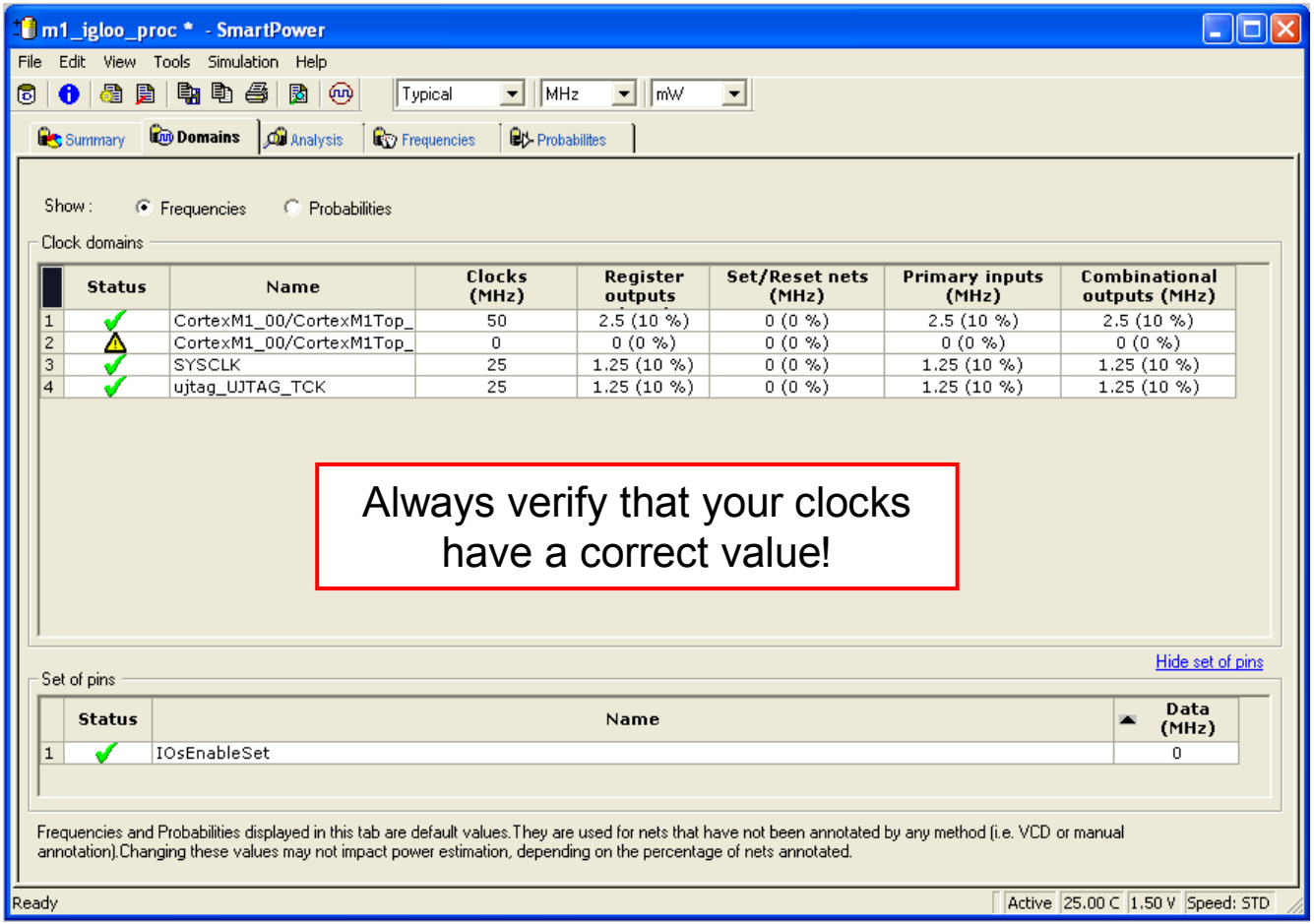


	Status	Name	Clocks (MHz)	Register outputs	Set/Reset nets (MHz)	Primary inputs (MHz)	Combinational outputs (MHz)
1	✓	CortexM1_00/CortexM1Top_	50	3 (12 %)	0 (0 %)	3 (12 %)	5.75 (23 %)
2	✓	CortexM1_00/CortexM1Top_	10	1 (20 %)	0 (0 %)	0.05 (1 %)	0.05 (1 %)
3	✓	SYSCLK	25	1.875 (15 %)	0 (0 %)	2.875 (23 %)	3 (24 %)
4	✓	ujtag_UJTAG_TCK	25	0.287 (2.3 %)	0 (0 %)	0.312 (2.5 %)	1.25 (10 %)

If you don't have simulation data, this step is crucial for the quality of the estimation. Beware that Garbage-In => Garbage-Out

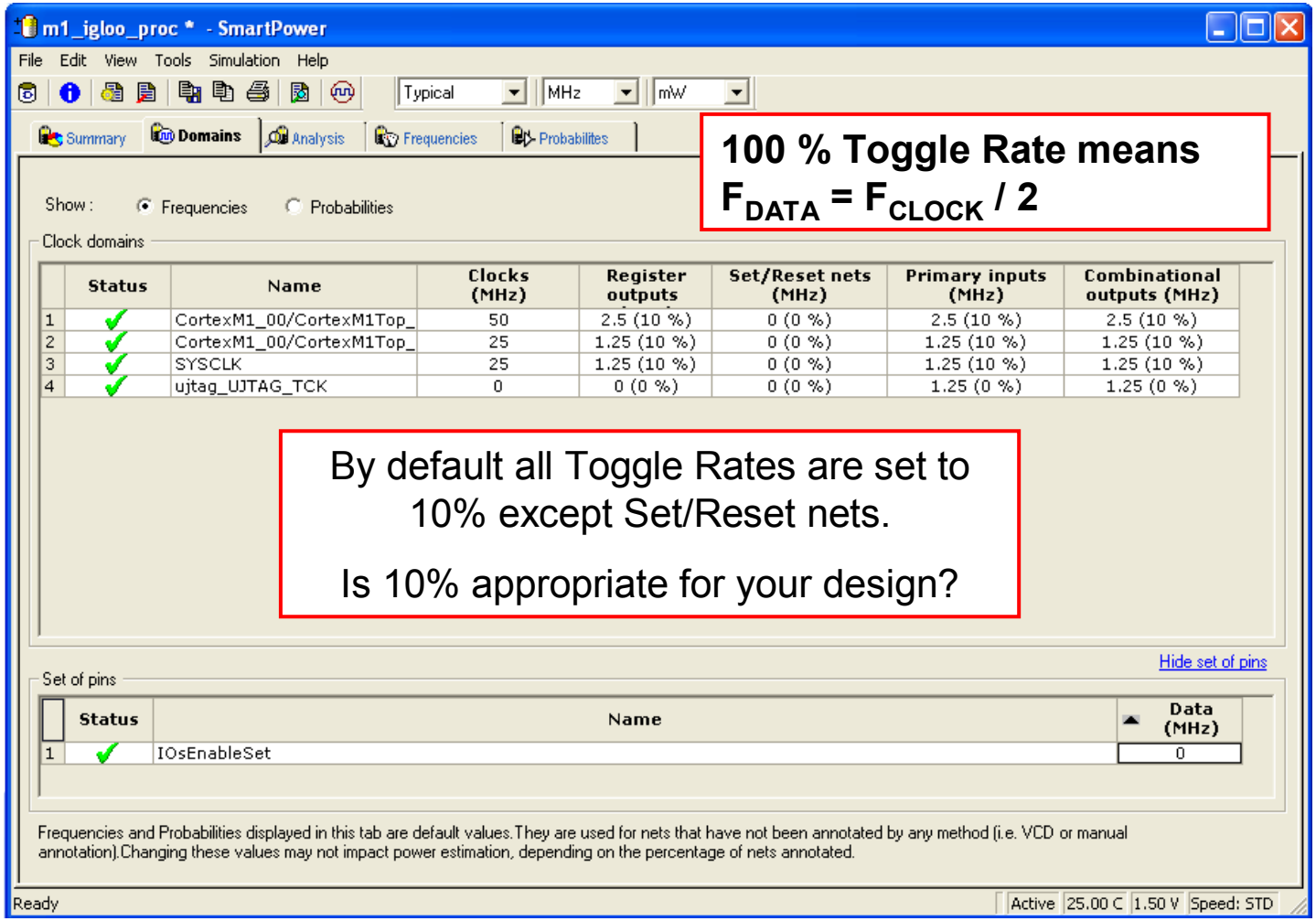
SmartPower Domain Tab (2)

- Garbage-In => Garbage-Out…. What Do We mean?
  - Default Vector-less flow: Clock Constraints are taken from SmartTime



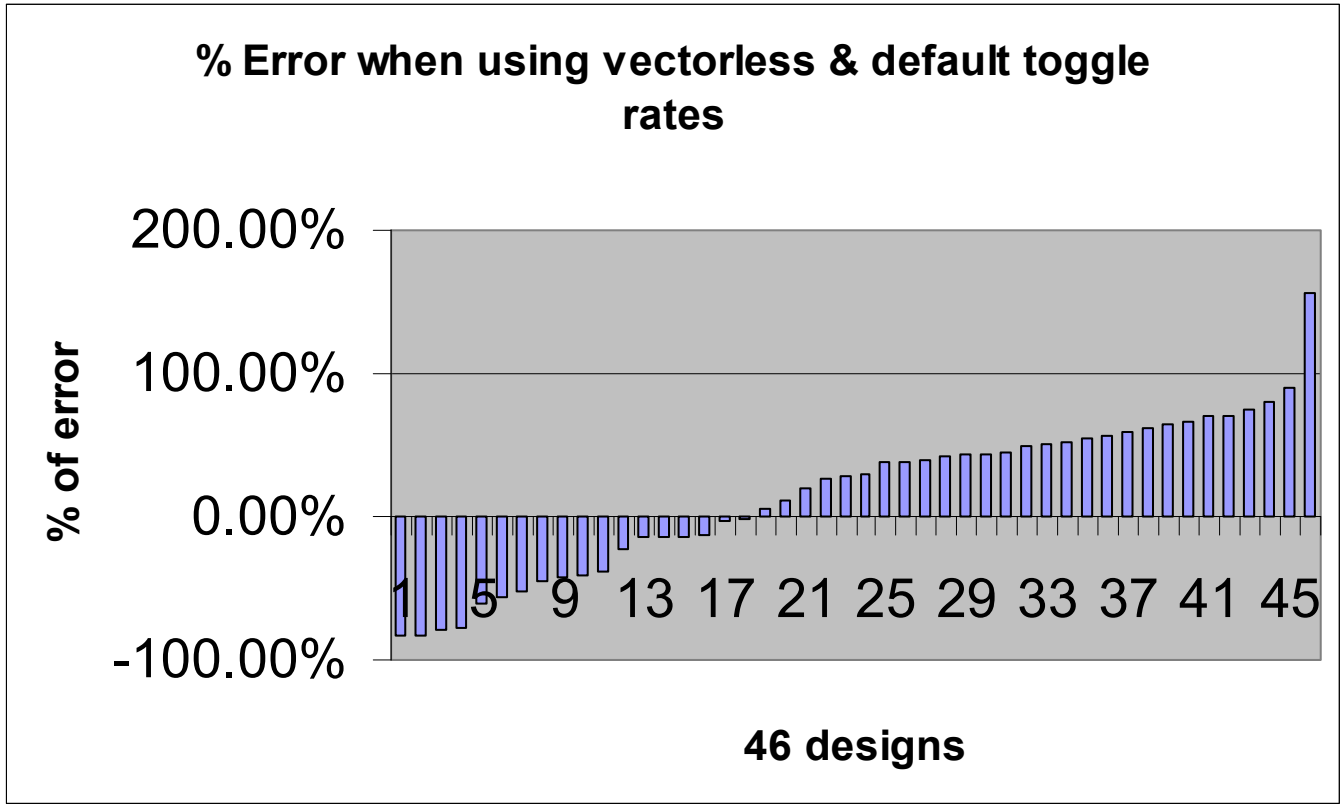
SmartPower Domain Tab (3)

- Garbage-In => Garbage-Out…. What Do We mean?
  - Default Vector-less flow: All Toggle Rates are set to 10%



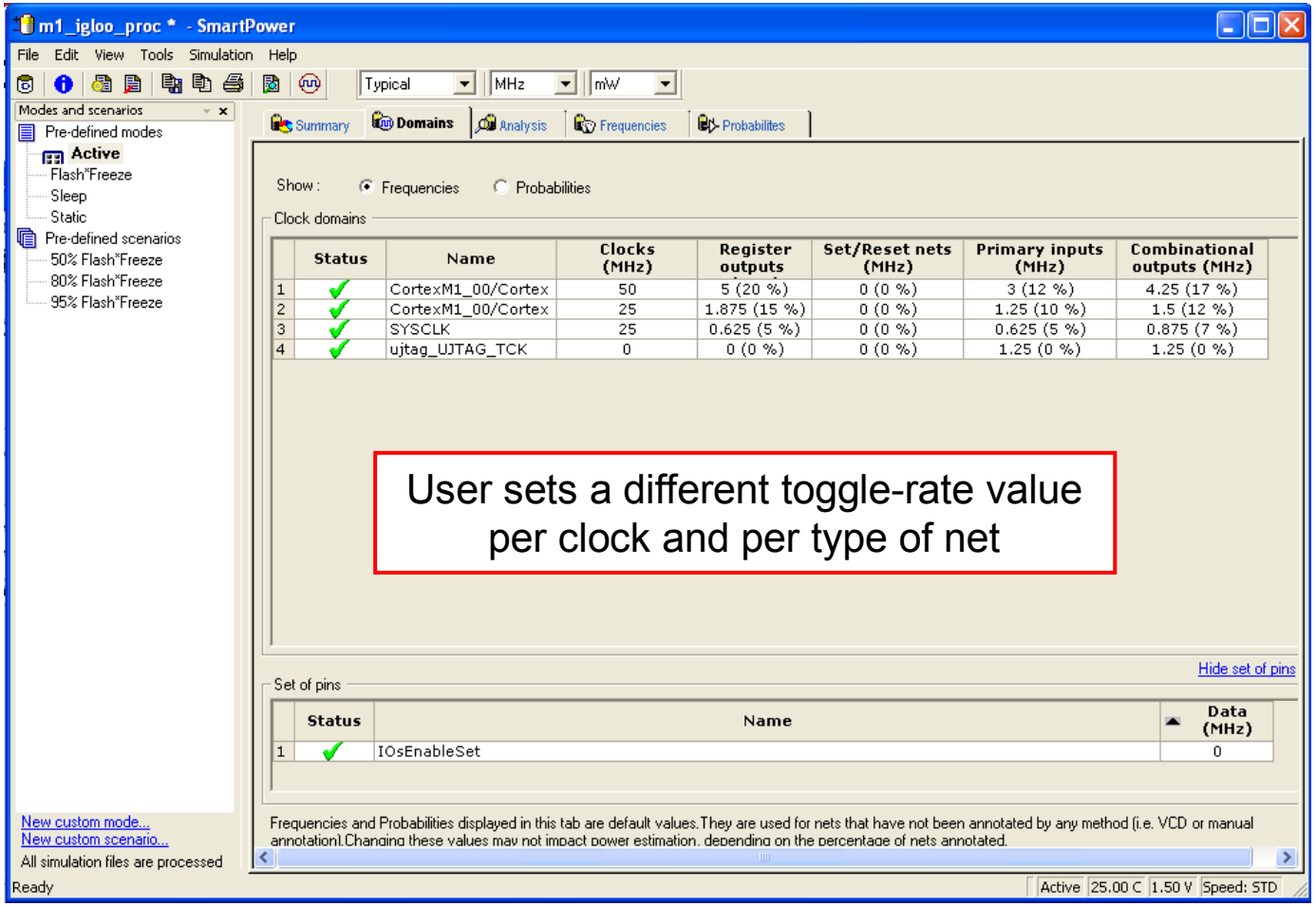
SmartPower Domain Tab (4)

- Default vector-less flow with 10% toggle rates leads to +/- 100% error when compared to a simulation flow



SmartPower Domain Tab (5)

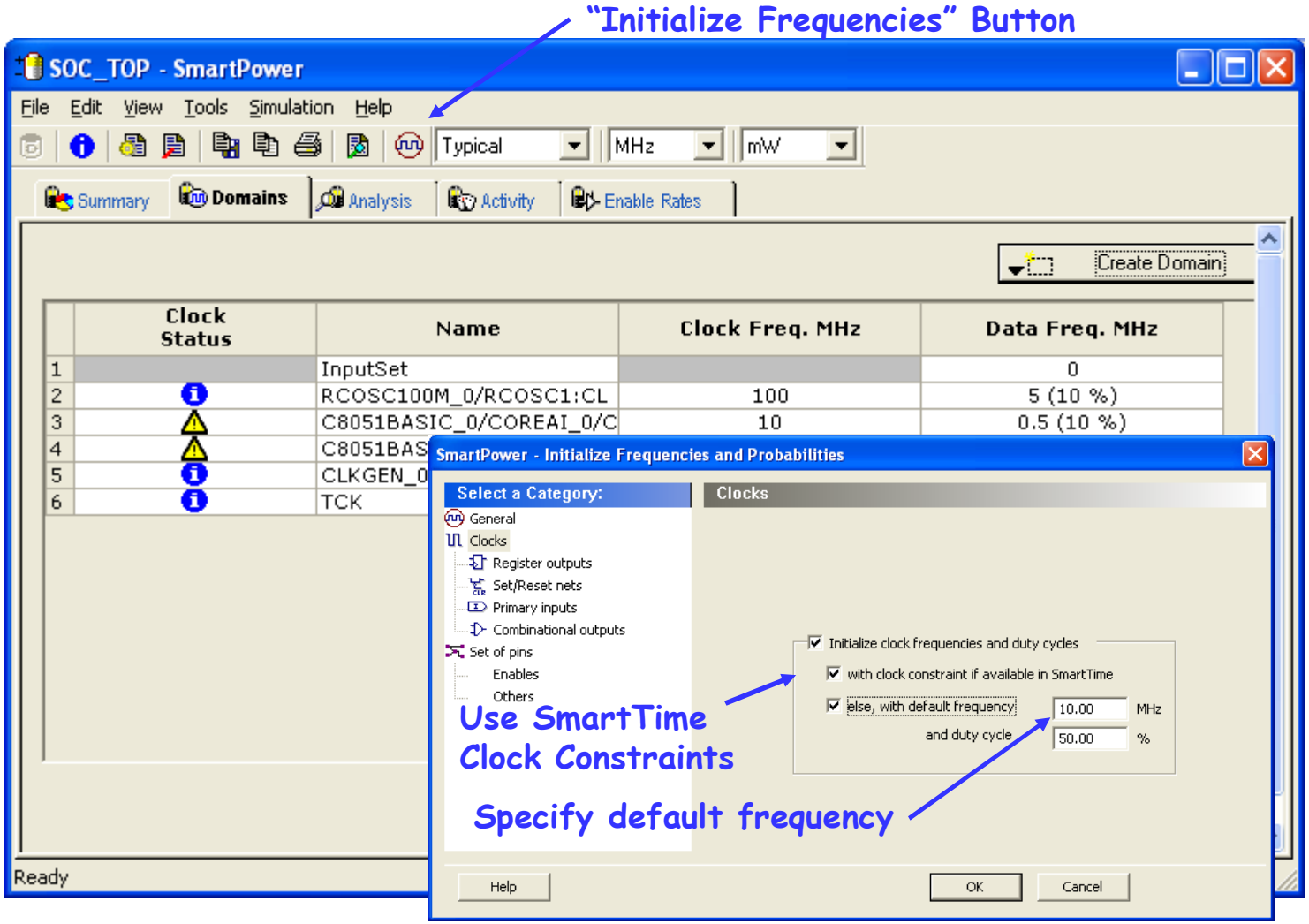
- Improved Vector-less flow: Design specific Toggle Rates



SmartPower Domains Tab

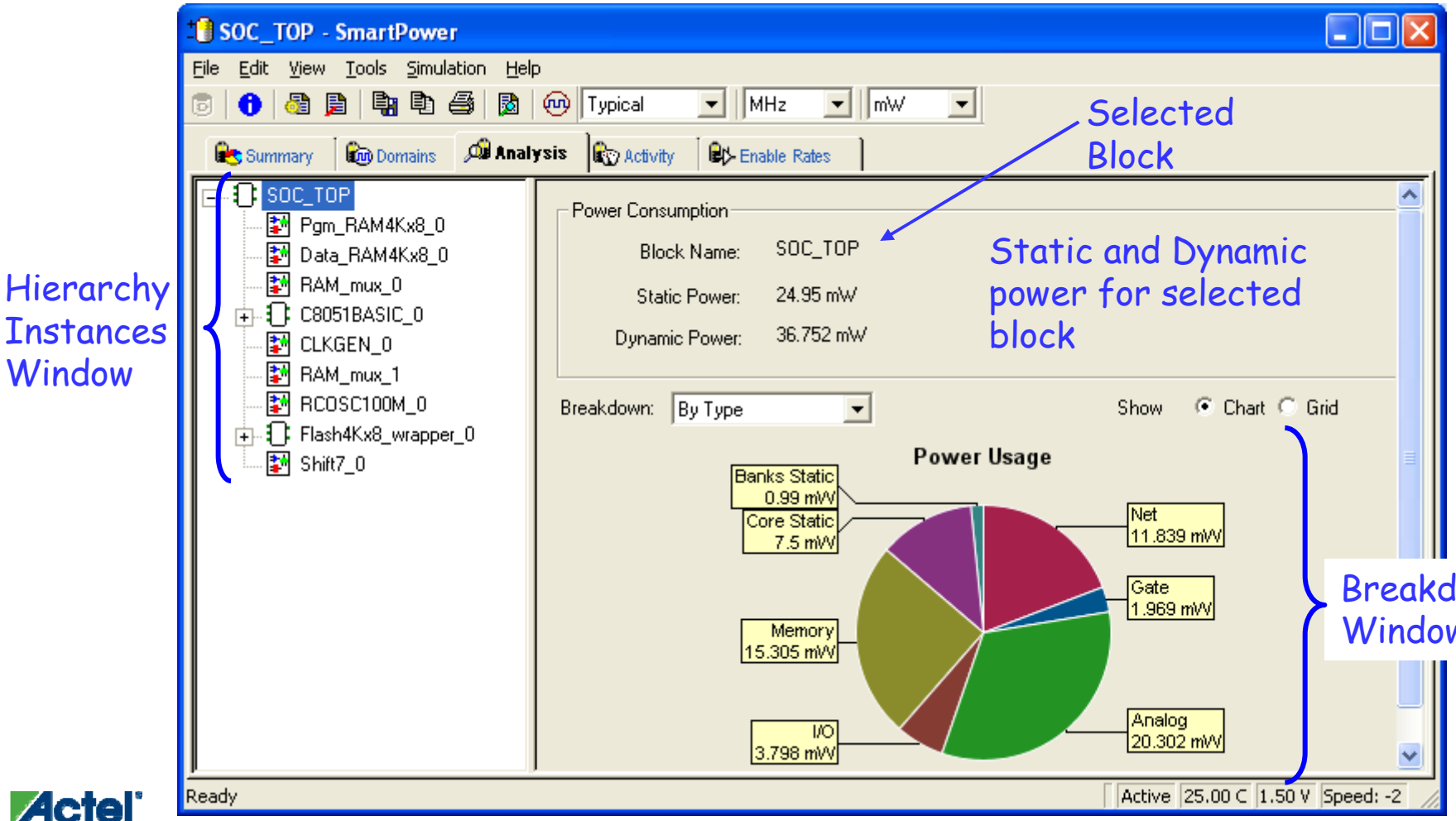
Initialize Frequencies

- Tools > Initialize Frequencies and Probabilities



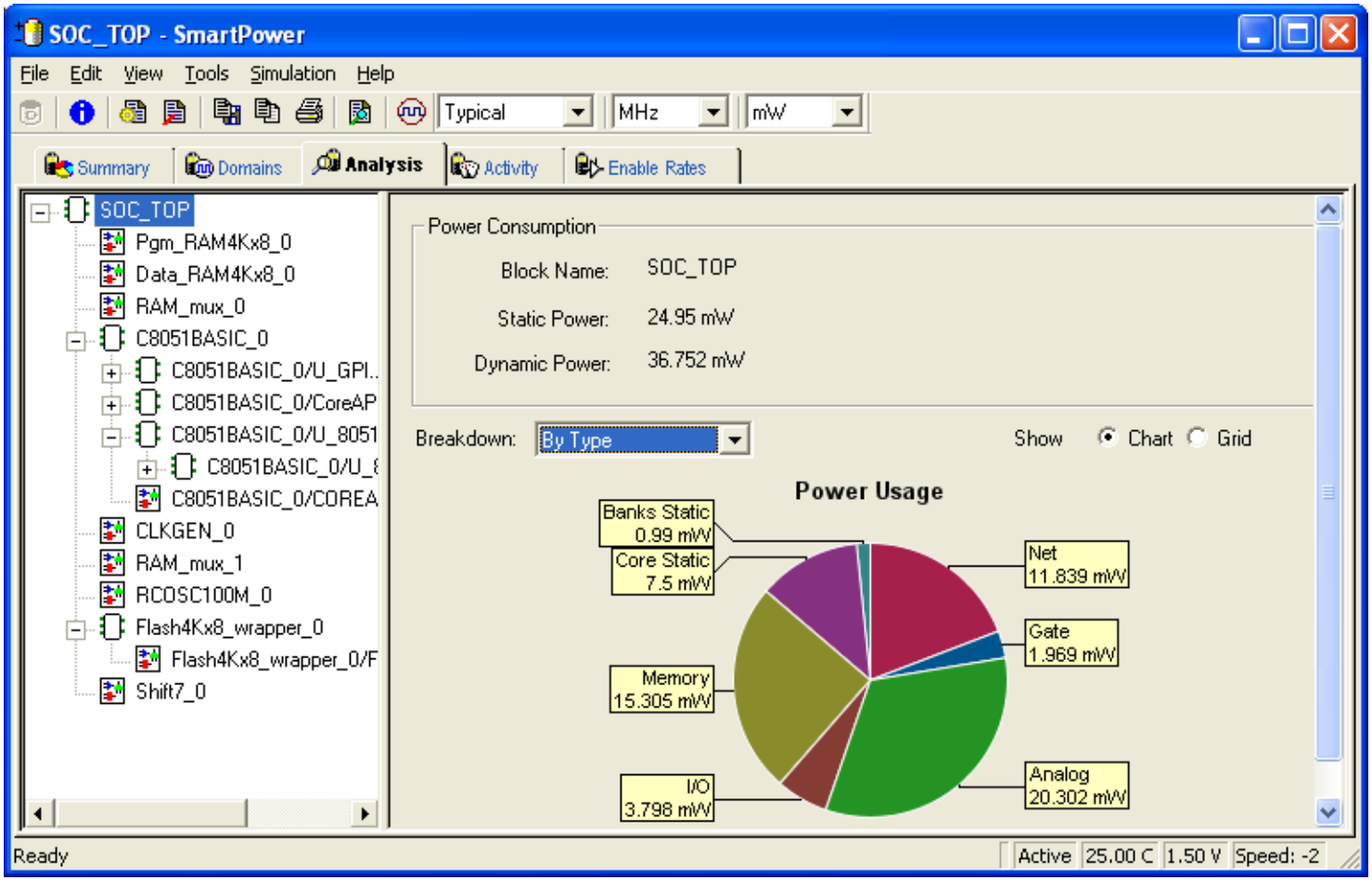
SmartPower Analysis Tab

- Provides Hierarchical Report of Dynamic Power Consumption



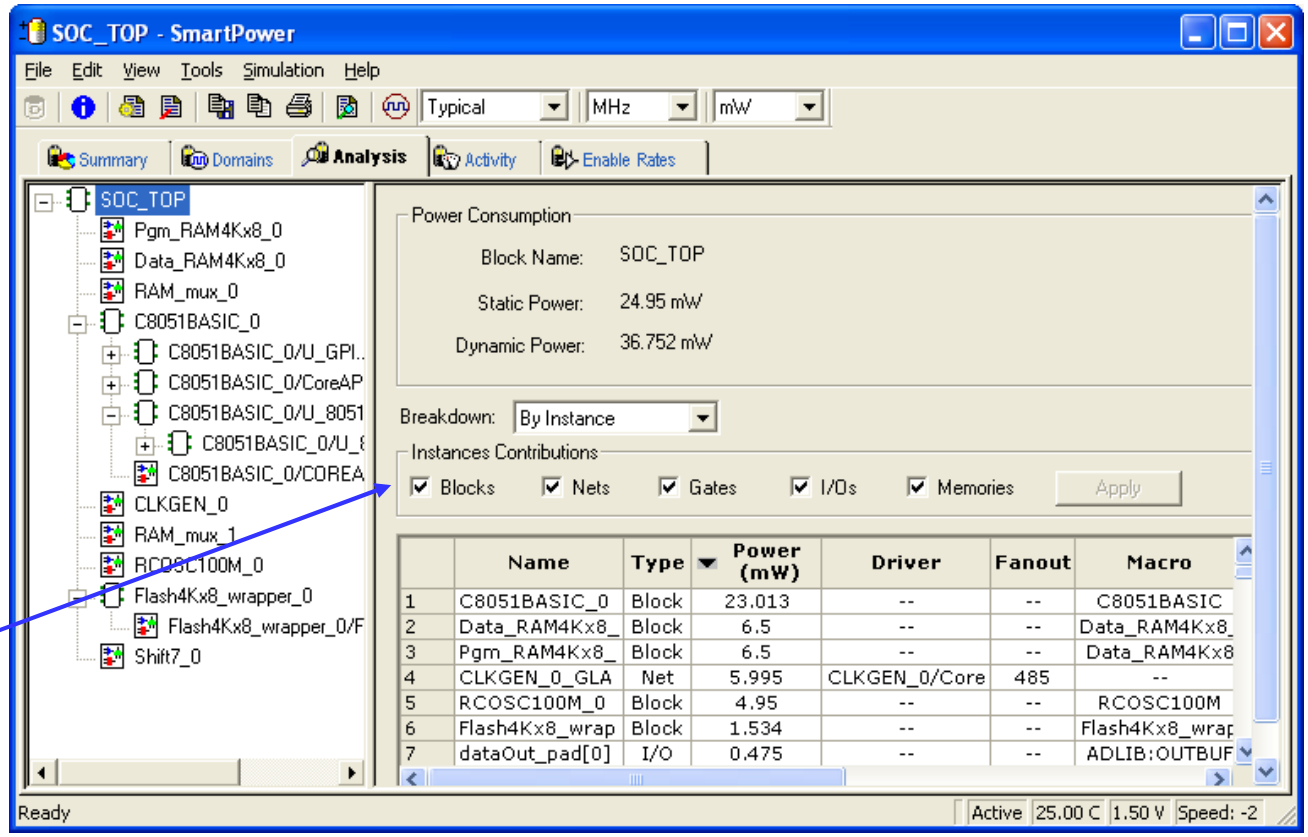
SmartPower Analysis Tab
Power Breakdown By Type

- Analysis Grid Displays Power Breakdown for Selected Block by:
  - Net, Gate, I/O, Memory, Clock, Core Static, Banks Static
  - Grid or pie chart display



SmartPower Analysis Tab
Power Breakdown By Instance

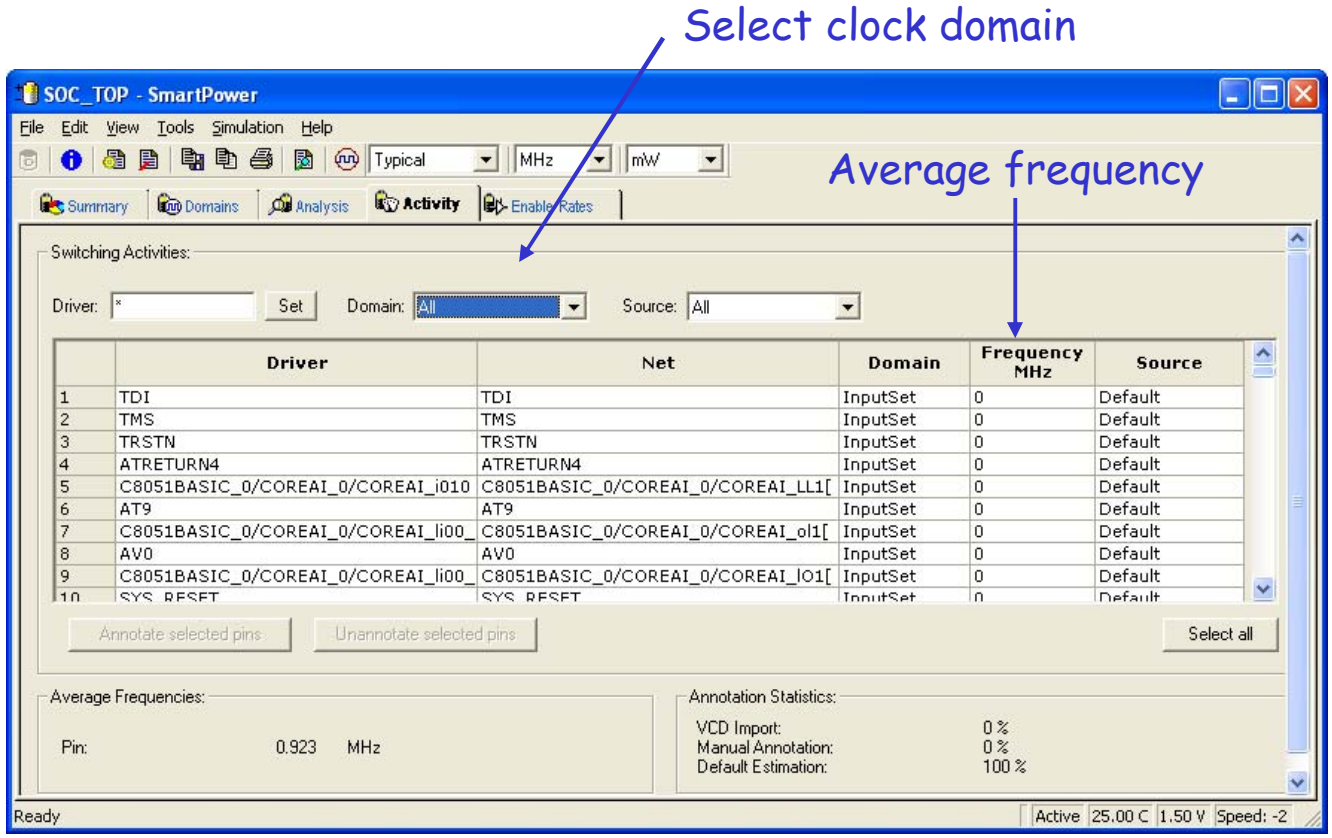
- Analysis Grid Displays Power of all Sub-elements of Selected Block



Limit components to be displayed by checking or un-checking boxes

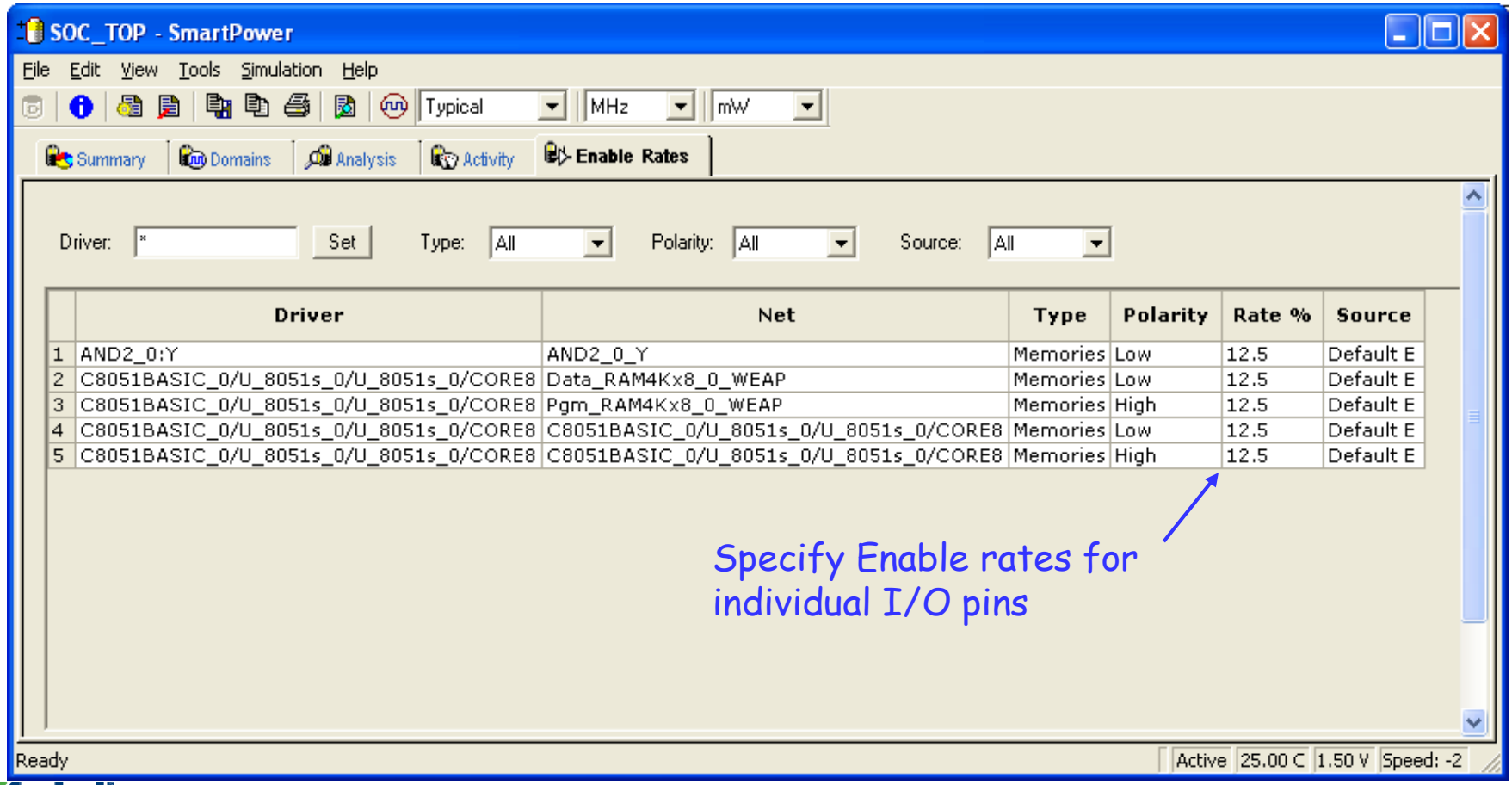
SmartPower Activity Tab

- Allows Entry of Switching Activity Information on Interconnects of Design
  - Use to Confirm Data Entered through VCD or SAIF File Import or pin-by-pin annotation



SmartPower Enable Rates Tab

- Specify Output Enable Rate for Each Bi-directional and Tri-state I/O
  - Percentage of Time I/O is used as an Output

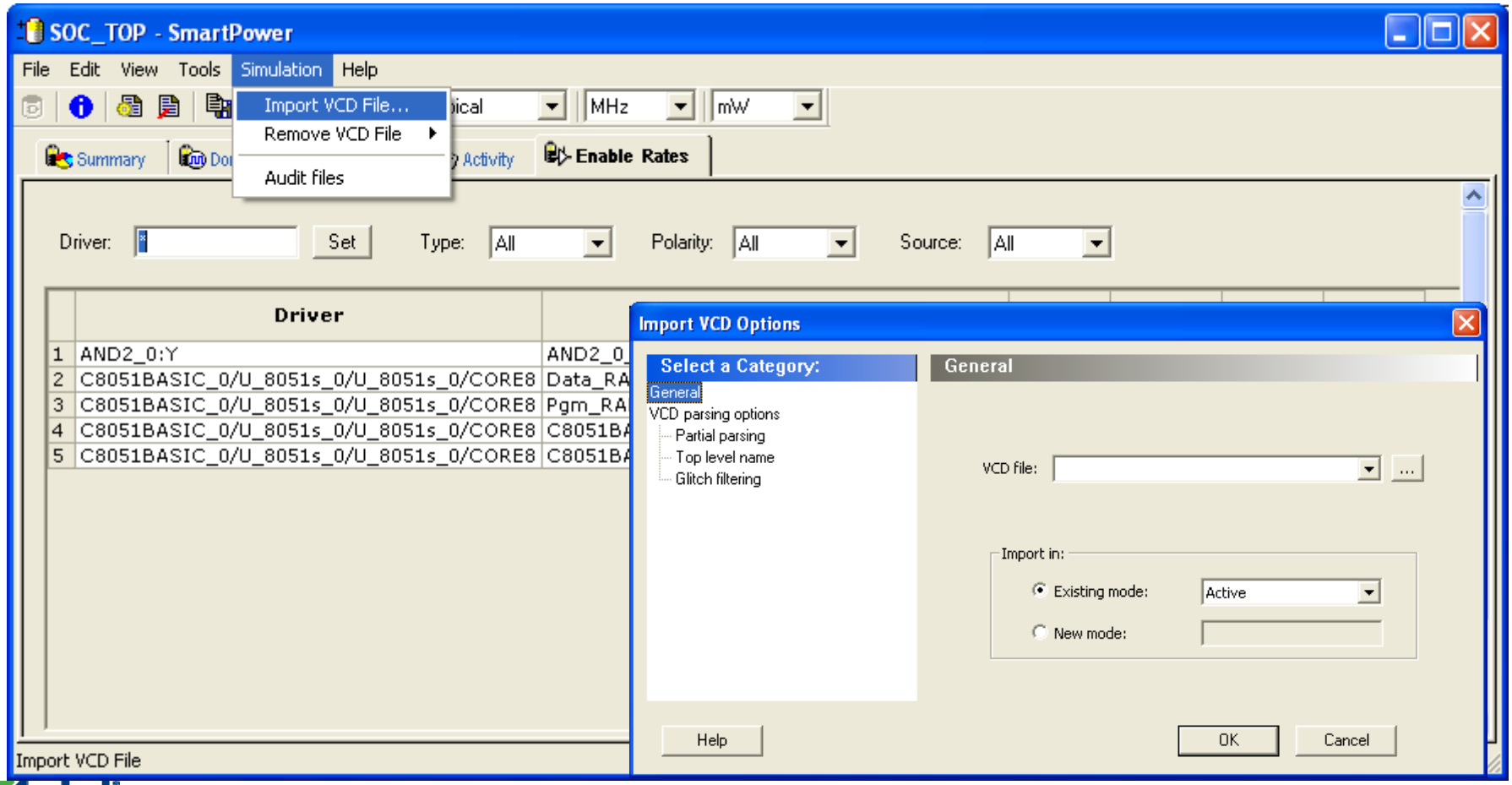


	Driver	Net	Type	Polarity	Rate %	Source
1	AND2_0:Y	AND2_0_Y	Memories	Low	12.5	Default E
2	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	Data_RAM4Kx8_0_WEAP	Memories	Low	12.5	Default E
3	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	Pgm_RAM4Kx8_0_WEAP	Memories	High	12.5	Default E
4	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	Memories	Low	12.5	Default E
5	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	C8051BASIC_0/U_8051s_0/U_8051s_0/CORE8	Memories	High	12.5	Default E

SmartPower

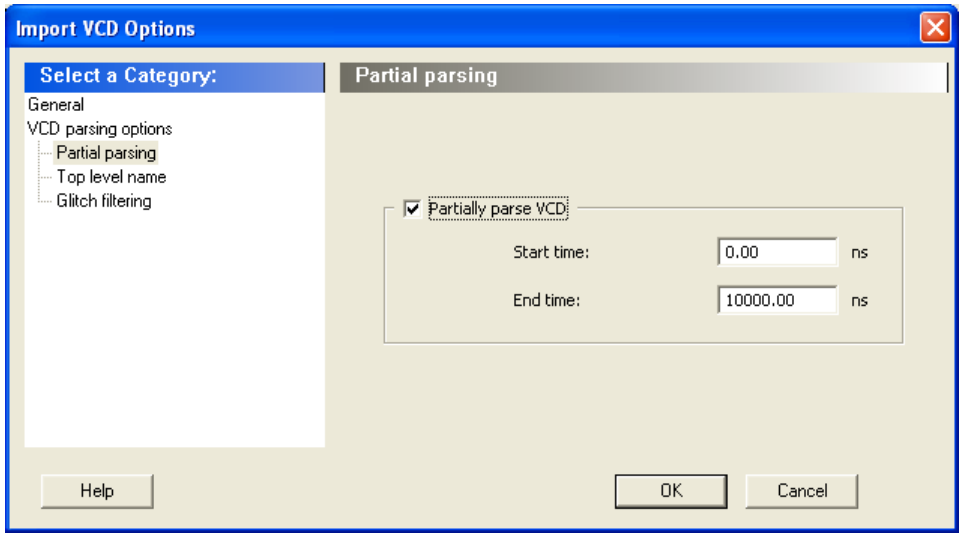
Importing a VCD or SAIF File

- Switching information for the design can be entered by importing a VCD or SAIF file

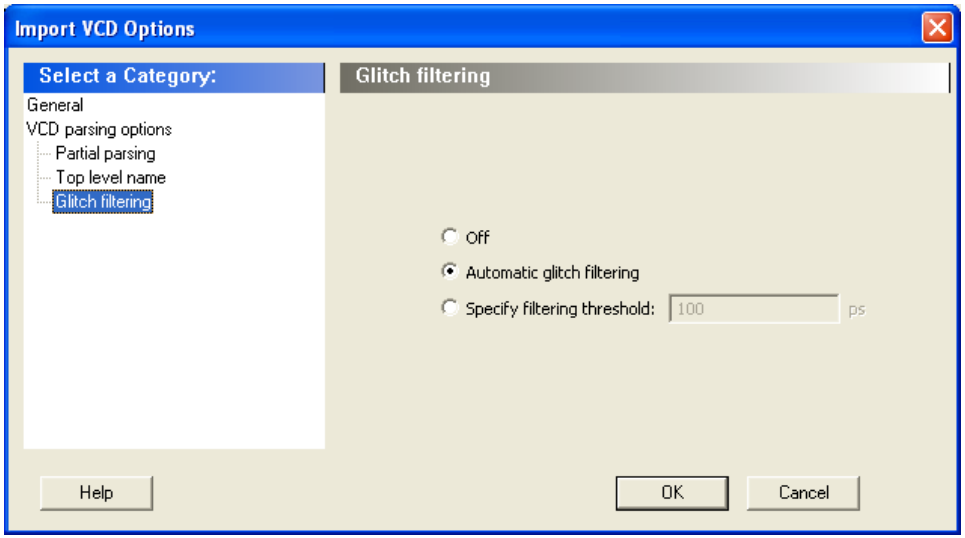


Simulation File Import Options

Specify start and end times for partial parsing of simulation file

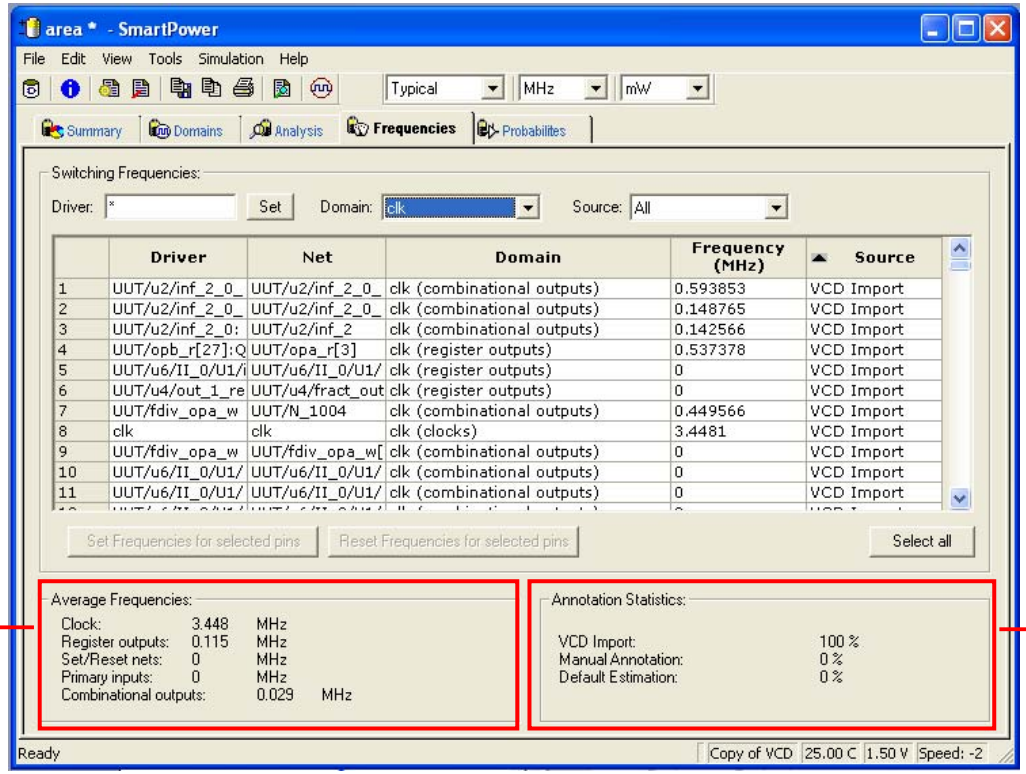


Specify glitch filtering
•Automatic or user-defined threshold



Simulation Based Flow

- Verify Simulation Coverage and Frequencies

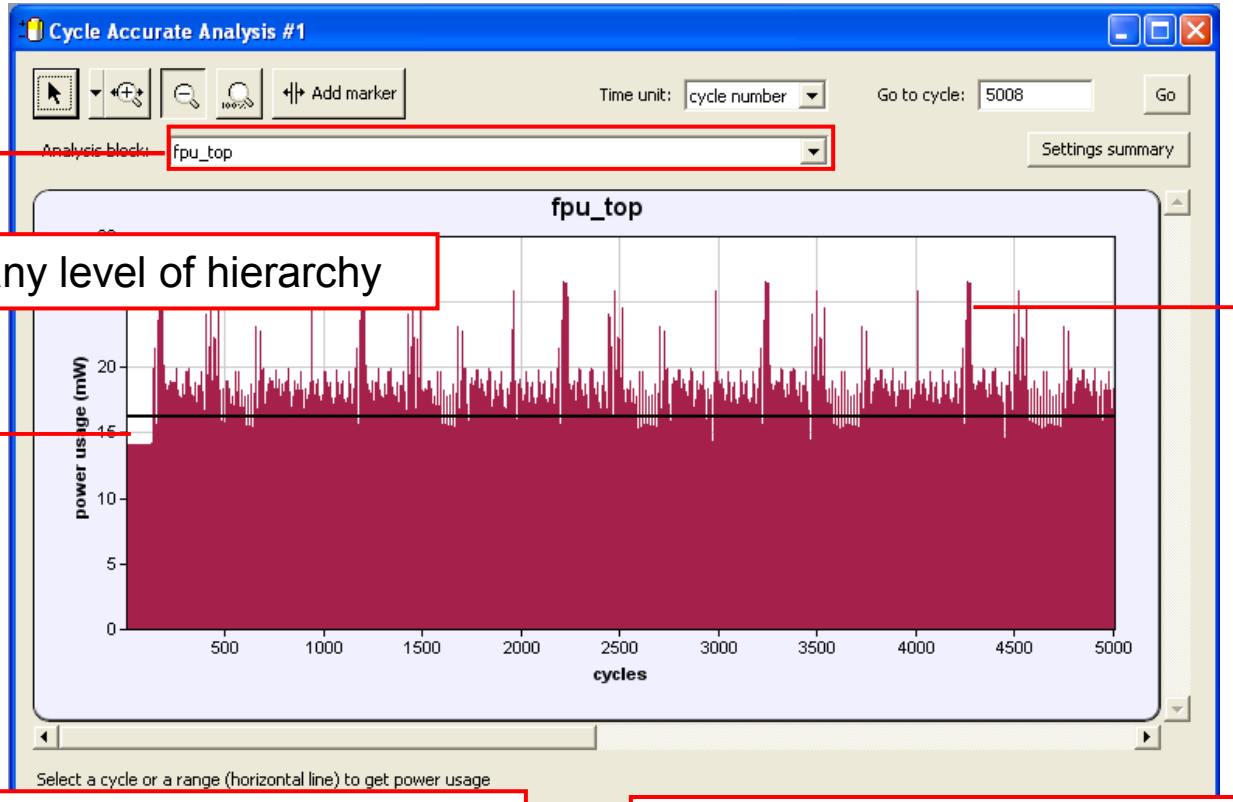


You can also check your Frequencies by type of element: clocks, inputs, set/reset…

If VCD import < 95%, I would suspect wrong VCD or behavioral VCD

Simulation Based Flow
Cycle Accurate Power Analysis

- Use Cycle Accurate Analysis to "*debug*" your Simulation in Terms of Power



You can debug any level of hierarchy

At the very minimum, you can verify that the average is not impacted too much by the initial reset.

Verifying that your power peaks are happening when they are expected

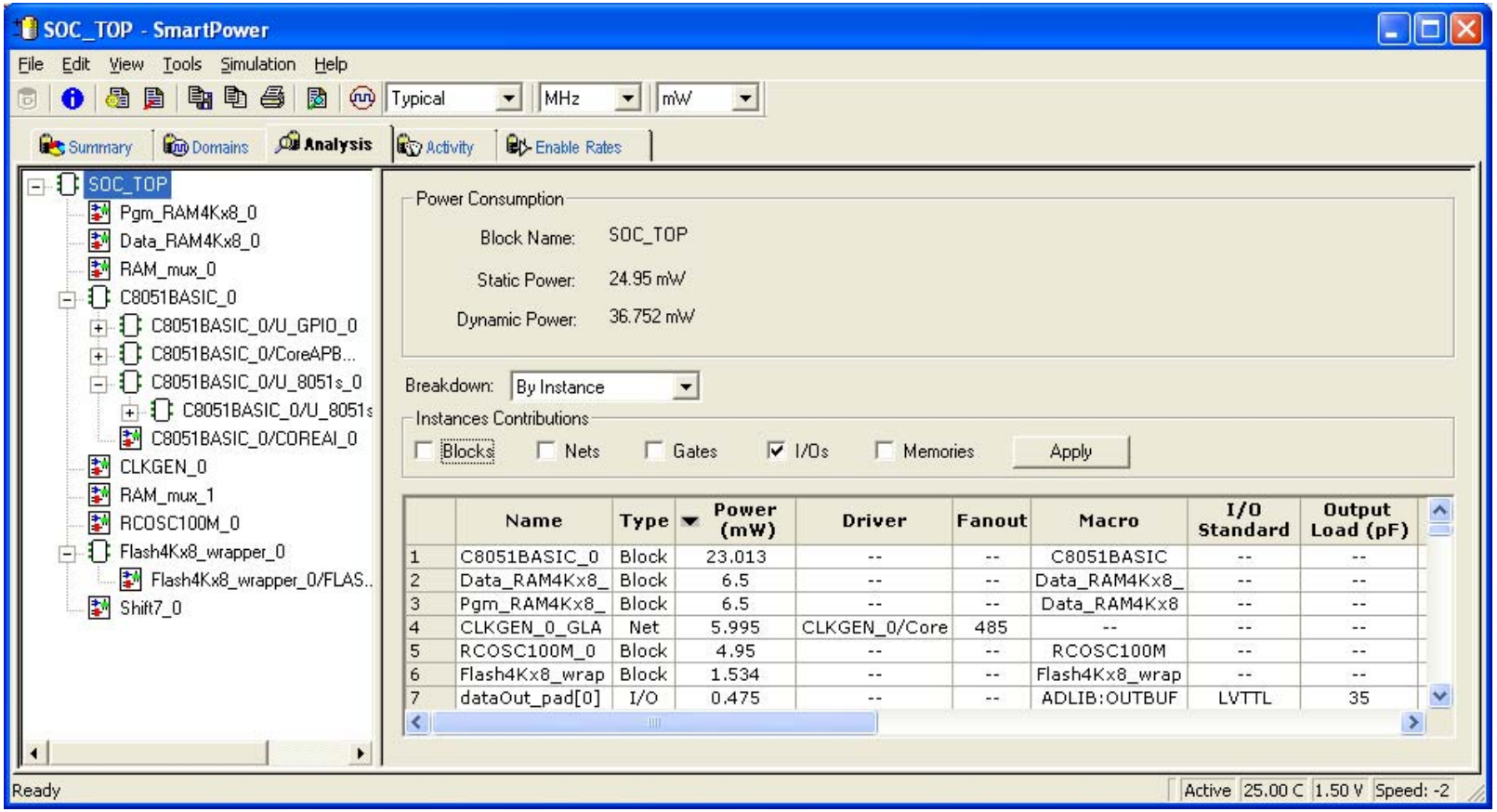
Power Analysis
Power Consumption For Specific Clock Domain

- Confirm SmartPower is in Toggle-rate Mode (Default Setting)
  - Confirm “Use Toggle Rates” Option is Checked in SmartPower Preferences
- Set all the Clock Frequencies Except the Clock Domain of Interest to Zero in the Domains tab
  - The Total Power Consumption for Clock Domain is Displayed in the Summary tab
    - Includes the contribution of the clock buffer, clock tree, and all clock input of the registers of specific clock domain
- To Determine the Power of the Clock Tree Only, set the Toggle Rate of the Data for the Clock Domain to Zero
  - Total Power Displayed is the Power of the Clock Tree

Power Analysis

Dynamic I/O Power – Analysis Tab

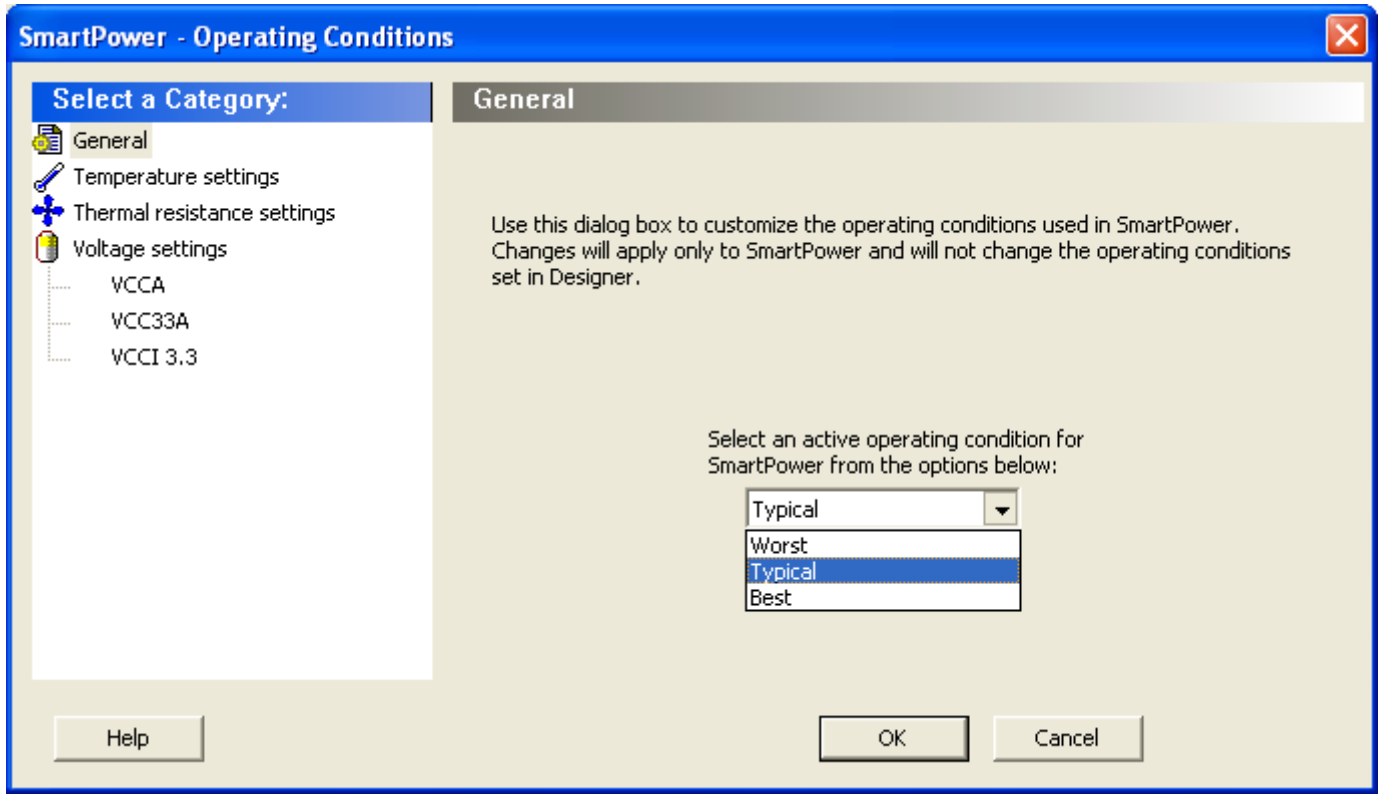
- Select the Highest Hierarchy Level in the Hierarchy tree in the left pane and Select “By Instance” from the Breakdown drop-down list
- Un-check all categories under Instances Contributions except I/Os
  - The table displays a list of all I/Os in the design and the power contribution of each individual I/O



SmartPower Operating Conditions

SmartPower Junction Temperature
Operating Conditions - General

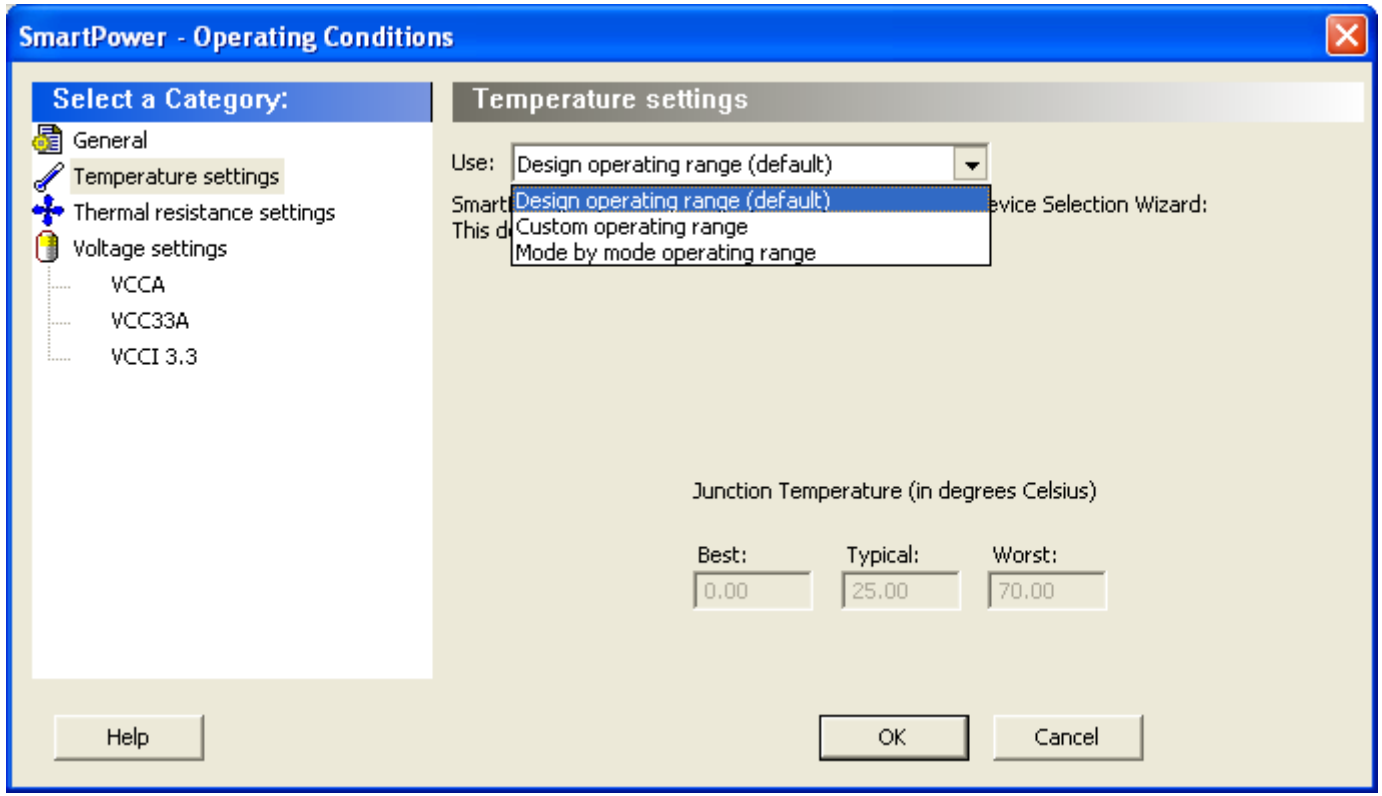
- Select Operating Conditions



SmartPower Junction Temperature

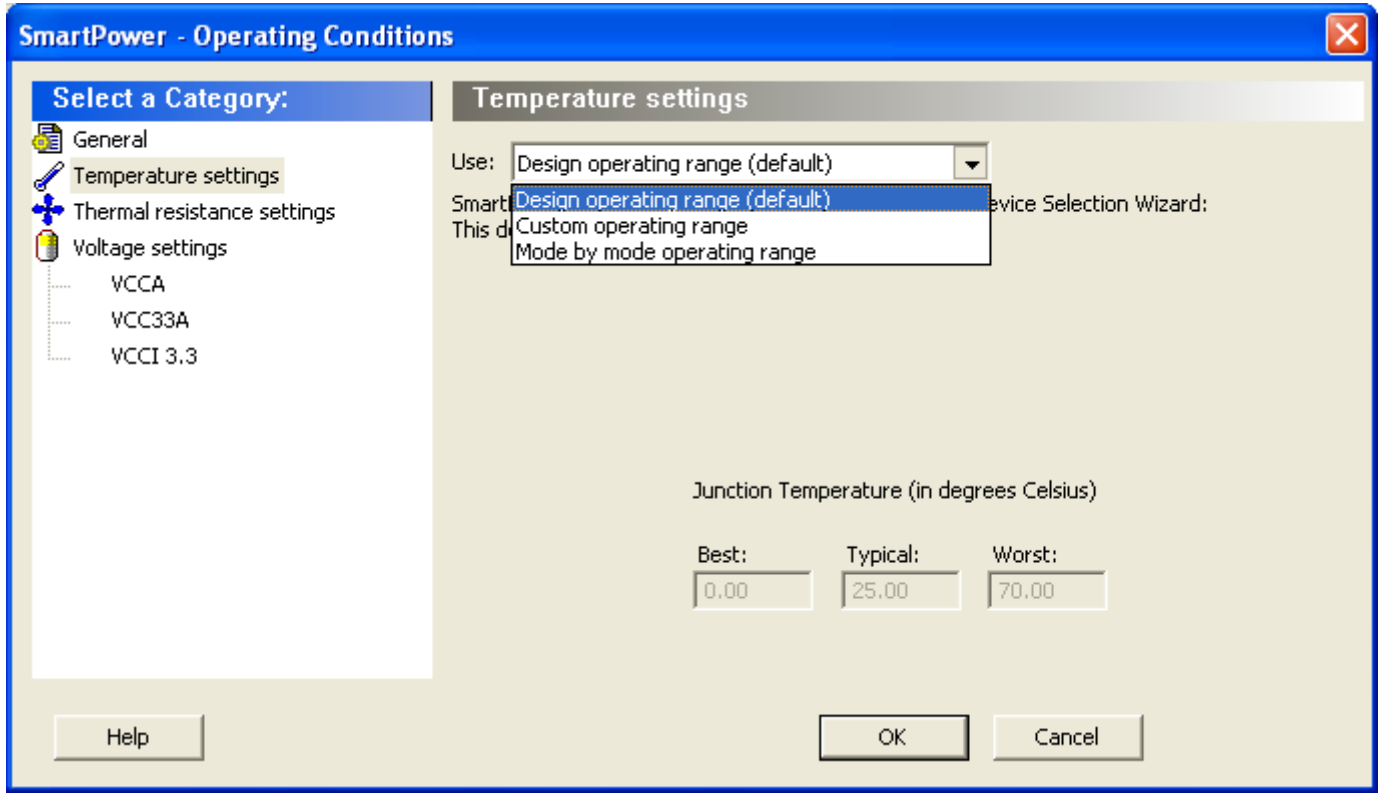
Operating Conditions – Temperature Settings

- Define Temperature Settings:
  - Default, Custom Operating range or Mode by Mode range



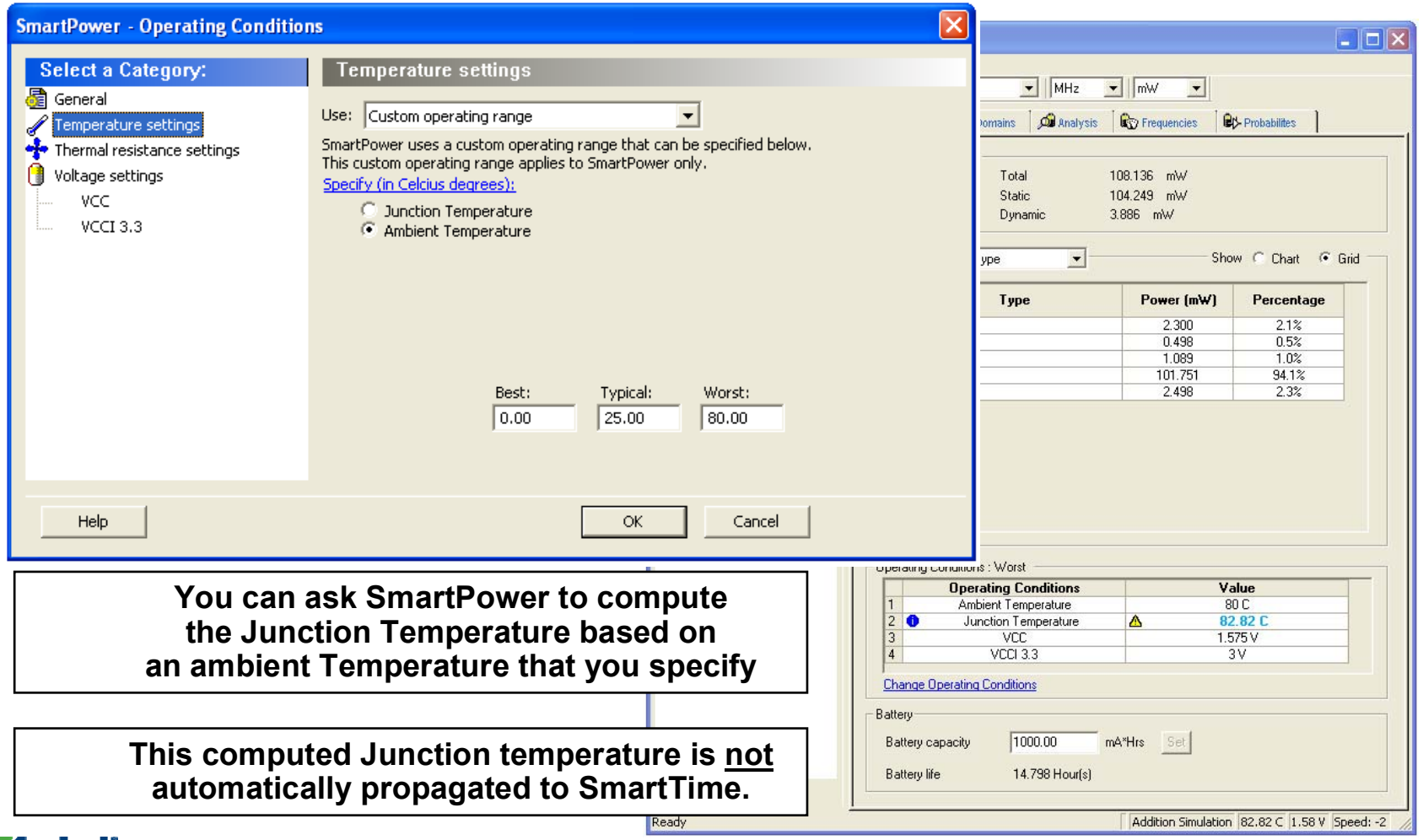
SmartPower Junction Temperature
Operating Conditions – Temperature Settings

- Define Temperature Settings:
  - Default, Custom Operating range or Mode by Mode range
    - Specify ambient or junction temperature in custom settings



SmartPower Custom Ambient Temp

- Advanced: Custom Ambient Temperature Settings:



You can ask SmartPower to compute the Junction Temperature based on an ambient Temperature that you specify

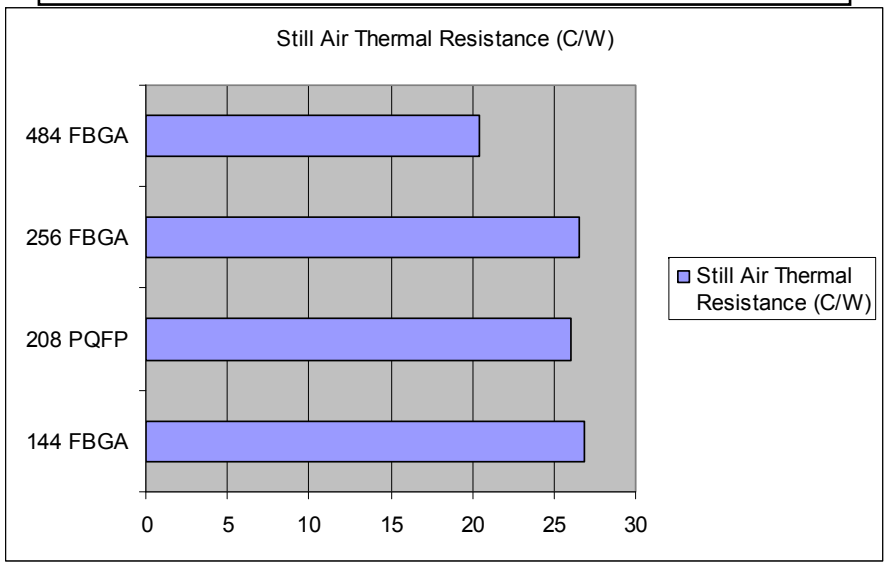
This computed Junction temperature is <u>not</u> automatically propagated to SmartTime.

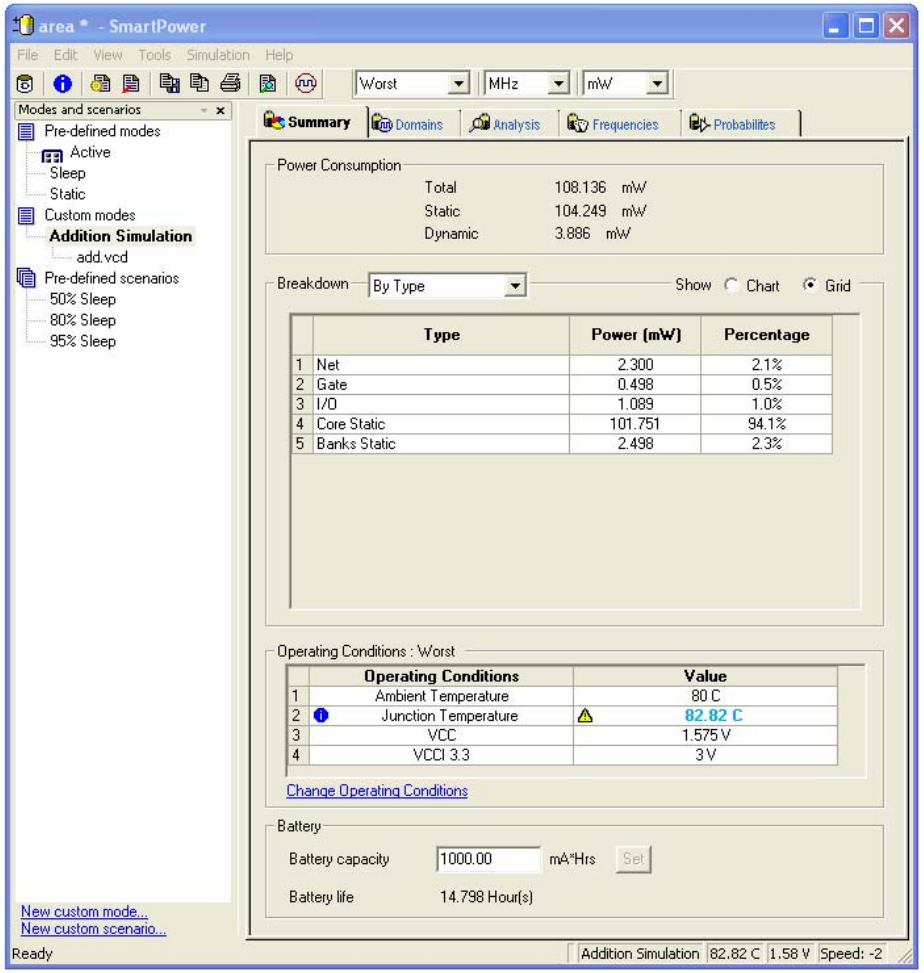
SmartPower Computes Junction Temp

- Junction Temperature Computation:

T-Junction = T-Ambient + P*θ_{JA}

θ_{JA} is a function of the package and cooling style

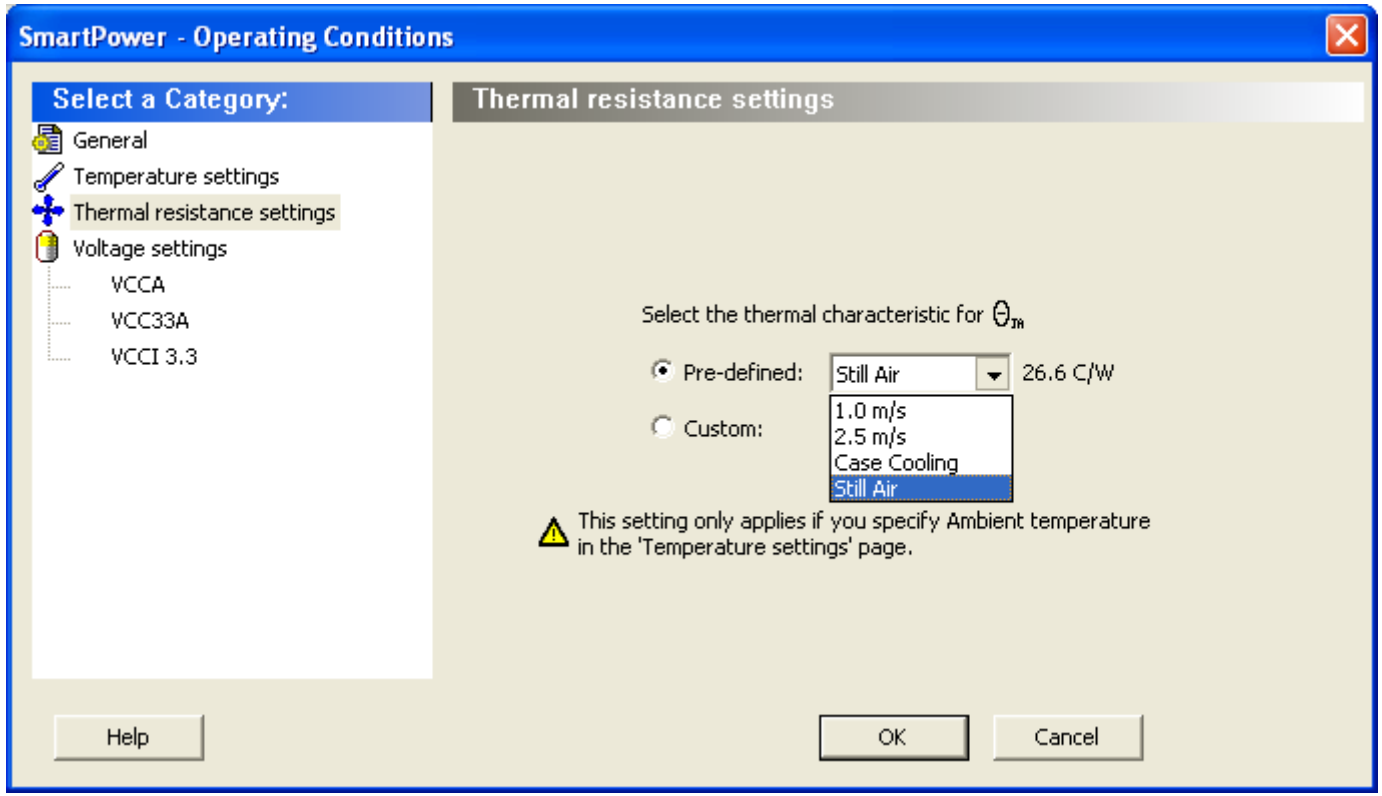




SmartPower Junction Temperature

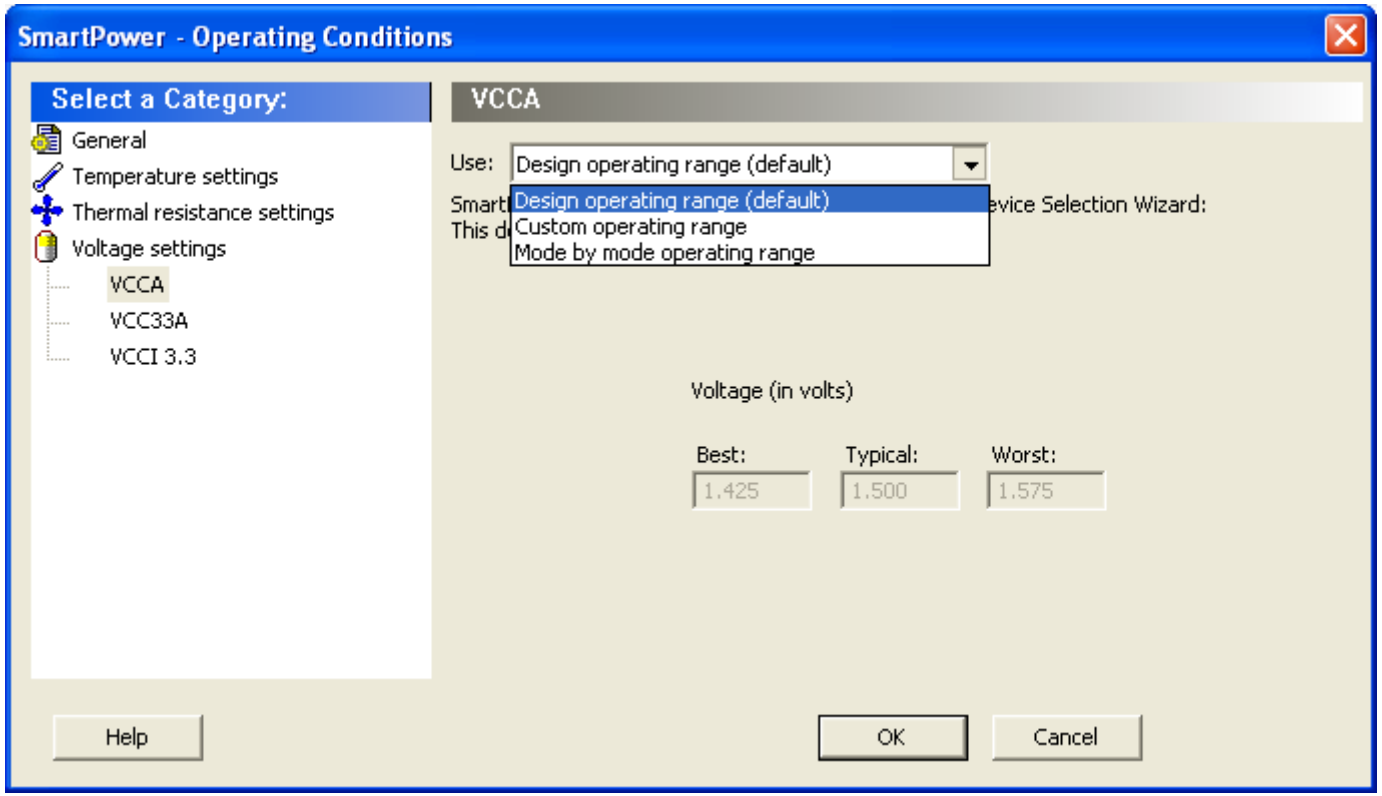
Operating Conditions – Thermal Resistance Settings

- Define Cooling Scenarios
  - Pre-defined or custom



SmartPower Custom Voltage Settings

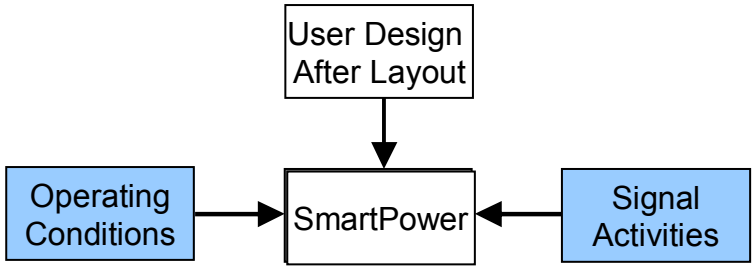
- Define Custom Voltage Settings
  - Operating range, custom operating range or mode by mode



SmartPower Operating Modes

Using SmartPower Modes and Scenarios

- What is a Mode in SmartPower?
  - It is a way to save a set of parameters defining the power of a design
  - You can create modes to record:
    - Operating Conditions: Specific Temperature or Voltage Settings
    - Activities: Different simulations, or turning one clock Off…

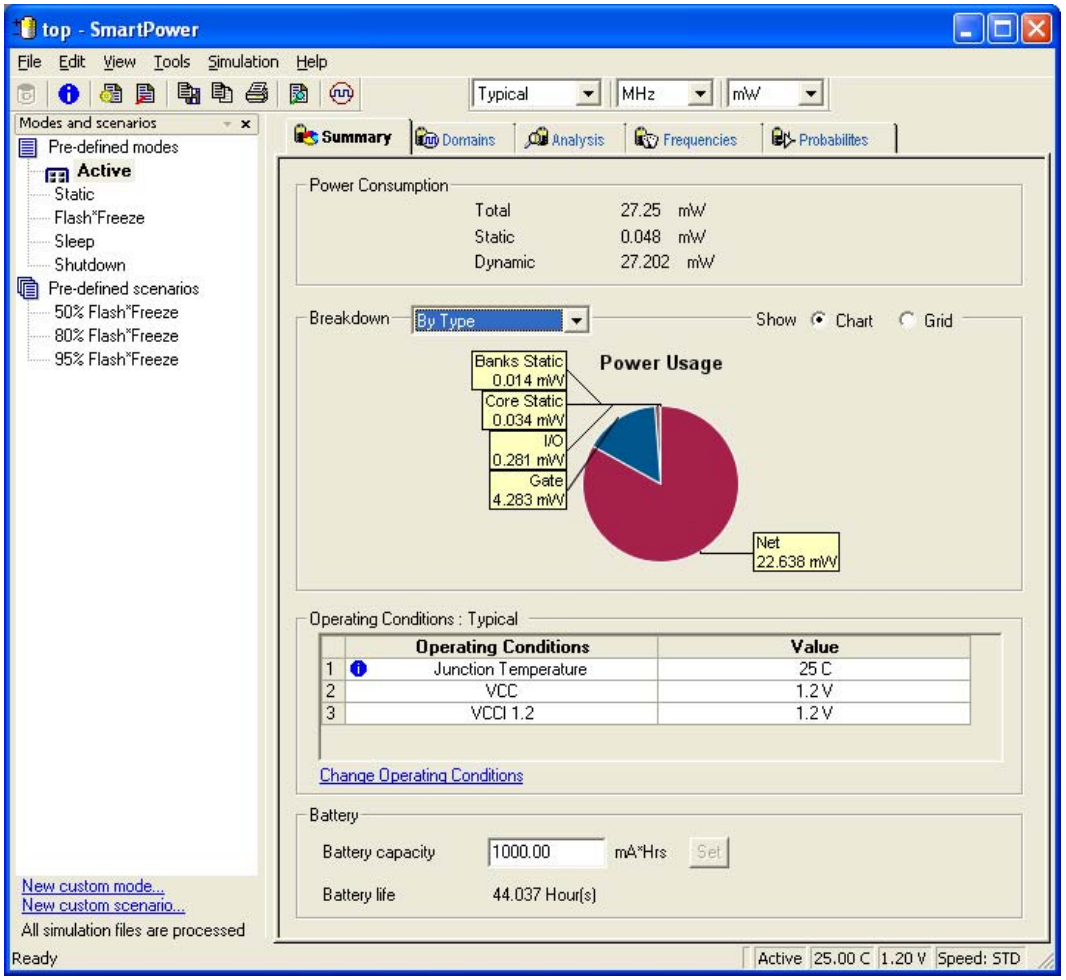


  - We also offer predefined Modes:
    - Flash*Freeze
    - Sleep
    - Stand-by (Fusion Only)

Using SmartPower Modes and Scenarios

- What is a Scenario in SmartPower?
  - It is a way to combine different modes to estimate the power over the life of a design, and not just in one mode
  - You can create Scenarios:
    - If you want to mix different simulations
    - If Operating Conditions are changing over the life of a Product
      - Temperature profile for Automotive
      - Voltage variations due to battery decay
    - If you want to mix different modes (Flash*Freeze and Active)
  - We also offer predefined Scenario:
    - 95% Flash*Freeze – 5% Active
    - 80% Flash*Freeze – 20% Active
    - 50% Flash*Freeze – 50% Active

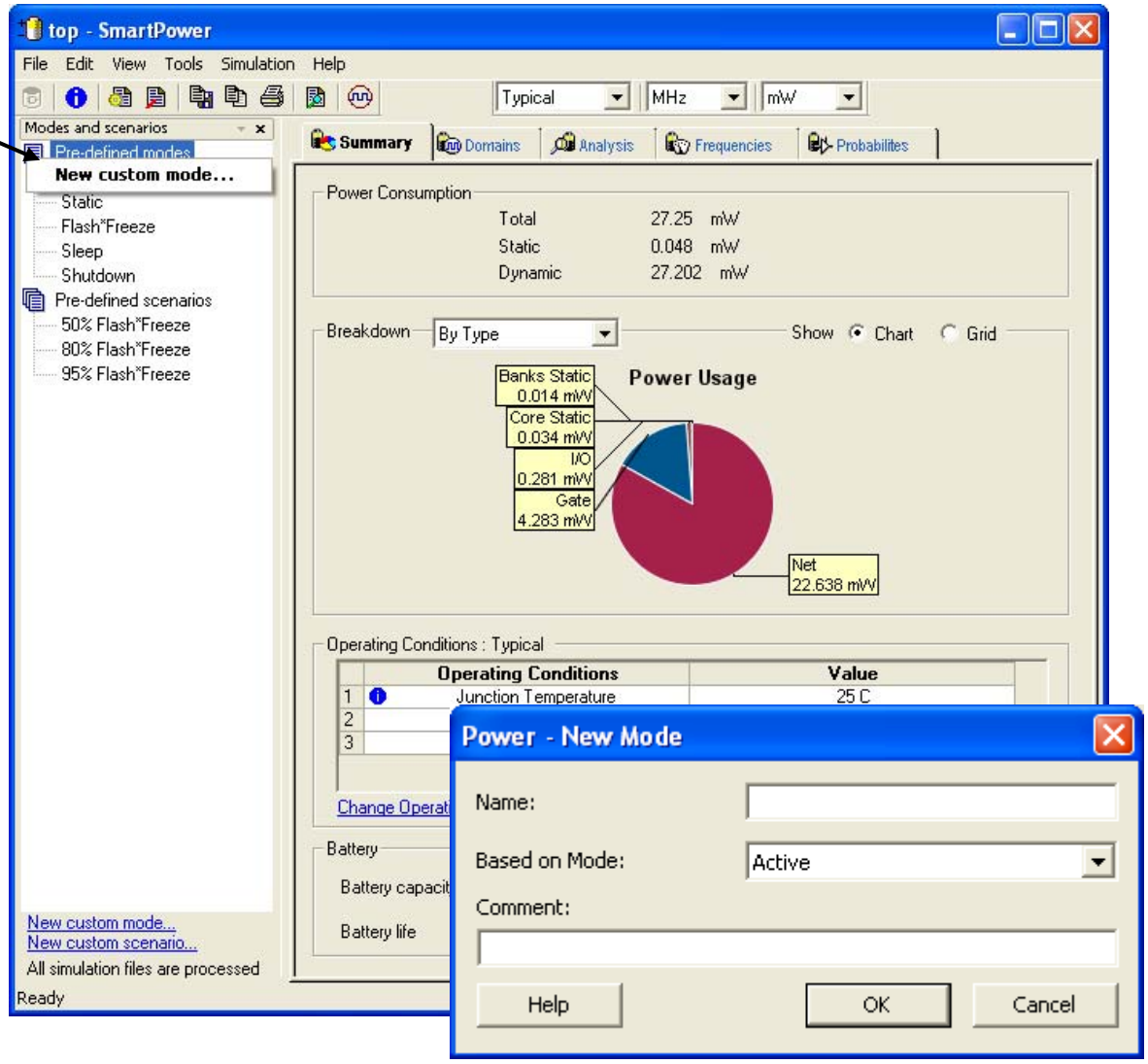
Using SmartPower Modes and Scenarios



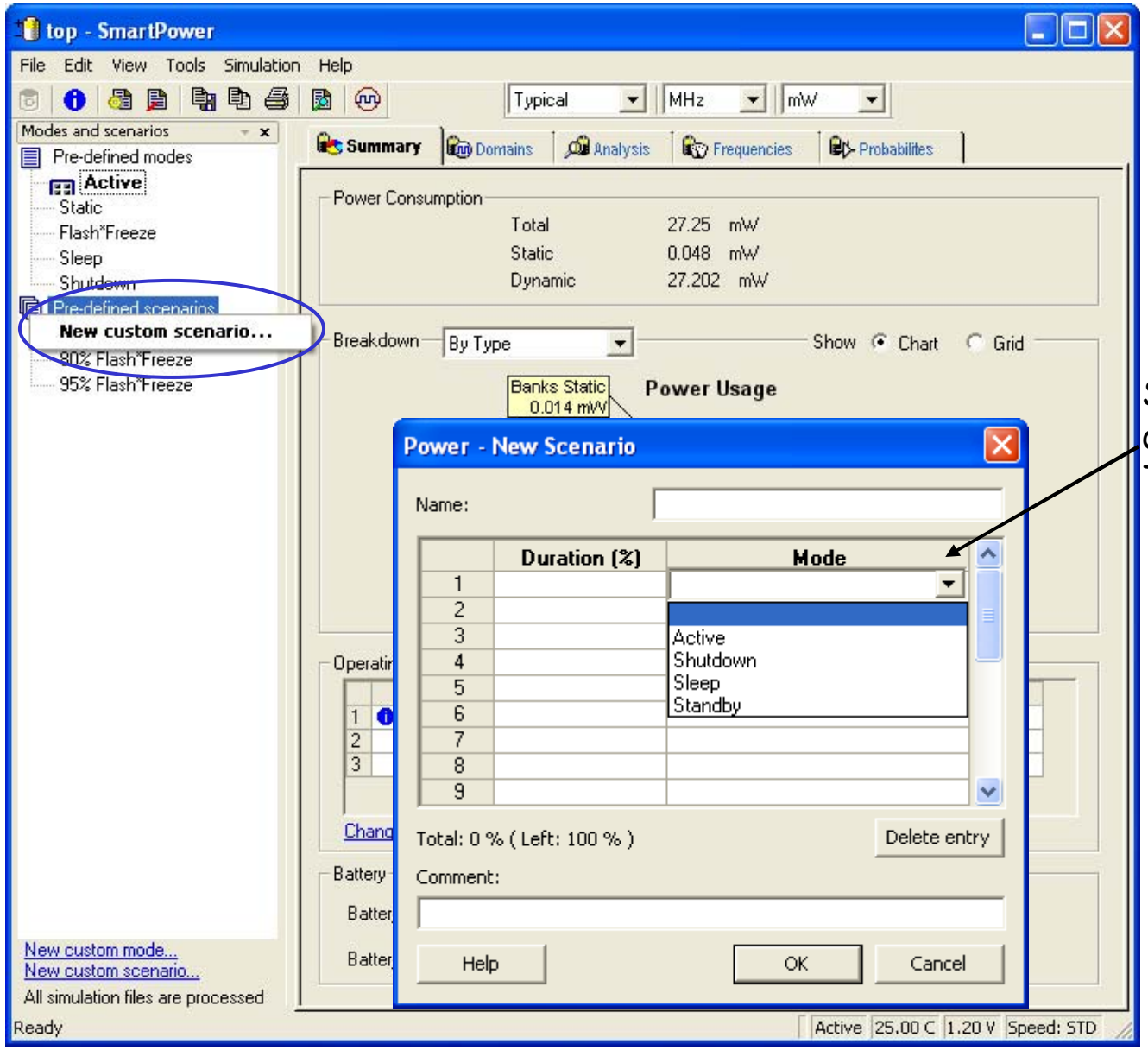
- Supports Analysis with all modes of use for the FPGA
  - Active
  - Shutdown
  - Flash*Freeze (IGLOO)
  - Sleep
  - Standby
- Summary tab supports analysis of any mode
- Use Combination of Modes to Create a Power "Scenario"

SmartPower Custom Modes

Click to create custom mode



Creating a Power Scenario



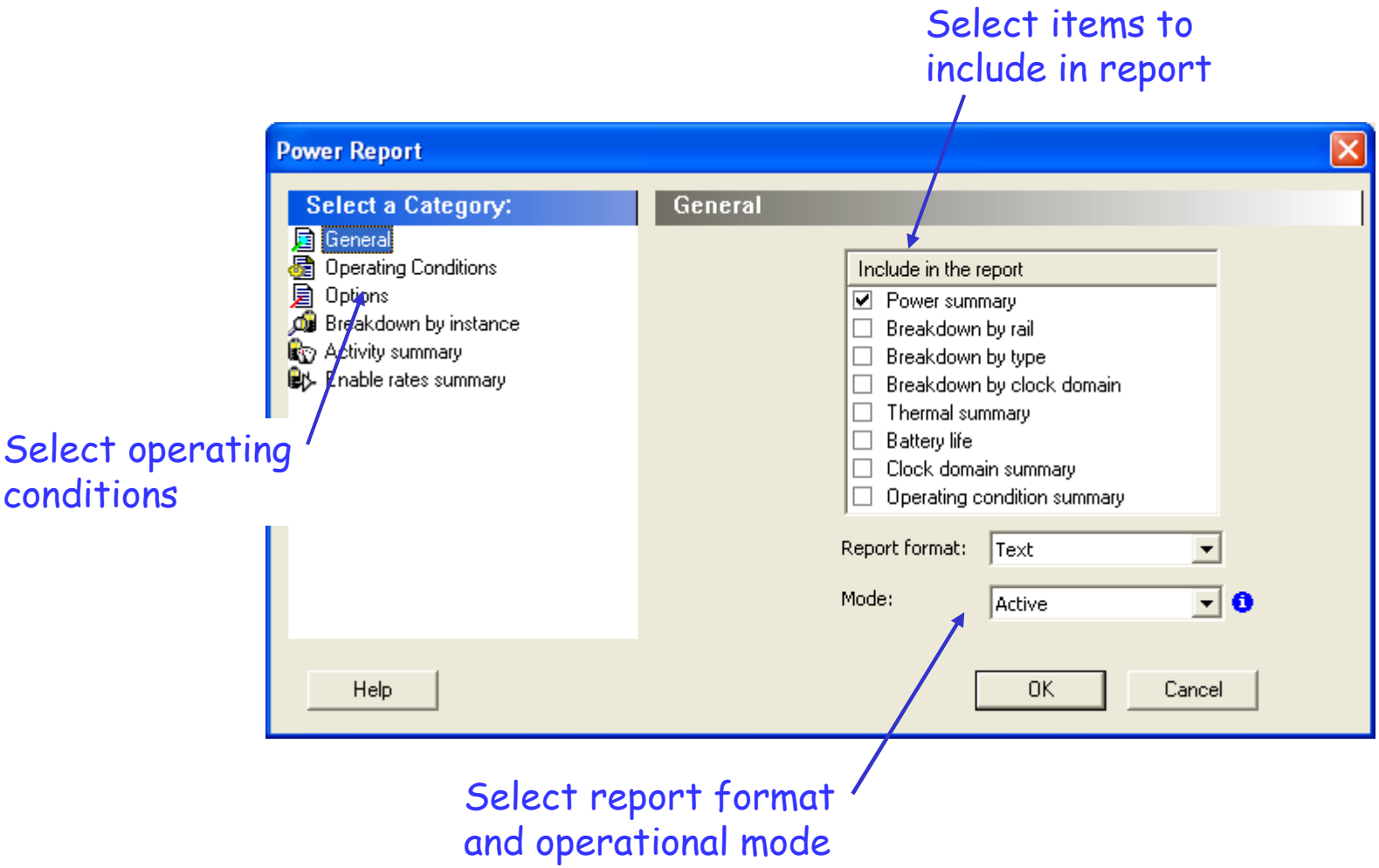
Specify mode and duration
Total must = 100%

SmartPower Reports

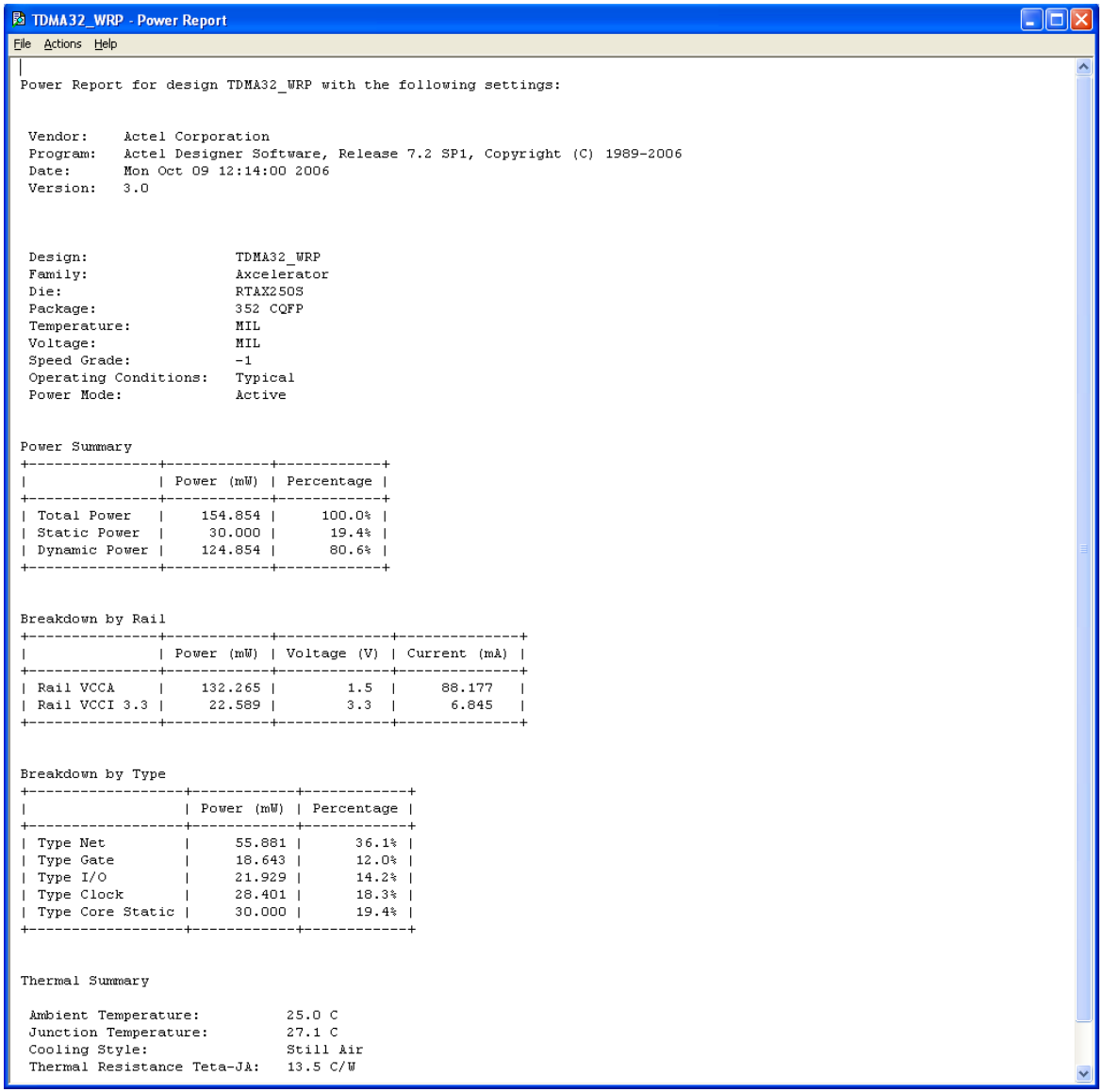
SmartPower Power Reports

- SmartPower can Generate the Following Reports:
  - Power Report
  - Scenario Power Report
  - Cycle Accurate Power Report
  - Activity and Hazards Power Report
- Text or CSV (Comma Separated Value) Format
- Customize Power Report
  - Items to Include in Report
  - Breakdown by Instance
  - Limit Number of Included Instances
  - Specify Minimum Power Level for Inclusion in Report
- Options Button Invokes Preferences Window

SmartPower Power Report Options



Power Report



TDMA32_WRP - Power Report

File Actions Help

```
Power Report for design TDMA32_WRP with the following settings:


 Vendor:    Actel Corporation
 Program:   Actel Designer Software, Release 7.2 SP1, Copyright (C) 1989-2006
 Date:      Mon Oct 09 12:14:00 2006
 Version:   3.0



 Design:                TDMA32_WRP
 Family:                Axcelerator
 Die:                   RTAX250S
 Package:               352 CQFP
 Temperature:           MIL
 Voltage:               MIL
 Speed Grade:           -1
 Operating Conditions:  Typical
 Power Mode:            Active


Power Summary
+---------------+-----------+------------+
|               | Power (mW) | Percentage |
+---------------+-----------+------------+
| Total Power   |    154.854 |     100.0% |
| Static Power  |     30.000 |      19.4% |
| Dynamic Power |    124.854 |      80.6% |
+---------------+-----------+------------+


Breakdown by Rail
+---------------+-----------+------------+-------------+
|               | Power (mW) | Voltage (V) | Current (mA) |
+---------------+-----------+------------+-------------+
| Rail VCCA     |    132.265 |        1.5  |     88.177   |
| Rail VCCI 3.3 |     22.589 |        3.3  |      6.845   |
+---------------+-----------+------------+-------------+


Breakdown by Type
+-----------------+-----------+------------+
|                 | Power (mW) | Percentage |
+-----------------+-----------+------------+
| Type Net        |     55.881 |      36.1% |
| Type Gate       |     18.643 |      12.0% |
| Type I/O        |     21.929 |      14.2% |
| Type Clock      |     28.401 |      18.3% |
| Type Core Static |     30.000 |      19.4% |
+-----------------+-----------+------------+


Thermal Summary

 Ambient Temperature:          25.0 C
 Junction Temperature:         27.1 C
 Cooling Style:                Still Air
 Thermal Resistance Teta-JA:   13.5 C/W
```

Power Report
Breakdown by Instance

TDMA32_WRP - Power Report

File Actions Help

Power Report for design TDMA32_WRP with the following settings:

Vendor: Actel Corporation
Program: Actel Designer Software, Release 7.2 SP1, Copyright (C) 1989-2006
Date: Mon Oct 09 12:19:20 2006
Version: 3.0

Design: TDMA32_WRP
Family: Axcelerator
Die: RTAX250S
Package: 352 CQFP
Temperature: MIL
Voltage: MIL
Speed Grade: -1
Operating Conditions: Typical
Power Mode: Active

Power Summary

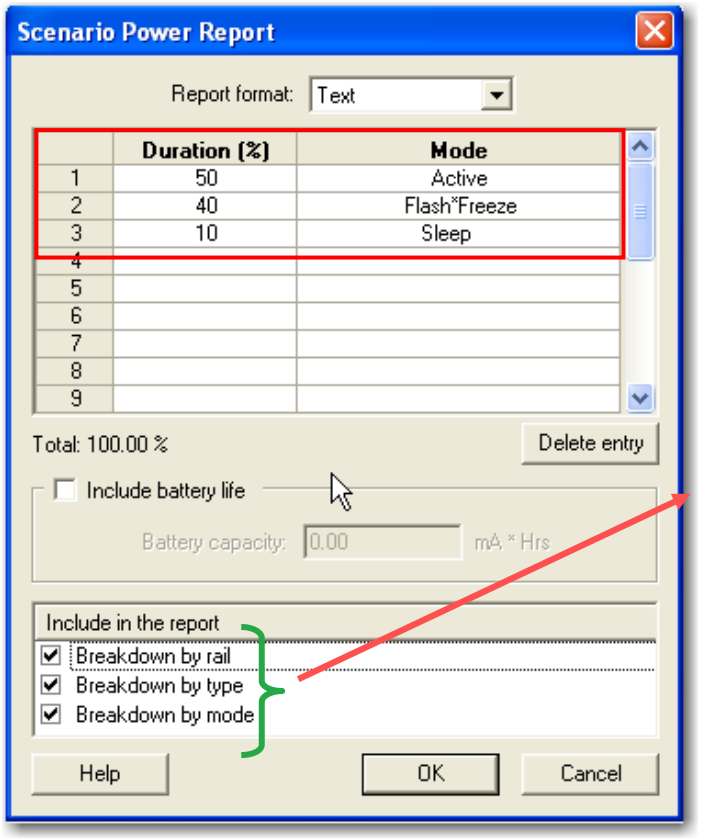
	Power (mW)	Percentage
Total Power	154.854	100.0%
Static Power	30.000	19.4%
Dynamic Power	124.854	80.6%

Breakdown by Instance

	Type	Power (mW)	Macro
PCI_CLK	Net	25.272	
MAKE_CLKS_MAKE_H_CLK	I/O	3.129	ADLIB:HCLKBUF
DP_DONE_pad	I/O	0.721	ADLIB:OUTBUF
WR_BE_NOW_pad[0]	I/O	0.721	ADLIB:OUTBUF
WR_BE_NOW_pad[1]	I/O	0.721	ADLIB:OUTBUF
WR_BE_NOW_pad[2]	I/O	0.721	ADLIB:OUTBUF
WR_BE_NOW_pad[3]	I/O	0.721	ADLIB:OUTBUF
BAR0_MEM_CYC_pad	I/O	0.719	ADLIB:OUTBUF
BAR1_CYC_pad	I/O	0.719	ADLIB:OUTBUF
BE_GNT_pad	I/O	0.719	ADLIB:OUTBUF
CONFIG_CYC_pad	I/O	0.719	ADLIB:OUTBUF
DMA_GNT_pad	I/O	0.719	ADLIB:OUTBUF
DP_START_pad	I/O	0.719	ADLIB:OUTBUF

Scenario Power Report

- Create a Power Report Based on a Power Profile
  - Report shows where/how much power is consumed



Power Report

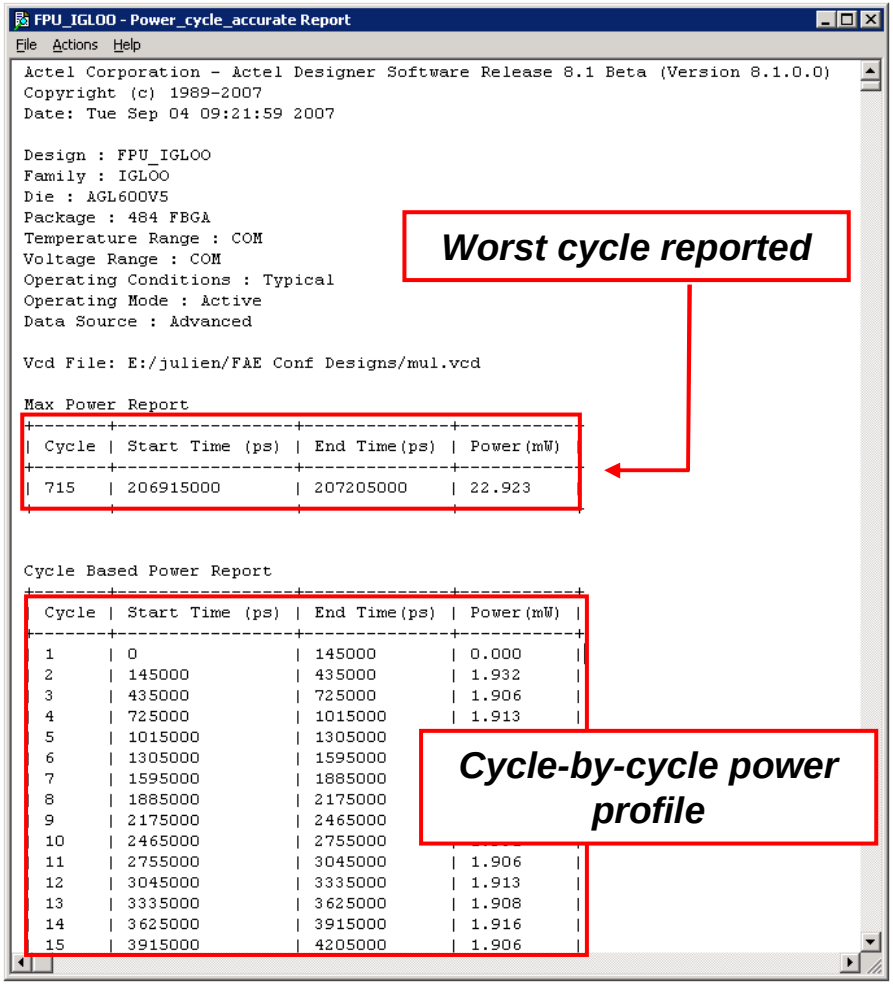
```
Scenario Average Power Summary
+---------------+------------+------------+
|               | Power (uW) | Percentage |
+---------------+------------+------------+
| Total Power   |   9023.903 |      100.0% |
| Static Power  |     25.260 |        0.3% |
| Dynamic Power |   8998.643 |       99.7% |
+---------------+------------+------------+

Breakdown by mode
+-------------------+-------------------------+--------------------+-------------------------+
| Mode              | Power in the Mode (uW) | Duration           | Power*Duration Weight |
+-------------------+-------------------------+--------------------+-------------------------+
| Mode Active       |              18032.986 |              50.0% |                99.9%    |
| Mode Flash*Freeze |                 16.800 |              40.0% |                 0.1%    |
| Mode Sleep        |                  6.900 |              10.0% |                 0.0%    |
+-------------------+-------------------------+--------------------+-------------------------+


Breakdown By Rail
+--------------+------------+-------------+--------------+
| Rail         | Power (uW) | Voltage (V) | Current (uA) |
+--------------+------------+-------------+--------------+
| Rail VCCA    |   4720.684 |       1.200 |   3933.903   |
| Rail VCCI 1.5 |   4303.219 |       1.500 |   2868.813   |
+--------------+------------+-------------+--------------+


Breakdown By Type
+-------------------+------------+------------+
| Type              | Power (uW) | Percentage |
+-------------------+------------+------------+
| Type Banks Static |      4.140 |       0.0% |
| Type Core Static  |     21.120 |       0.2% |
| Type Gate         |    920.786 |      10.2% |
| Type I/O          |   4299.079 |      47.6% |
| Type Net          |   3778.778 |      41.9% |
+-------------------+------------+------------+
```

Cycle Accurate Power Report
Overview

- Feature Definition
  - Offers Peak Power Analysis from a Simulation (VCD) file
  - Allows Cycle by Cycle analysis of the Power over the simulation
- Why Cycle Accurate Power Report?
  - For critical applications, a cycle accurate report allows designing for worst-case scenario
  - A cycle accurate report allows simulation and analysis of power saving scenarios (example: clock with variable frequency)

Cycle-Accurate Power Reporting

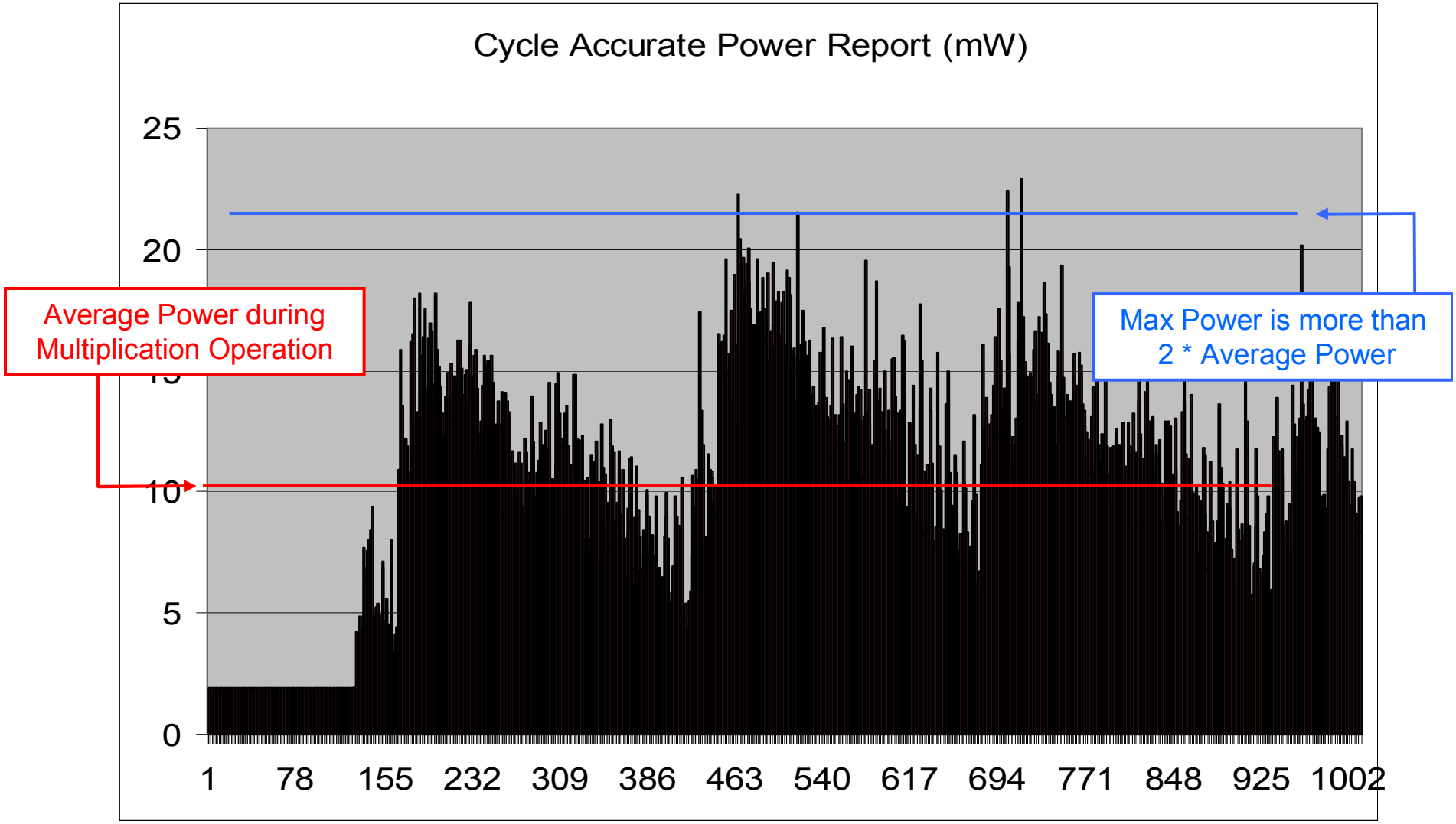
- Accurate report of peak power per clock cycle
  - Based on testbench
- Cycle-by-cycle analysis of the power over the simulation
- Allows designing for worst-case scenario



Select Options

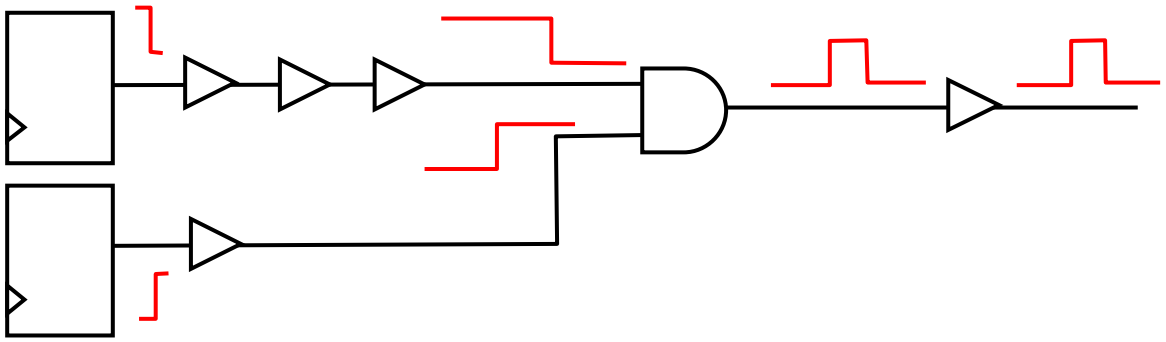
-Specify sampling period
-Report format and results

Cycle Accurate Power Report



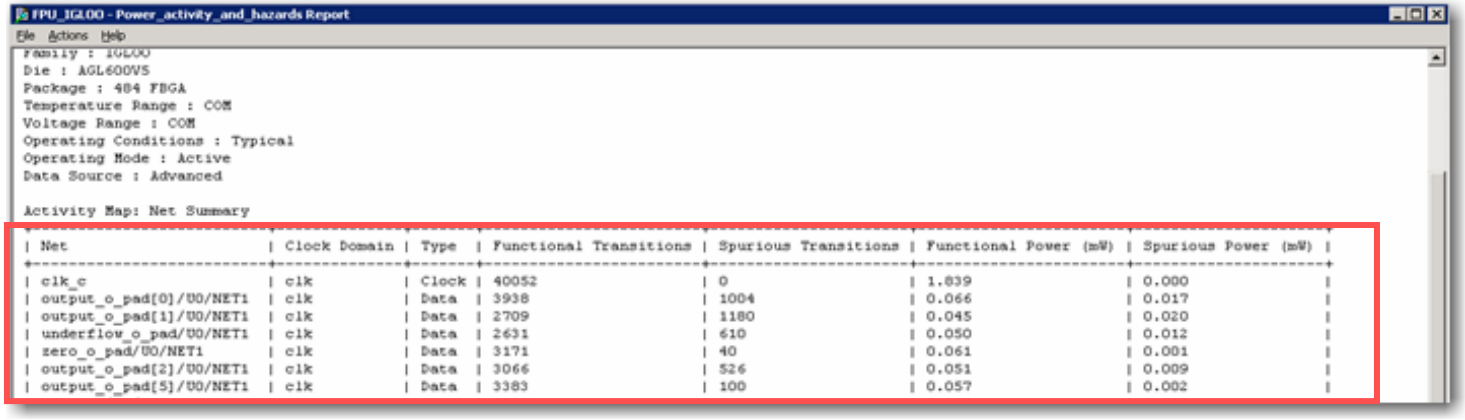
Activity and Hazards Power Report
Overview

- Feature Definition
  - Allows the analysis of power dissipated by Hazards from a Simulation (VCD) file.
  - Definition of Hazard: **Due to the delay mismatch along different fan-in paths of each internal gate, the output of a gate may have unexpected transitions before settling to the correct logic level. These transitions are called hazards or spurious transitions**



Hazard Analysis

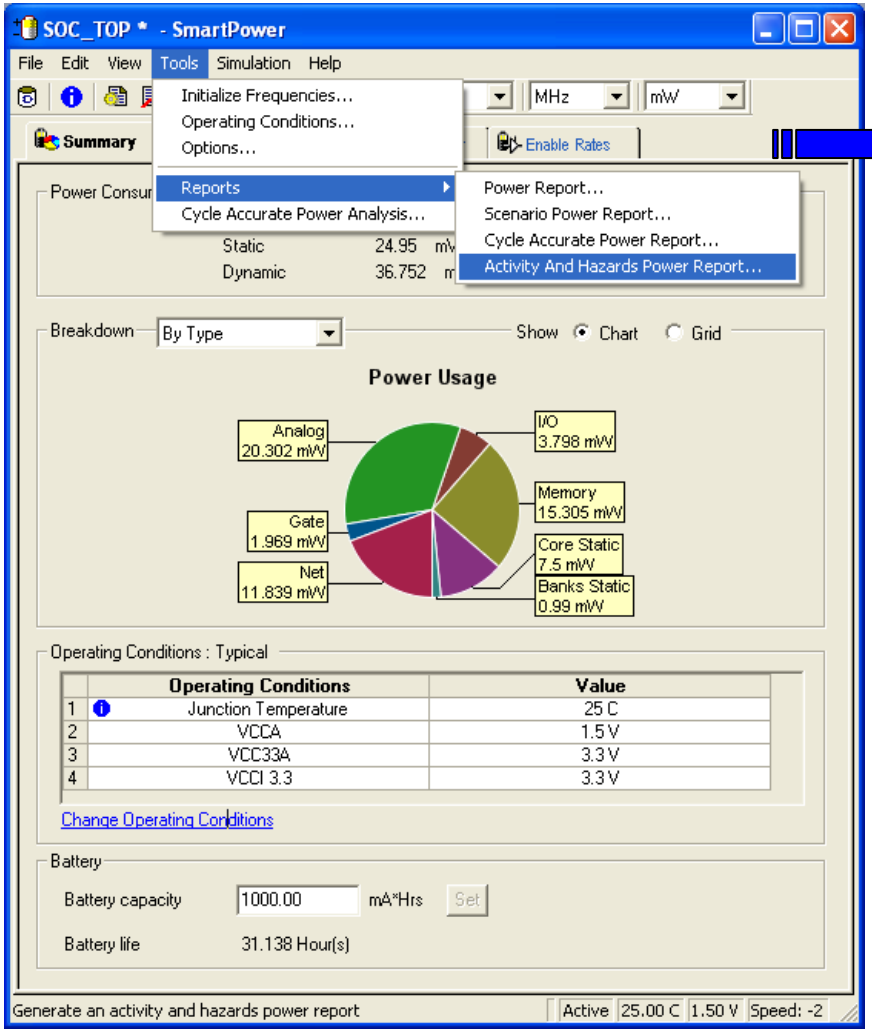
- Hazards can account for as much as 20% of total global power*
- Power dissipation caused by hazards can be much as 70% of total power

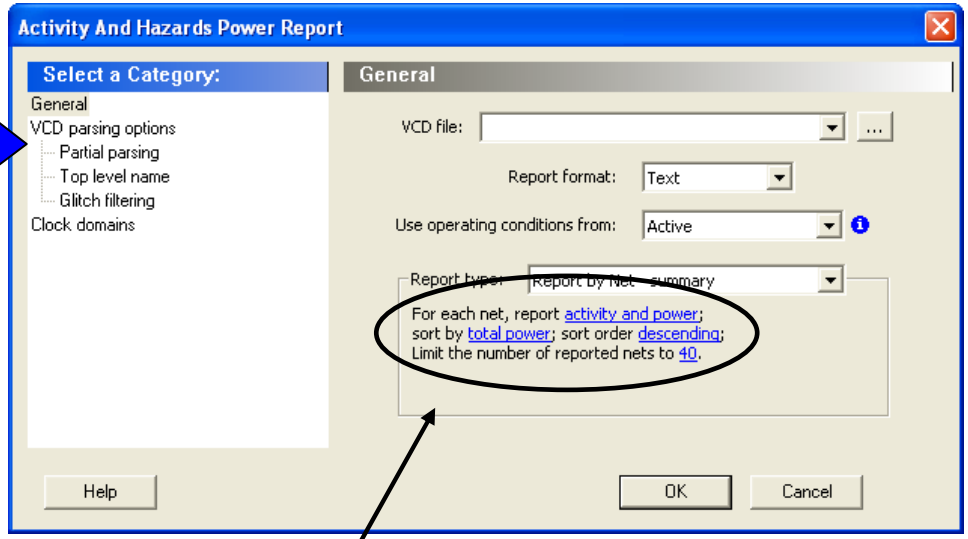


- Report Highlights
  - # of functional transitions, # of hazards and their associated power

*According to Favalli and Benini, *International Symposium on Low Power Electronics and Design*, 1995

Activity and Hazards Power Report
Options





Query Options:
Report: Activity, Activity and Power, Power
Sort by: Functional power, functional transitions, spurious power, spurious transitions, total power
Sort order: ascending, descending
Number of reported nets: enter a value

Activity and Hazards Power Report
Example 1

- **For each net, report activity; sort by spurious transitions; sort order descending; Limit the number of reported nets to 50.**

FPU_IGLOO - Power_activty_and_hazards_1.rpt Report

File Actions Help

```
Family : IGLOO
Die : AGL600V5
Package : 484 FBGA
Temperature Range : COM
Voltage Range : COM
Operating Conditions : Typical
Operating Mode : Active
Data Source : Advanced

Activity Map: Net Summary
```

Net	Clock Domain	Type	Functional Transitions	Spurious Transitions
UUT/u5/prod1_2[34]	clk	Data	4734	32544
UUT/u5/prod1_2[37]	clk	Data	4407	31992
UUT/u5/prod1_2[29]	clk	Data	4276	31584
UUT/u5/prod1_2[36]	clk	Data	4912	29008
UUT/u5/prod1_2[30]	clk	Data	4680	28788
UUT/u5/prod1_2[39]	clk	Data	3952	28754
UUT/u5/prod1_2[40]	clk	Data	3692	28506
UUT/u5/prod1_2[31]	clk	Data	4506	28116
UUT/u5/prod1_2[32]	clk	Data	4237	27888
UUT/u5/prod1_2[33]	clk	Data	4470	27818
UUT/u5/prod1_2[35]	clk	Data	4420	27126
UUT/u5/prod1_2[41]	clk	Data	3284	27048
UUT/u5/prod1_2[38]	clk	Data	4863	26676
UUT/u5/prod1_2[42]	clk	Data	3660	26432
UUT/u5/prod1_2[28]	clk	Data	4917	25196
UUT/u5/prod1_2[20]	clk	Data	4498	24934
UUT/u5/prod1_2[27]	clk	Data	4497	24868
UUT/u5/prod1_2[26]	clk	Data	4735	24330
UUT/u5/prod1_2[44]	clk	Data	3707	24238
UUT/u5/prod1_2[22]	clk	Data	4227	24028
UUT/u5/prod1_2[43]	clk	Data	2854	23504
UUT/u4/un1_exp_in_4[4]	clk	Data	4521	23236
UUT/u5/prod1_2[23]	clk	Data	4186	23058
UUT/u5/prod1_2[25]	clk	Data	4510	22840
UUT/u5/prod1_2[21]	clk	Data	4163	21888
UUT/u4/un1_exp_in_4[3]	clk	Data	4613	21856

We analyze the simulation cycle by cycle, and net per net, and we report the number of functional transitions and spurious transitions for each net

Functional Transitions	Spurious Transitions
4734	32544
4407	31992
4276	31584
4912	29008
4680	28788
3952	28754
3692	28506
4506	28116

Activity and Hazards Power Report

Example 2

- **For each net, report activity and power; sort by power; sort order descending; Limit the number of reported nets to 40.**

FPU_IGLOO - Power_activity_and_hazards Report

File Actions Help

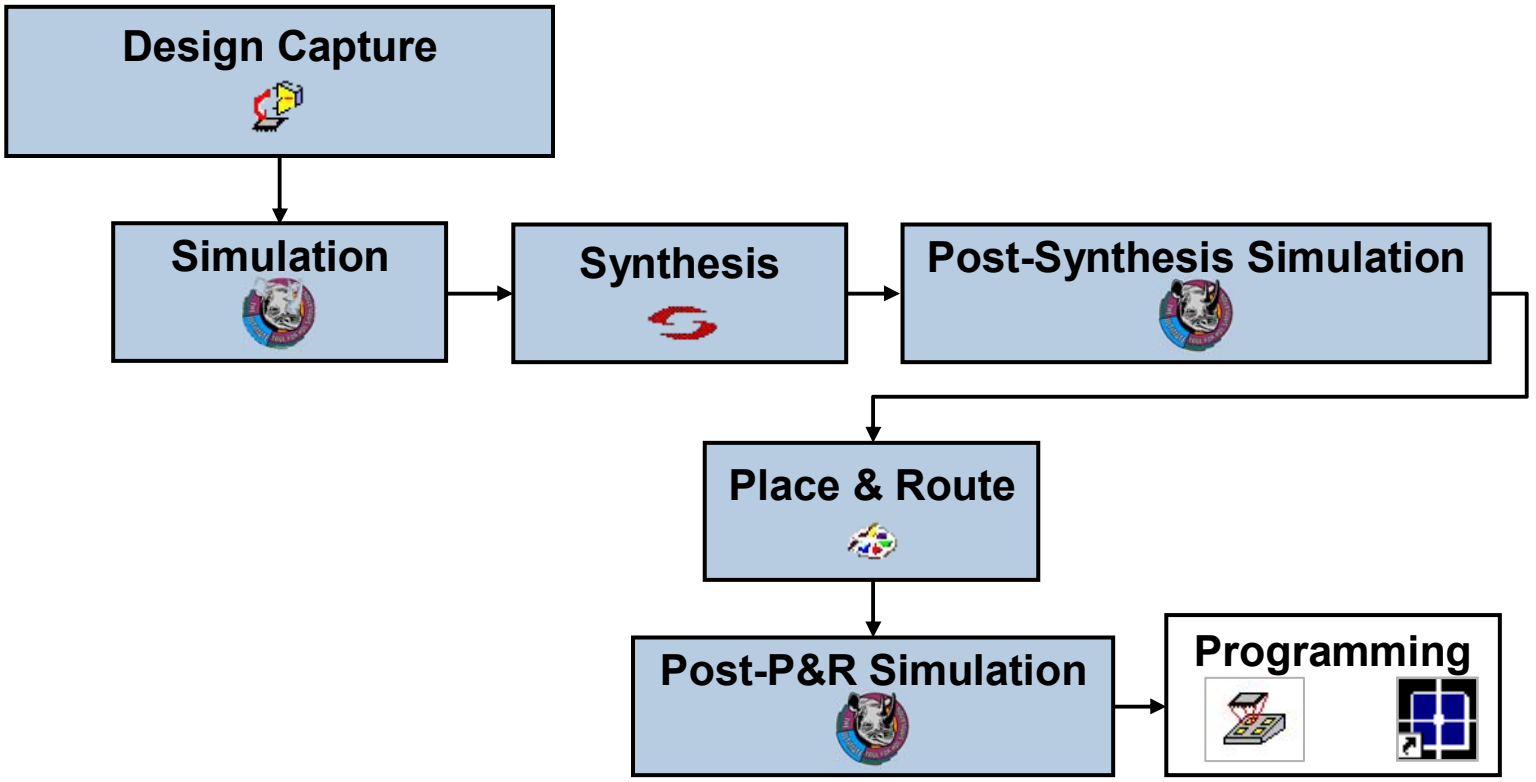
```
Family : IGLOO
Die : AGL600V5
Package : 484 FBGA
Temperature Range : COM
Voltage Range : COM
Operating Conditions : Typical
Operating Mode : Active
Data Source : Advanced

Activity Map: Net Summary
```

Net	Clock Domain	Type	Functional Transitions	Spurious Transitions	Functional Power (mW)	Spurious Power (mW)
clk_c	clk	Clock	40052	0	1.839	0.000
output_o_pad[0]/U0/NET1	clk	Data	3938	1004	0.066	0.017
output_o_pad[1]/U0/NET1	clk	Data	2709	1180	0.045	0.020
underflow_o_pad/U0/NET1	clk	Data	2631	610	0.050	0.012
zero_o_pad/U0/NET1	clk	Data	3171	40	0.061	0.001
output_o_pad[2]/U0/NET1	clk	Data	3066	526	0.051	0.009
output_o_pad[5]/U0/NET1	clk	Data	3383	100	0.057	0.002
clk_pad/U0/NET1	clk	Clock	20026	0	0.057	0.000
output_o_pad[4]/U0/NET1	clk	Data	3166	120	0.053	0.002
output_o_pad[3]/U0/NET1	clk	Data	3118	158	0.052	0.003
output_o_pad[7]/U0/NET1	clk	Data	3078	158	0.052	0.003
output_o_pad[13]/U0/NET1	clk	Data	3059	120	0.051	0.002
output_o_pad[12]/U0/NET1	clk	Data	3057	40	0.051	0.001
output_o_pad[14]/U0/NET1	clk	Data	2896	160	0.048	0.003
output_o_pad[23]/U0/NET1	clk	Data	2785	258	0.047	0.004
output_o_pad[15]/U0/NET1	clk	Data	2976	60	0.050	0.001
output_o_pad[31]/U0/NET1	clk	Data	3032	0	0.051	0.000
output_o_pad[6]/U0/NET1	clk	Data	2915	80	0.049	0.001
output_o_pad[20]/U0/NET1	clk	Data	2962	20	0.050	0.000
output_o_pad[8]/U0/NET1	clk	Data	2882	80	0.048	0.001
output_o_pad[26]/U0/NET1	clk	Data	2686	216	0.045	0.004
output_o_pad[24]/U0/NET1	clk	Data	2526	336	0.042	0.006
output_o_pad[16]/U0/NET1	clk	Data	2676	100	0.045	0.002
output_o_pad[25]/U0/NET1	clk	Data	2597	140	0.043	0.002
output_o_pad[10]/U0/NET1	clk	Data	2629	80	0.044	0.001
output_o_pad[9]/U0/NET1	clk	Data	2618	60	0.044	0.001
output_o_pad[28]/U0/NET1	clk	Data	2513	158	0.042	0.003
output_o_pad[17]/U0/NET1	clk	Data	2556	80	0.043	0.001
output_o_pad[11]/U0/NET1	clk	Data	2475	98	0.041	0.002
output_o_pad[27]/U0/NET1	clk	Data	2417	80	0.040	0.001
output_o_pad[22]/U0/NET1	clk	Data	2428	40	0.041	0.001
output_o_pad[21]/U0/NET1	clk	Data	2142	60	0.036	0.001
output_o_pad[18]/U0/NET1	clk	Data	2084	40	0.035	0.001
output_o_pad[19]/U0/NET1	clk	Data	1994	118	0.033	0.002
output_o_pad[29]/U0/NET1	clk	Data	1846	176	0.031	0.003
UUT/u4/f2i_shft[3]	clk	Data	3625	5492	0.013	0.019
UUT/u4/f2i_shft[2]	clk	Data	4504	1214	0.024	0.006
output_o_pad[30]/U0/NET1	clk	Data	1704	80	0.029	0.001
UUT/u4/shift_left[1]	clk	Data	3996	2572	0.017	0.011
ine_o_pad/U0/NET1	clk	Data	422	1010	0.007	0.017

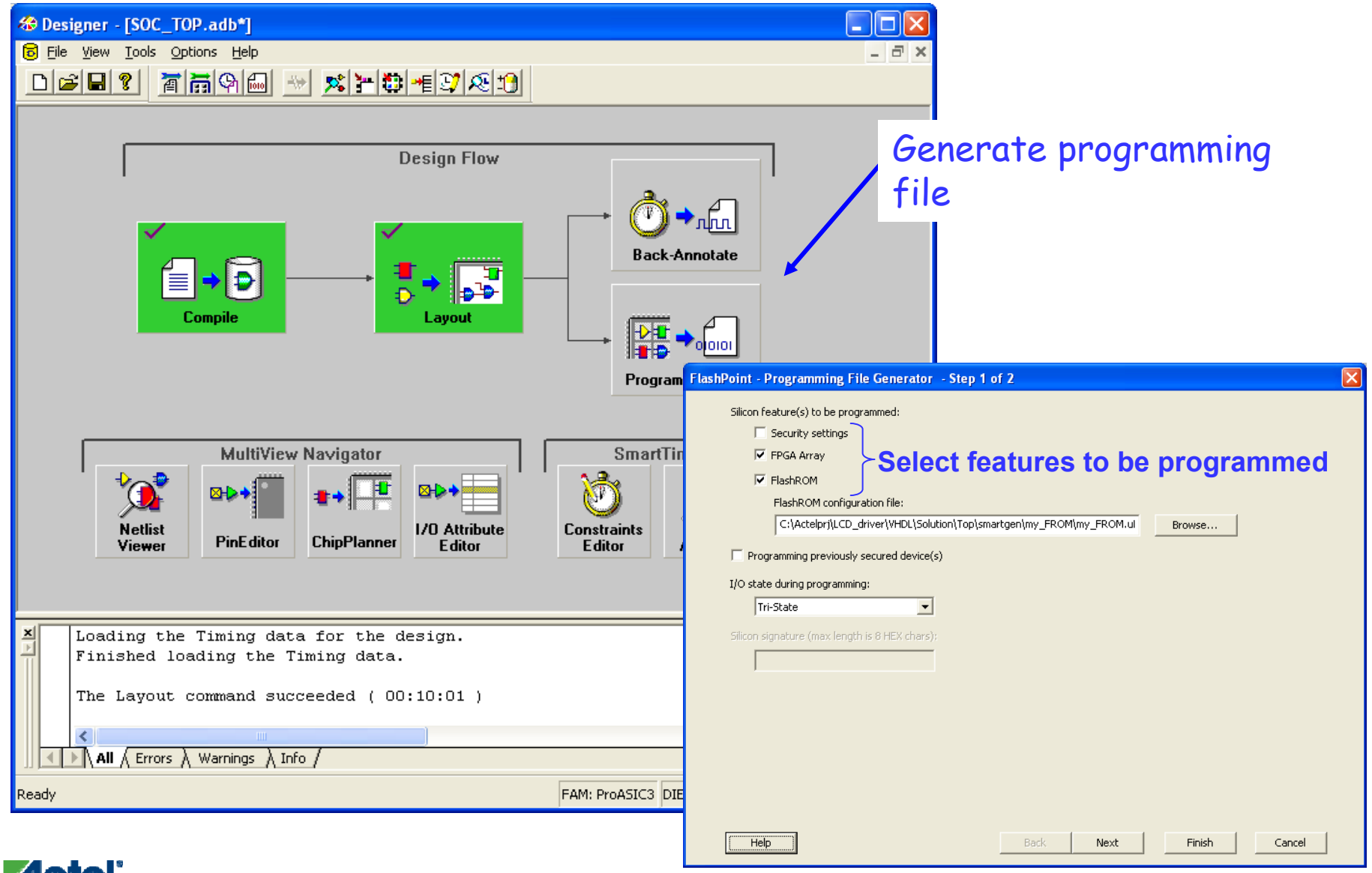
We are also able to report functional power and spurious power for each each net

s	Functional Power (mW)	Spurious Power (mW)
	1.839	0.000
	0.066	0.017
	0.045	0.020
	0.050	0.012
	0.061	0.001
	0.051	0.009
	0.057	0.002
	0.057	0.000
	0.053	0.002
	0.052	0.003

Programming & Debugging



Generating Programming File



Hands-on Labs 7, 8 and 9

- Complete Labs 7, 8 and 9 in the Lab Guide
- Lab 7
  - Layout Design
  - Generate Global Usage Report
  - Create backannotated timing file
  - Save your Work
- Lab 8
  - Run Post-layout timing simulation
  - Create VCD file
- Lab 9
  - Measure Power Consumption with SmartPower
  - Generate the programming file

Programming

Sculptor 3 Overview

- PC-based Parallel-port, Single Device Programmer
- Replaces Silicon Sculptor II
- Compatible with all Sculptor II Adapter Modules
- Supports all Devices and Families:
  - IGLOO/e, Fusion, ProASIC3/E, ProASIC PLUS
  - ARM-enabled Flash FPGAs
  - All anti-fuse products including RTAX4000S

Silicon Sculptor Software

- Available from Actel Website
  - http://www.actel.com/custsup/updates/silisculpt/
- Requirements (Windows Version)
  - Microsoft Windows 95/98, Win NT or Win 2000
- Requirements (DOS Version)
  - 286 with 4MB RAM, Approx. 6MB Hard Drive Space
  - DOS-driven Program - Memory Managers Not Required
  - DOS Shell from Windows 95/98 - OK
  - Does Not Work with Windows NT

Flash Programming

- Flash FPGA Devices Can Be Programmed Multiple Ways
  - Off-board Programming with Silicon Sculptor 3
  - In-System Programming (ISP) using JTAG Interface with:
    - Silicon Sculptor 3 (ProASIC, ProASIC^{PLUS}, ProASIC3/E, Fusion, IGLOO/e)
    - Flash Pro (ProASIC or ProASIC^{PLUS})
    - Flash Pro Lite (ProASIC^{PLUS})
    - Flash Pro 3 (ProASIC3/E, Fusion, IGLOO/e)
  - Programming via Microprocessor Interface
    - (ProASIC^{PLUS}, ProASIC3/E, Fusion, IGLOO/e)

FlashPro Programmer

ProASIC, ProASIC[PLUS]

- Small Form Factor - 24 in^3
- Low Cost
- Hardware Features
  - Small 26-pin Header
    - Samtec FTSH-113-01-L-D-K
  - 20” Ribbon Cable
  - ECP Parallel Port
- Software Features
  - Win 95/98/NT/00 O/S
  - STAPL Support
  - Daisy Chain Capability
  - Log File Generation
  - Self-test Option

2.5/3.3V	1	2	VDDP
2.5V/3.3V	3	4	VDDP
2.5V/3.3V	5	6	VPP
GND	7	8	VPN
GND	9	10	GND
GND	11	12	TCK
NC	13	14	TDI
NC	15	16	TDO
GND	17	18	TMS
GND	19	20	RCK
TRSTB	21	22	TRSTB
2.5V	23	24	VDDL/VDD
2.5V	25	26	VDDLVDD

FlashPro Lite Programmer

ProASIC^{PLUS}

- Low Cost
- Ultra-small Form Factor
- Hardware Features
  - Draws Power from Target Board
  - Connects to Parallel Port
  - Supports In-system Programming
    - Samtec 26-pin Header
- Software Features
  - Supports Windows 98, NT, 2000, and XP Operating Systems
  - STAPL Support
  - Free Software Updates

FlashPro 3

ProASIC3, ProASIC3L, Fusion and IGLOO

- USB 2.0 High-speed Interface
  - 10-pin JTAG ISP
  - Altera-compatible Interface
- Programs ProASIC3 Devices in Less than 2 Minutes
- Powered by USB Connection
  - Parallel Programming Requires Powered USB Hub
- Variable TCK (up to 24 MHz)
  - Recommend <= 20MHz for PA3/E
- Optional Transition Board provides Adapter Cables for 26- and 10-pin SAMTEC

TCK	1	2	GND
TDO	3	4	NC
TMS	5	6	V_{JTAG}
V_{PUMP}	7	8	TRST
TDI	9	10	GND

Programmer Summary

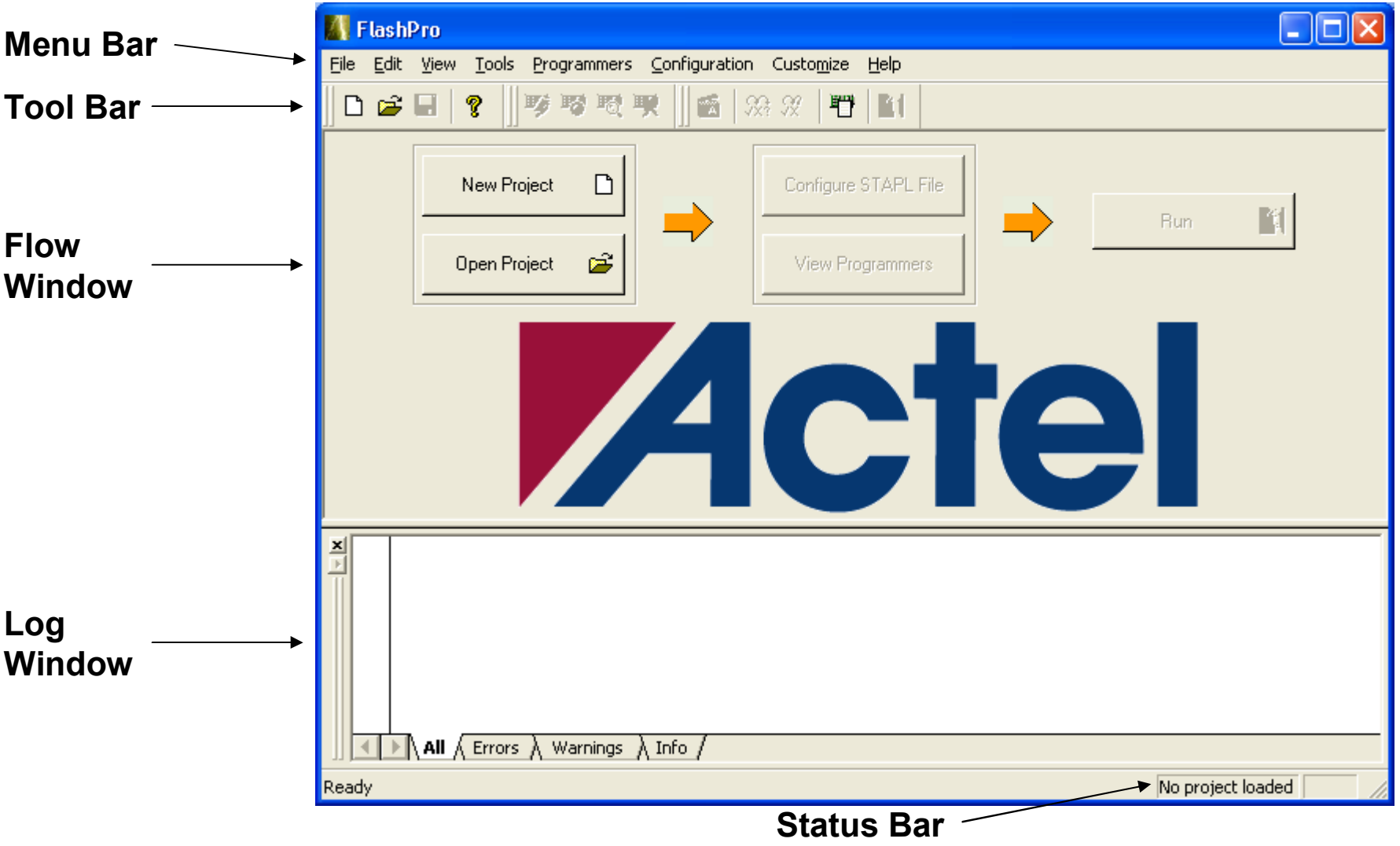
Programmer	Device Support	Availability
Silicon Scupltor 3	All antifuse FPGAs ProASIC3/E, ProASIC3L, Fusion, IGLOO/e ProASIC and ProASIC^{PLUS}	Available
FlashPro	ProASIC and ProASIC^{PLUS}	Available
FlashPro Lite	ProASIC^{PLUS}	Available
FlashPro 3	ProASIC3\E, Fusion, IGLOO/e	Available

FlashPro 8.5

FlashPro v8.5 Software Overview

- Works with all FlashPro series programmers
  - FlashPro3, FlashPro Lite and FlashPro
- Supports A500K, APA, ProASIC3/E/L, AFS and AGL with appropriate programmer
  - ProASIC3/E/L, AFS and AGL support via FlashPro3 only
- Allows programming in parallel via USB hubs
  - Huge increase in throughput for small-scale production
  - FlashPro3 and FlashPro can be operated in parallel
- Supports Chain Programming
- Supports Device Serialization

FlashPro User Interface

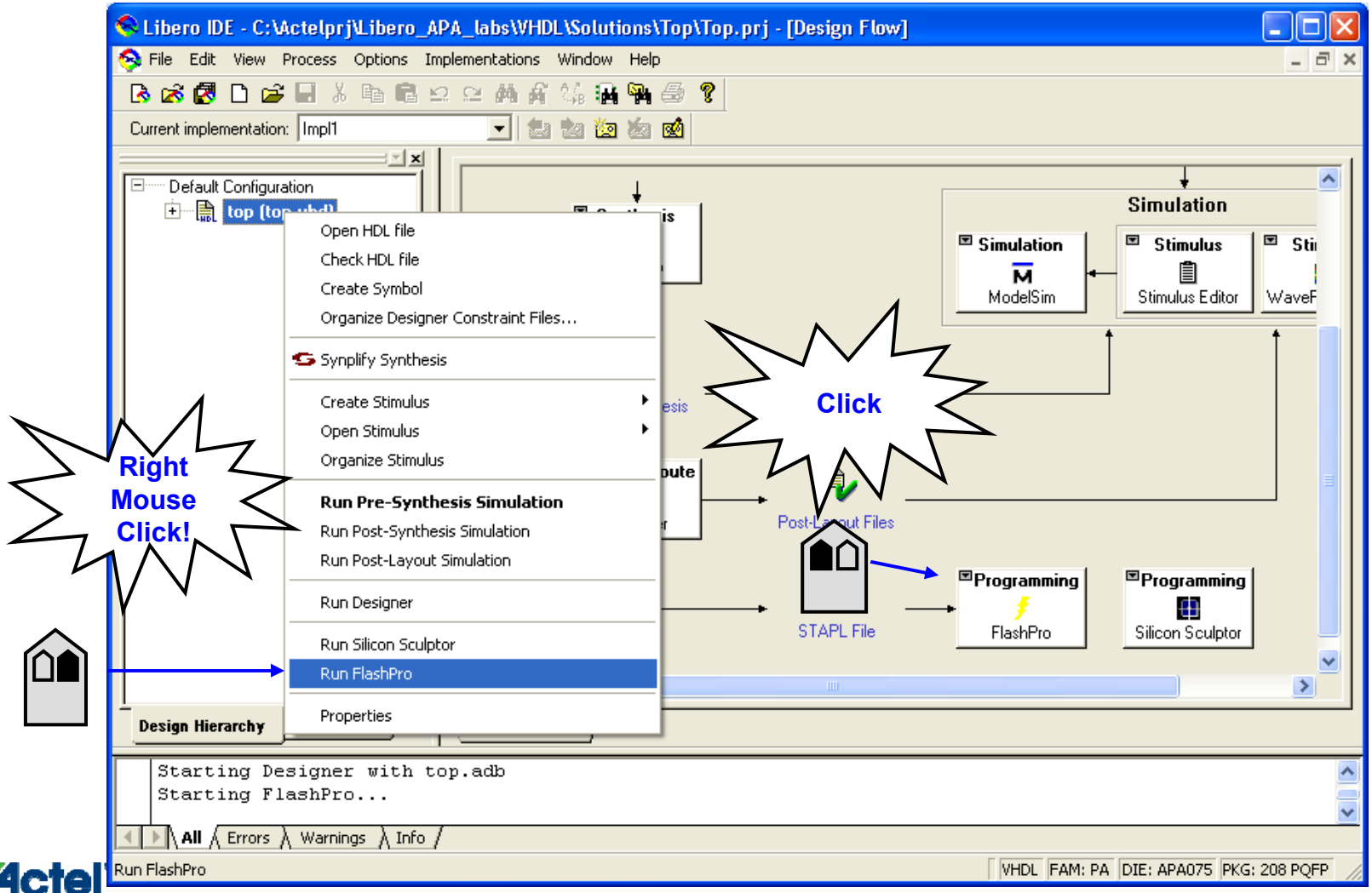


Programming with FlashPro

- Launch FlashPro
- Load STAPL File
- Select Action
- Execute Action

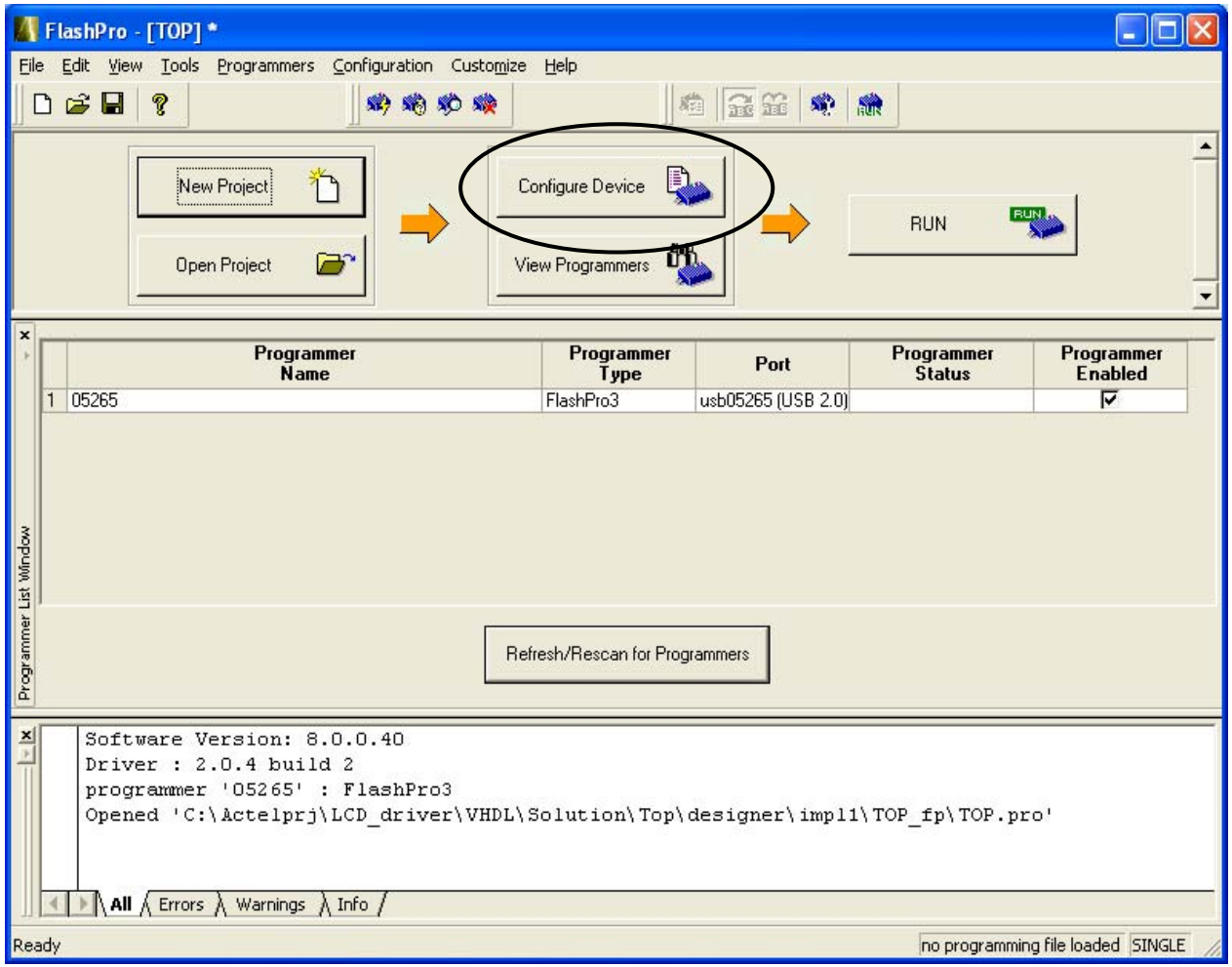
Invoking Programming Software from Libero

- Click “Programming” in Project Flow Window or…



FlashPro GUI

- Select Configure Device File to Display Programming File
  - Single File Configuration Window is Displayed

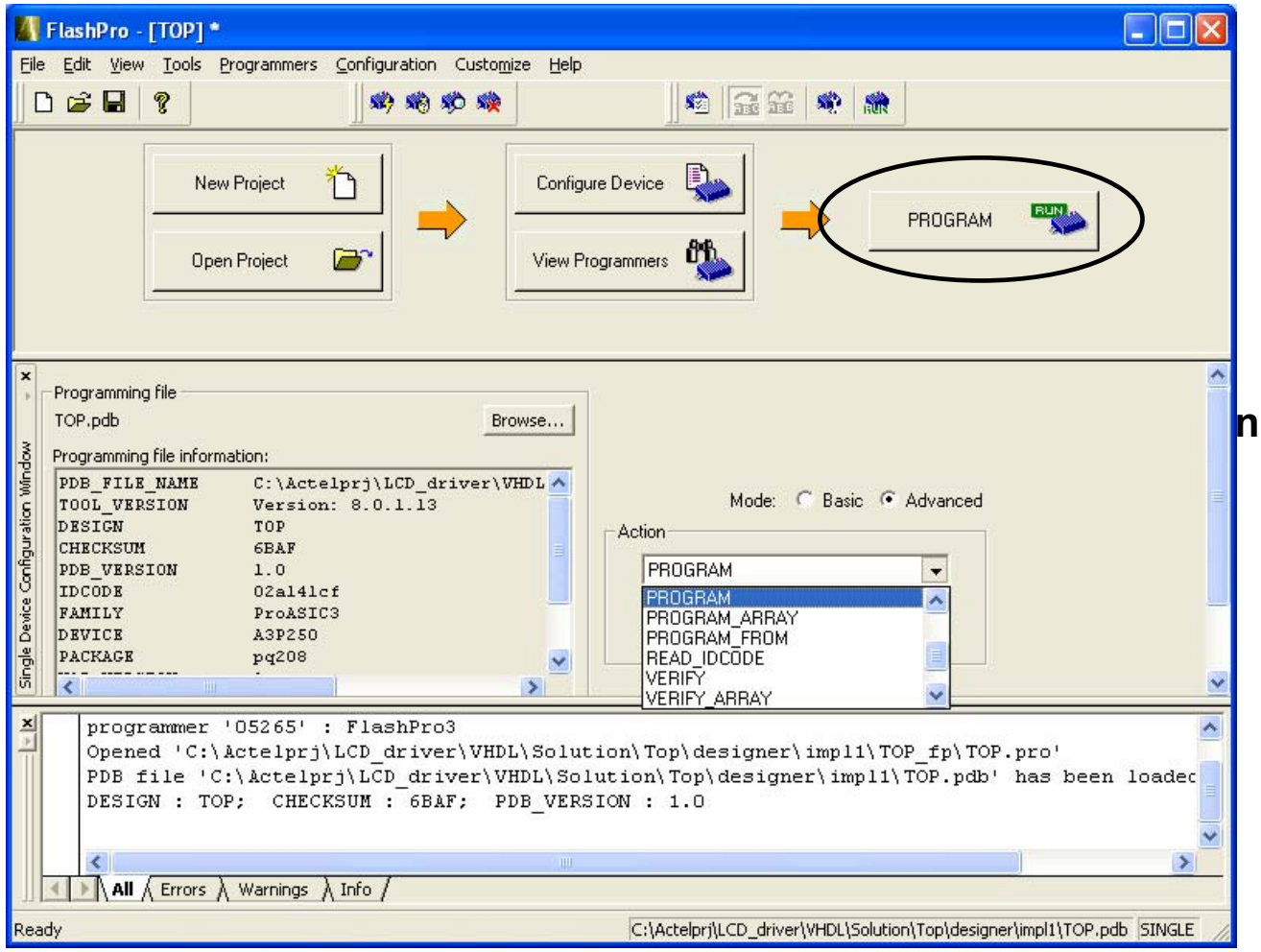


FlashPro

Select Action and Execute Action

- Select Action to Perform from Pull-down Menu

Programming file details

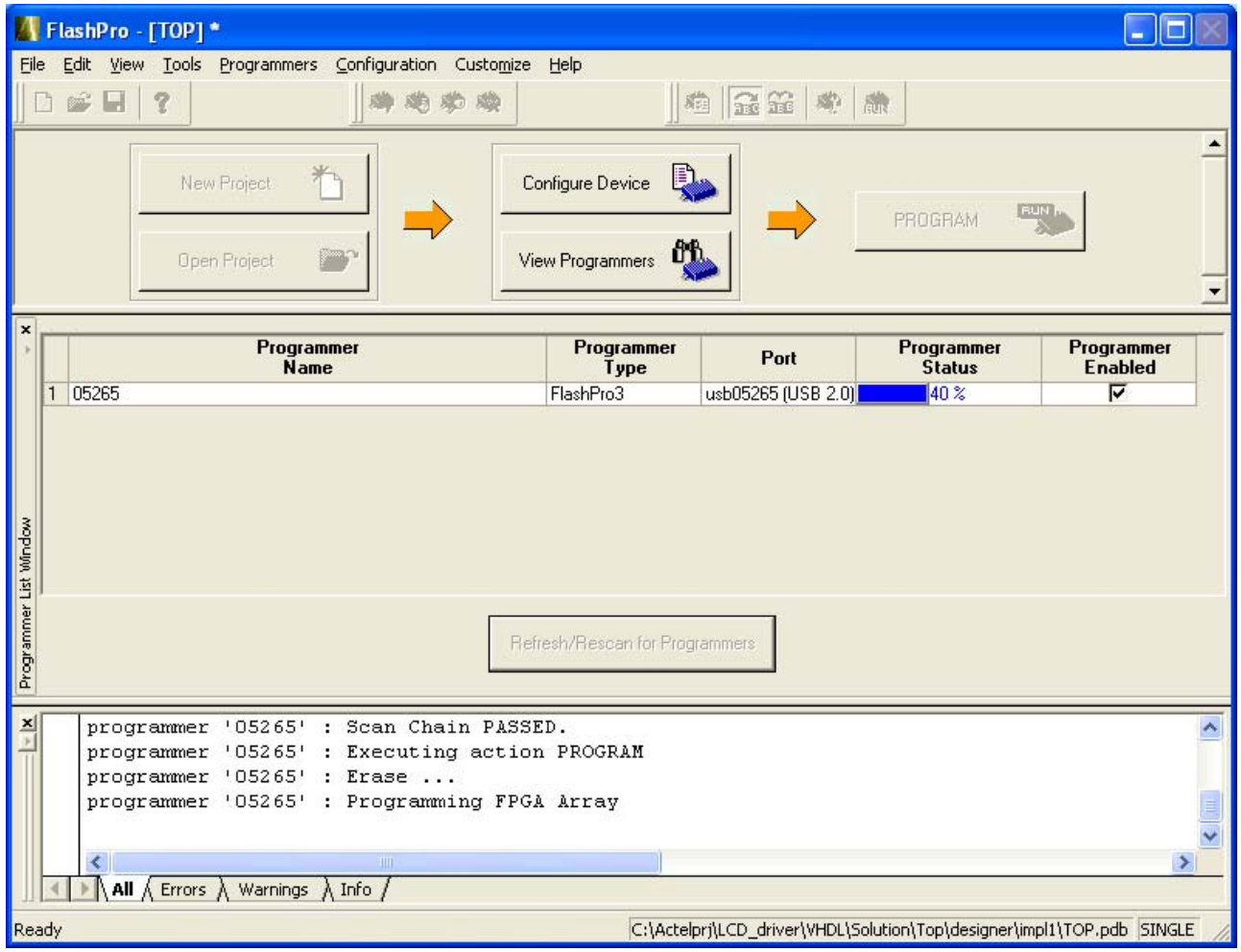


FlashPro

Programmer Status

- Programmer Status is Displayed in Programming List Window

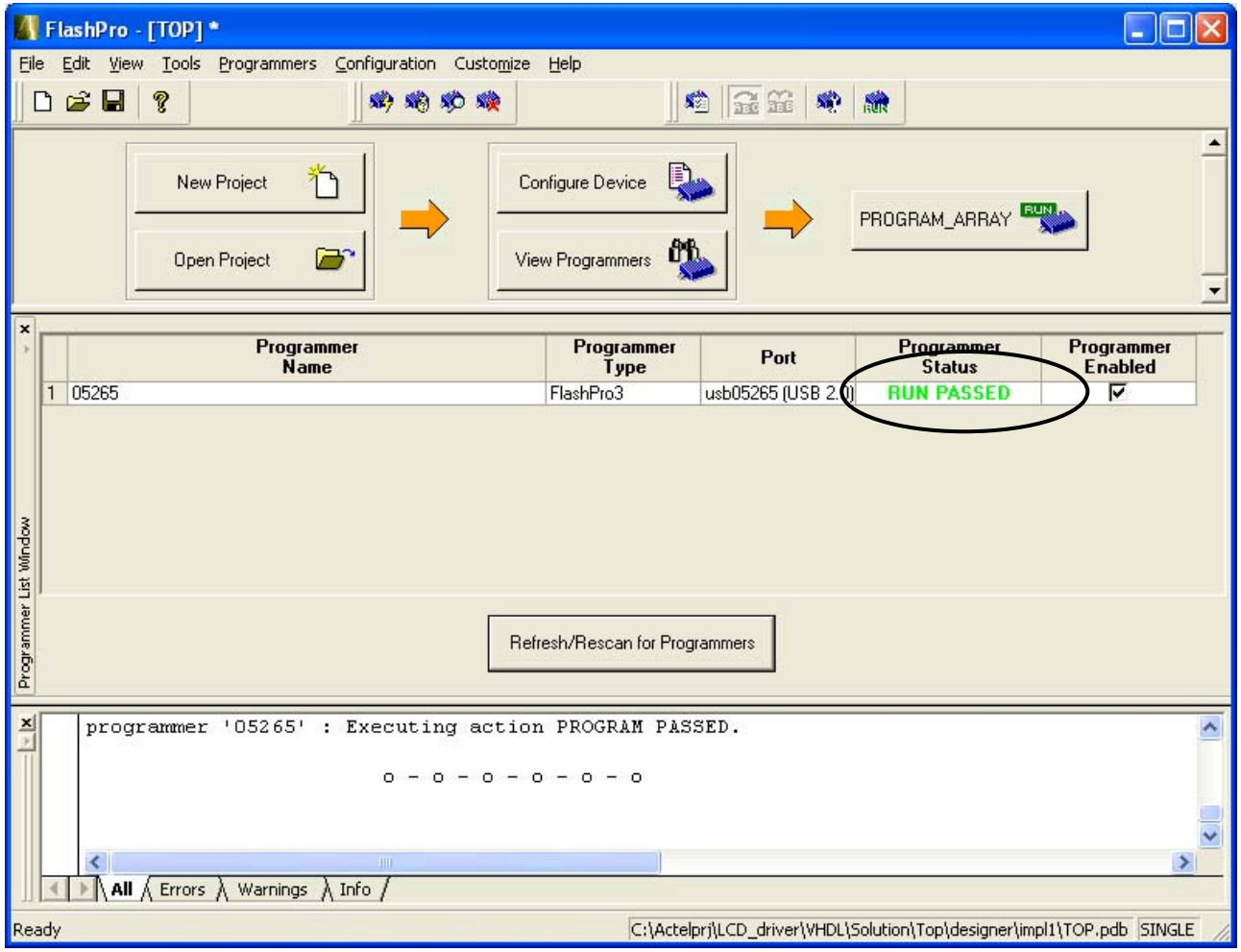
Programmer List Window



FlashPro

Programming Complete

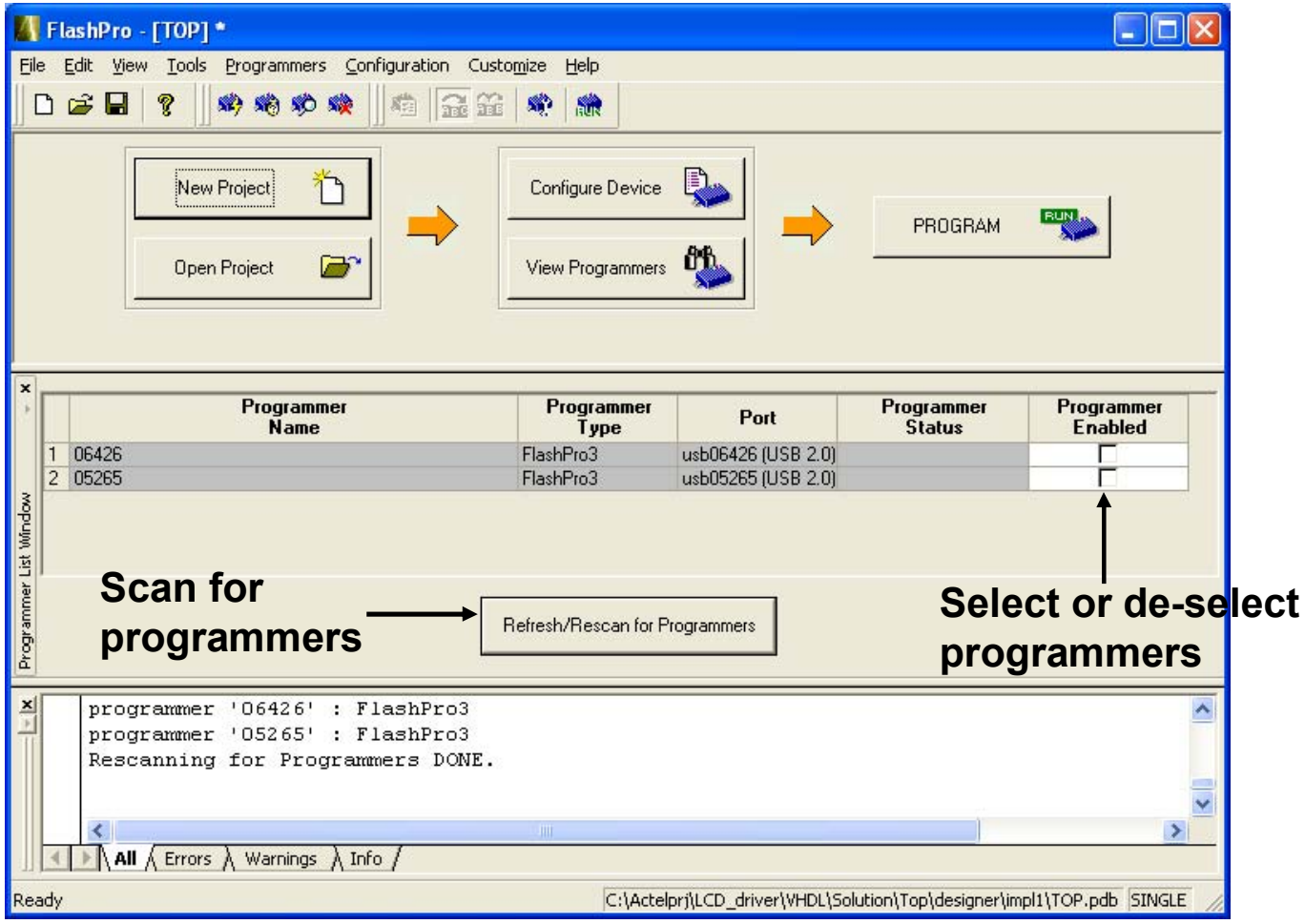
- Successful Programming is Indicated in the Programmer Status Column



FlashPro

Connecting Multiple Programmers

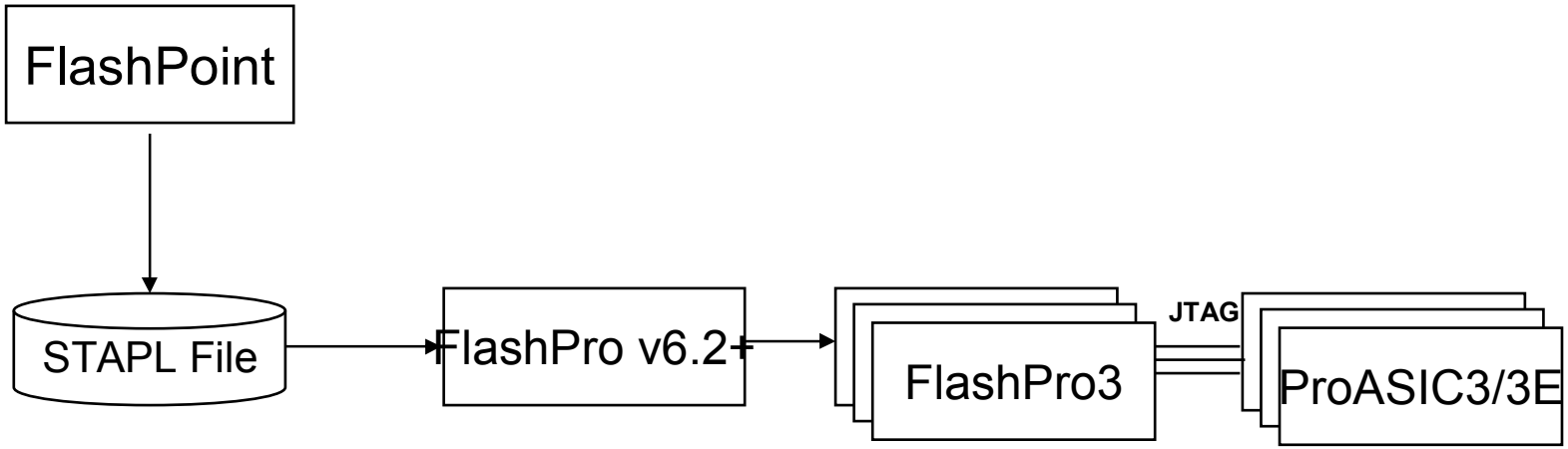
- FlashPro 8.5 Supports Connecting Multiple Programmers To One Computer



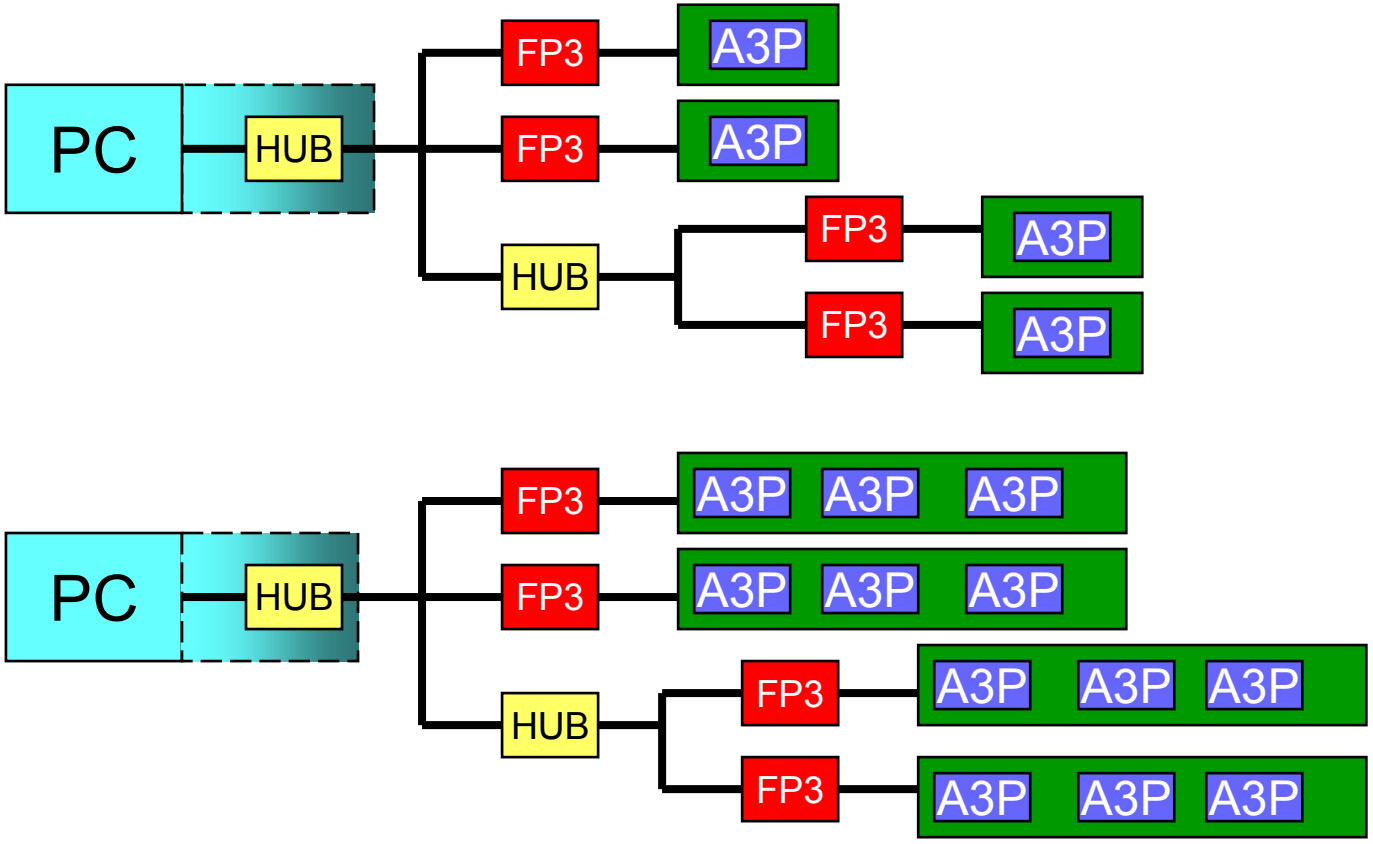
Parallel Programming

Parallel Programming

- **Operation/Production Programming**
  - **Programming multiple Flash devices using single GUI**



Parallel Programming (cont)

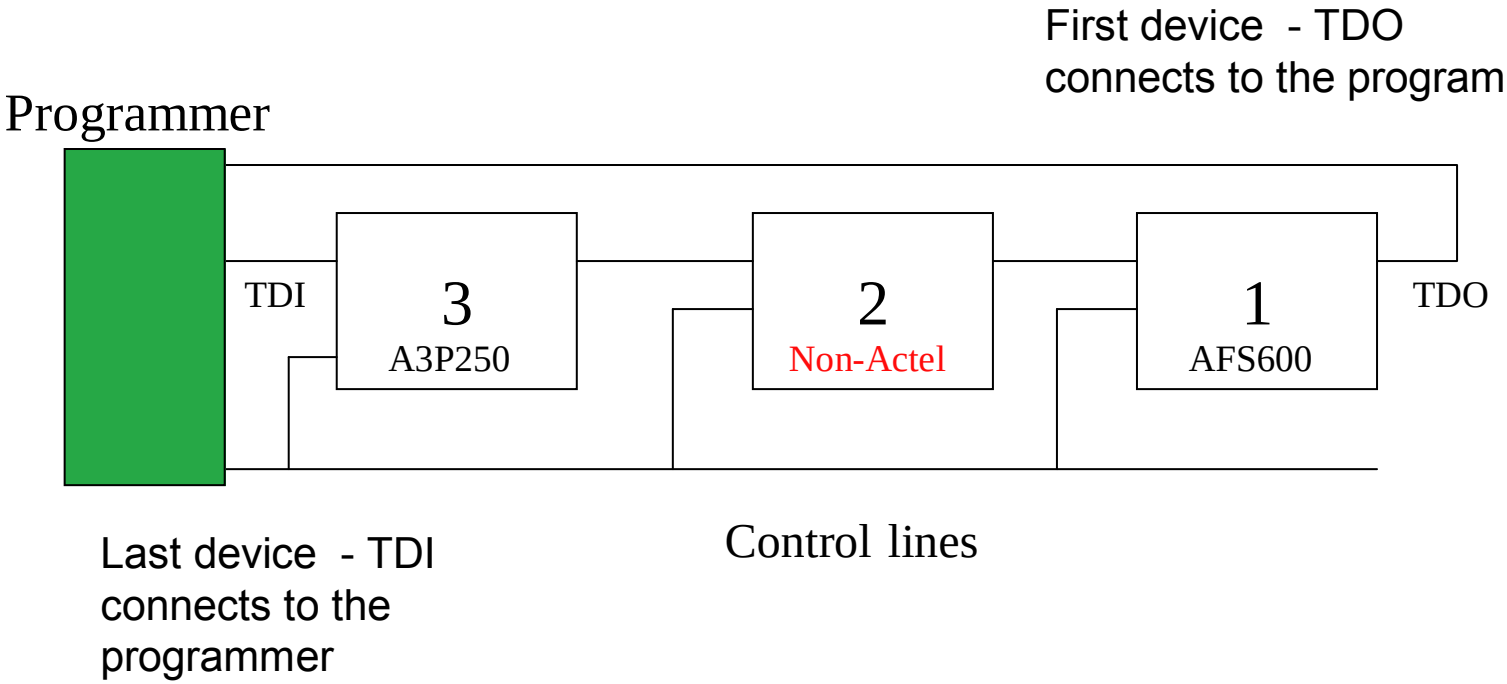


Chain Programming

Overview

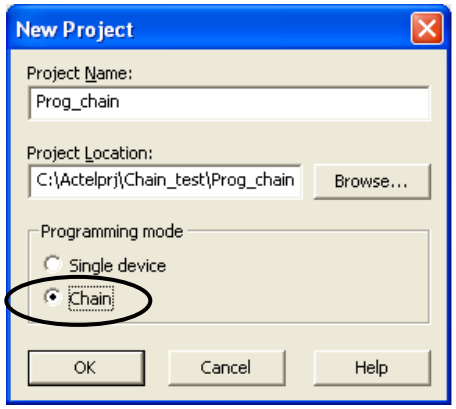
Why Chain Programming?

- **We want to program Actel devices with non-Actel devices in the same JTAG chain**
- **Devices are programmed in sequential order**



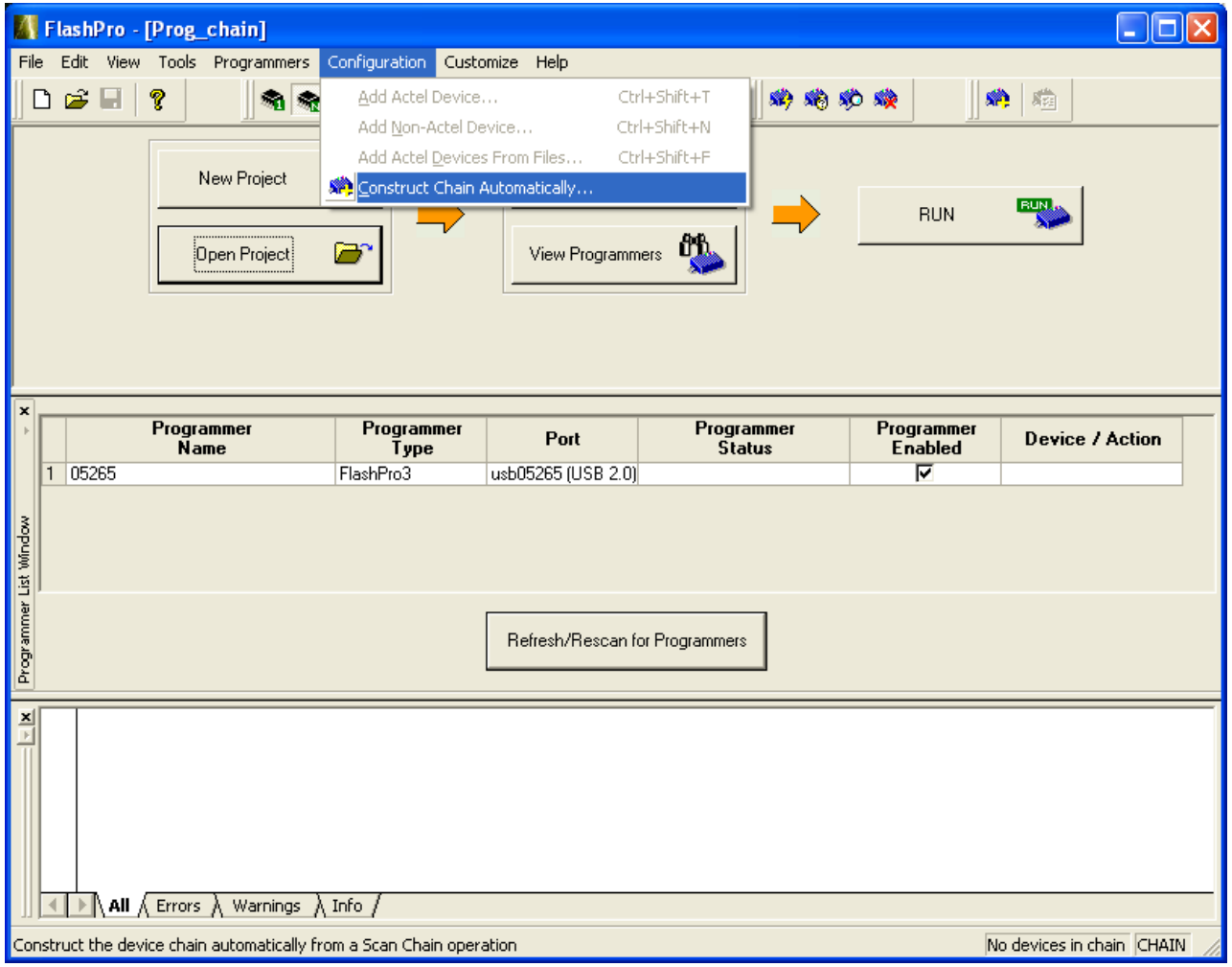
Chain Programming

- **Steps:**
  - **Create a FlashPro project**
    - **Specify Chain Programming**
  - **Configure the Chain**
  - **Select Programming file for device(s) to be programmed**
  - **Select BSDL file for device(s) to be bypassed**
    - **Can manually enter IR length and maximum TCK rate**
  - **Program Chain**



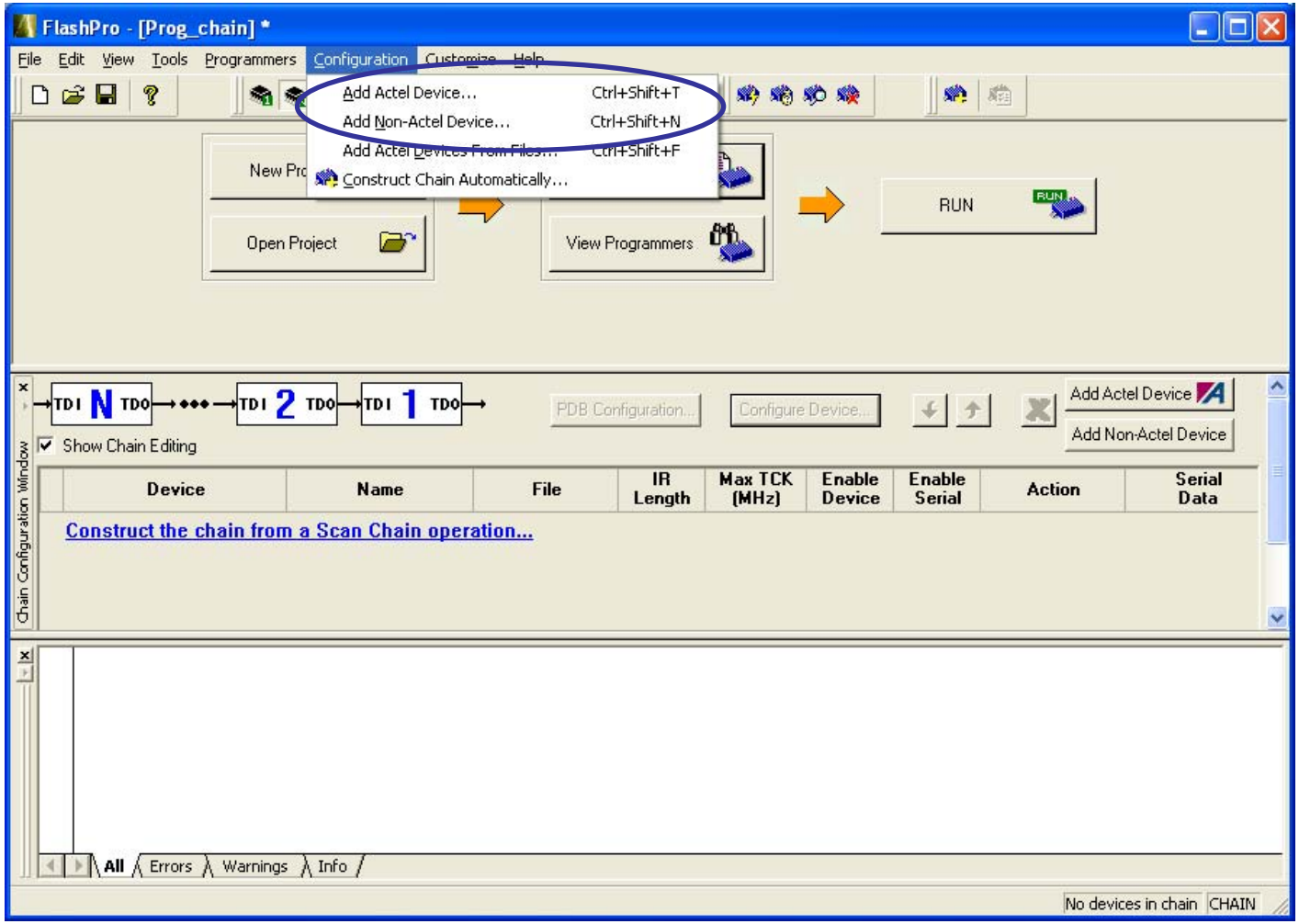
Chain Programming
Automatic Chain Configuration

- **Configuration > Construct Chain Automatically**

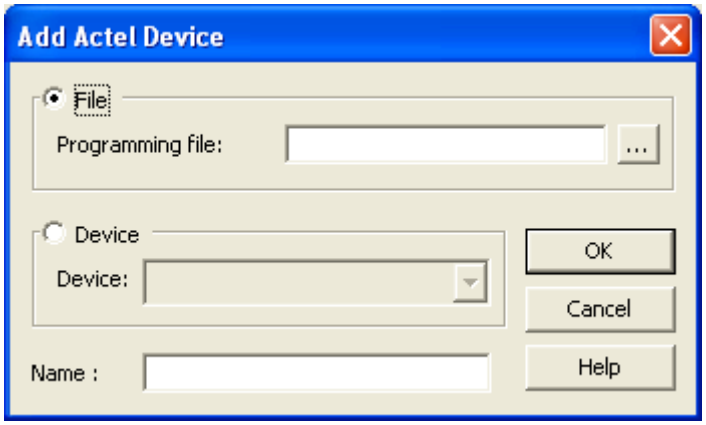


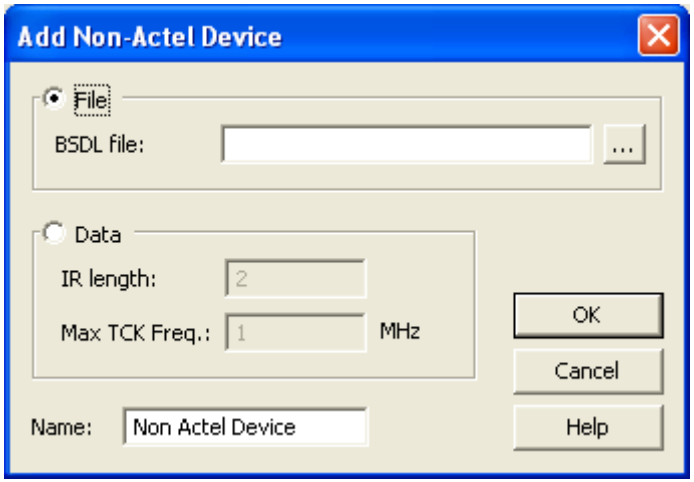
Chain Programming
Manual Chain Configuration

- **Manually Add Devices from Configuration Menu**



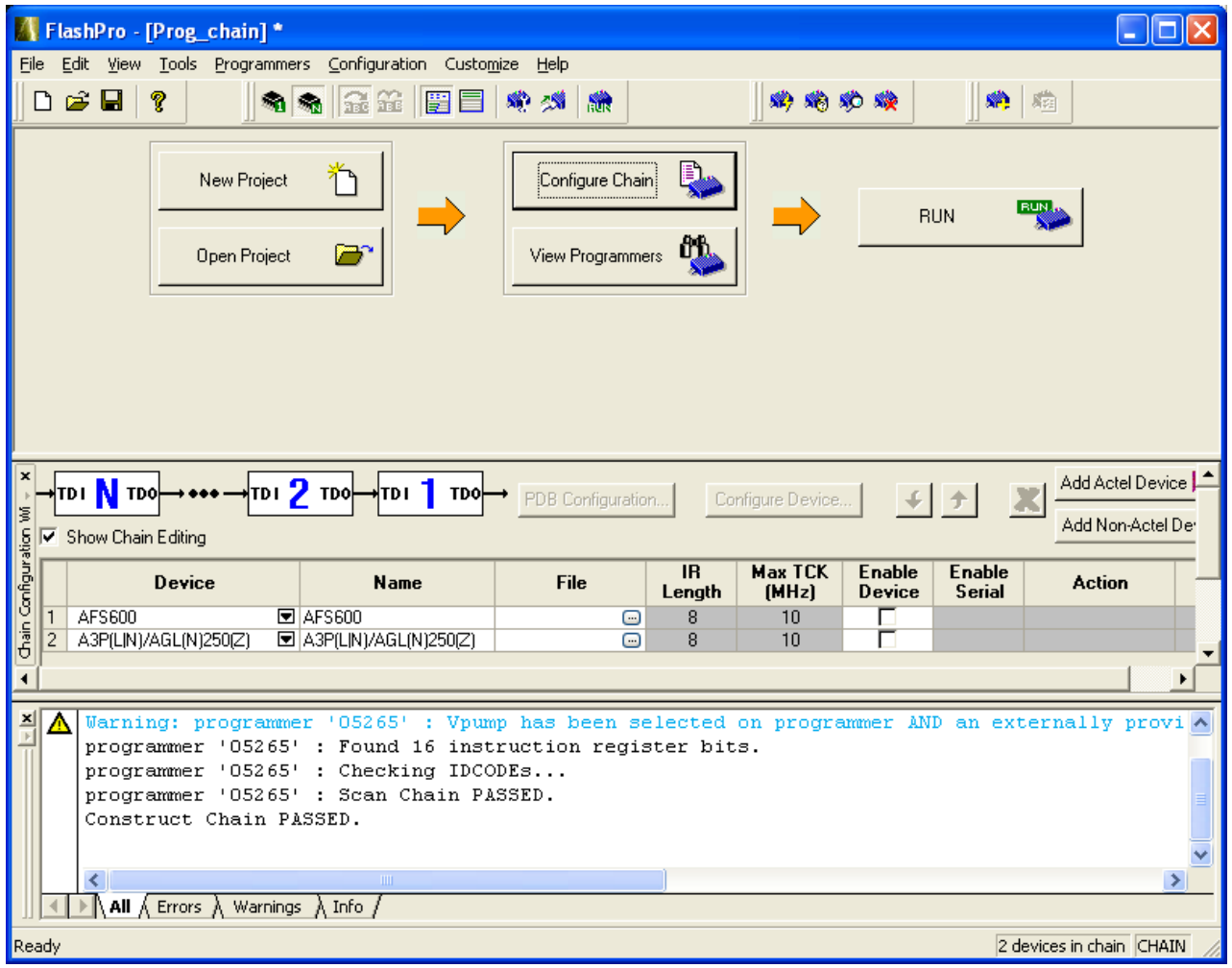
Manual Configuration
Adding a Device





- **Actel Device**
  - Select Programming file
  - User can specify custom name
- **Non-Actel Device**
  - User can select BSDL file
    - Provides the Max TCK Frequency and IR length for device
  - User can enter data directly if they do not have access to a BSDL file
  - User can enter custom name for device

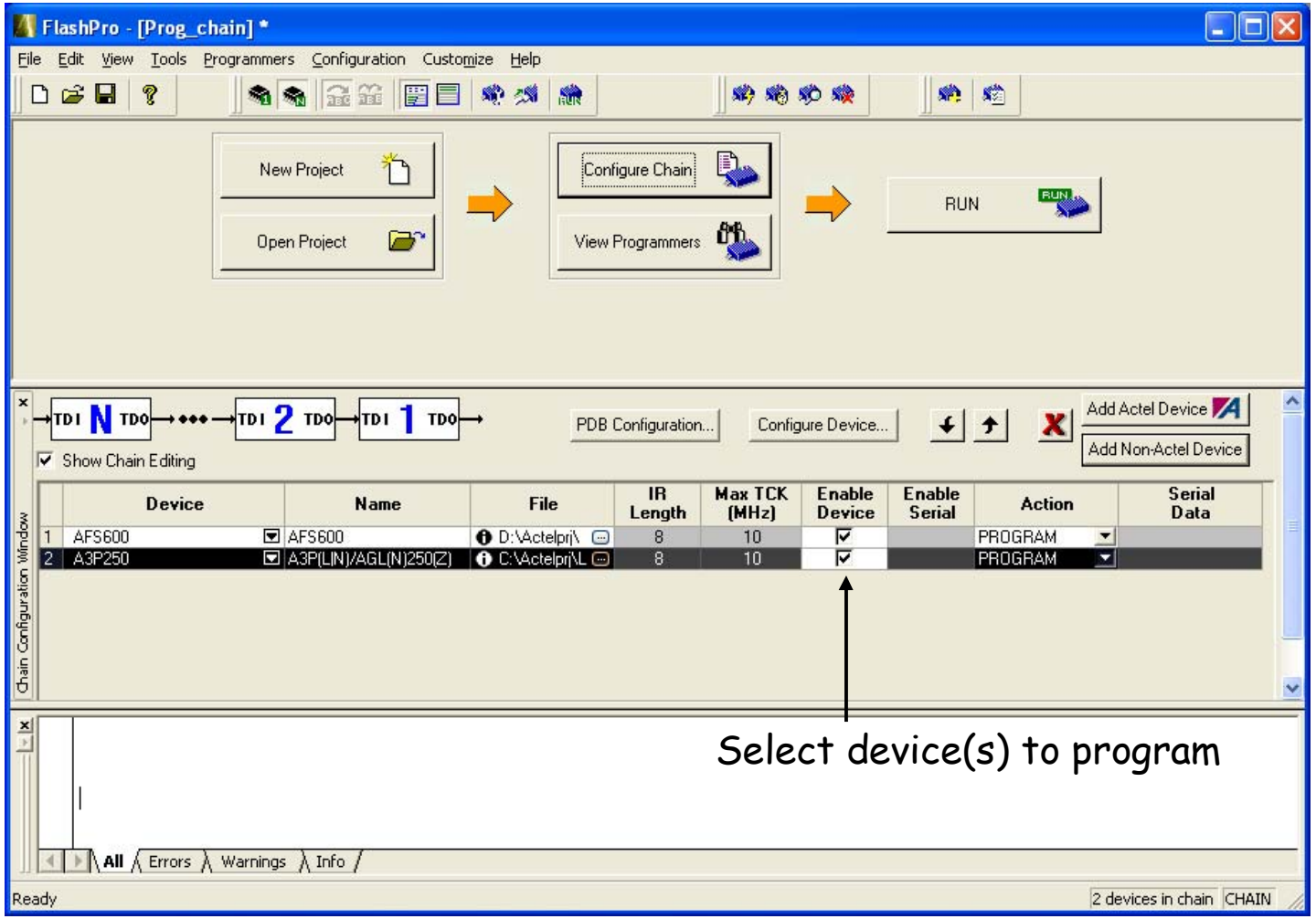
Configured Chain



Devices in chain

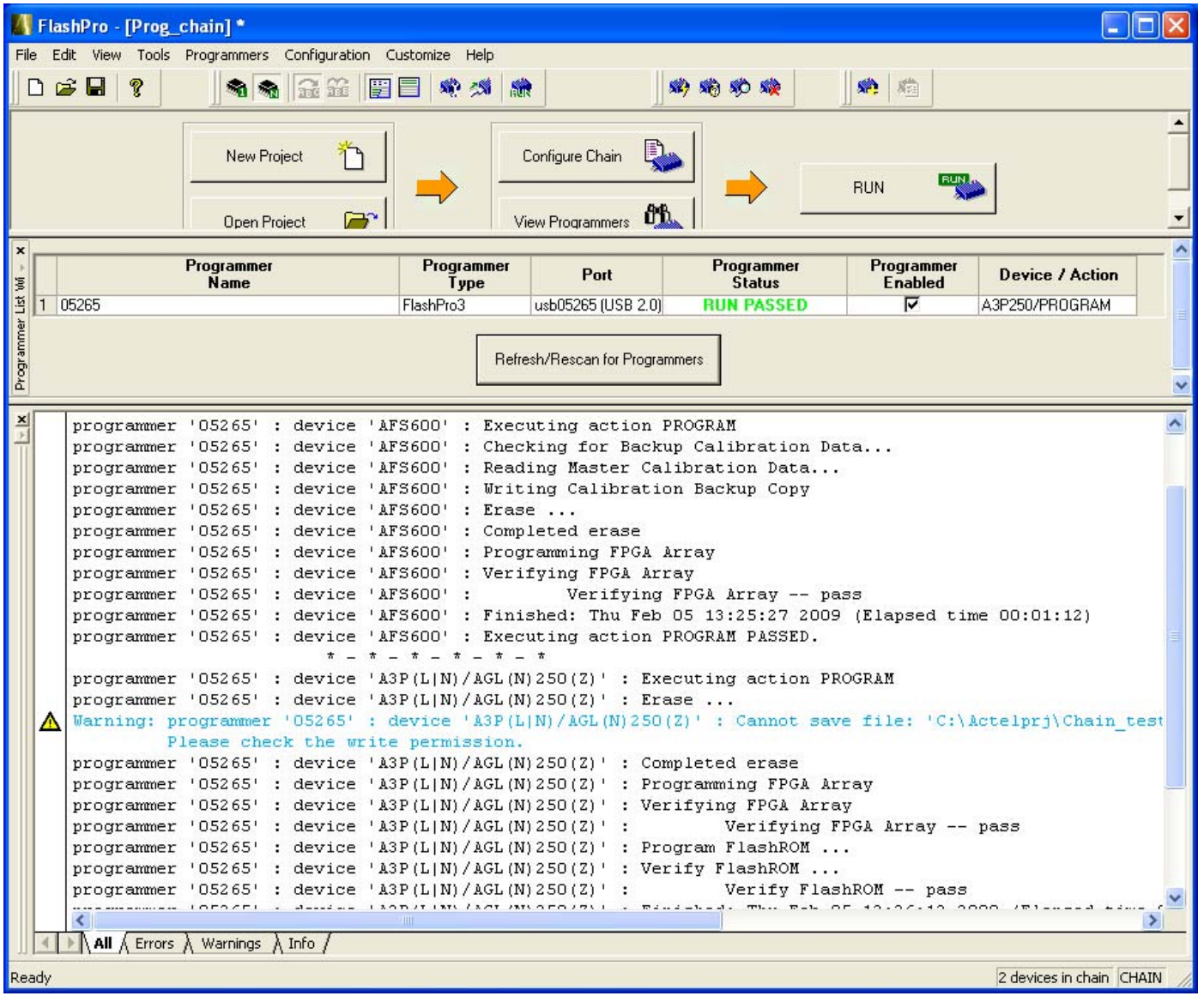
Log Window displays messages about chain configuration and programming

Programming the Chain



Programmed Chain

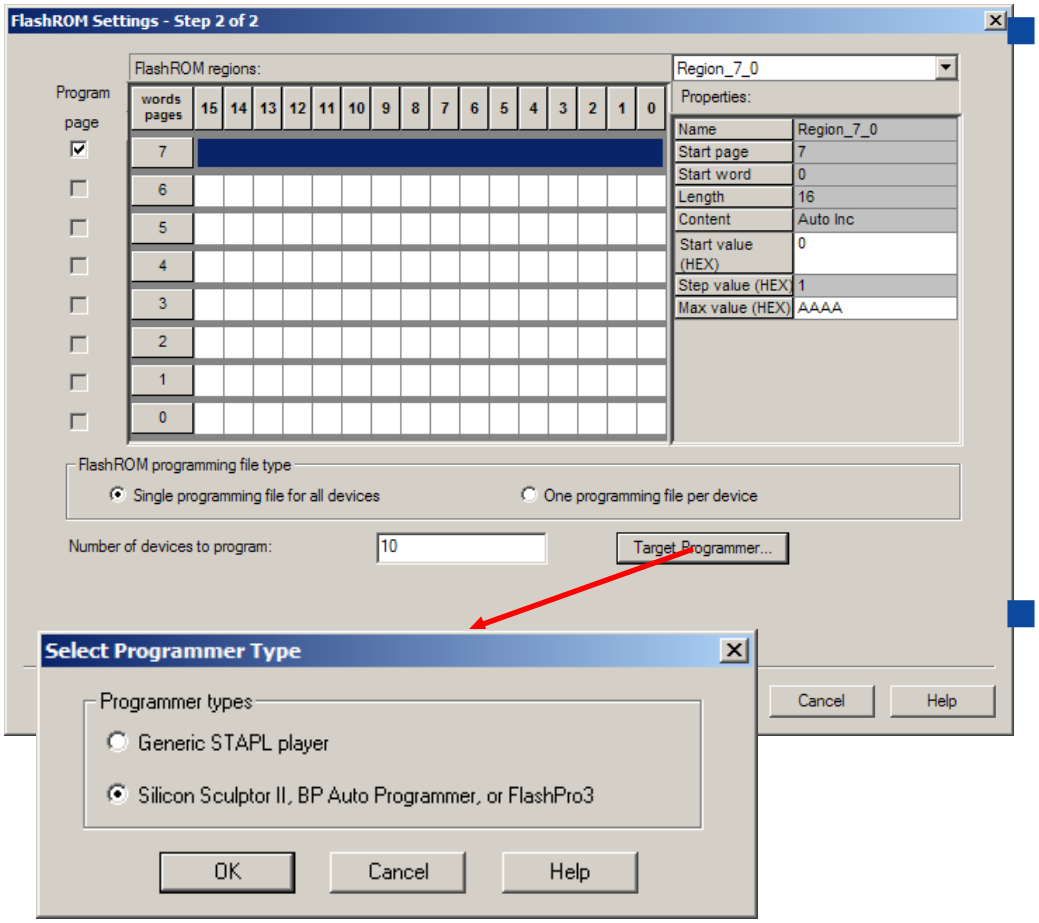
Messages indicate status of each device to be programmed



Device Serialization

Device Serialization: STAPL File Format

- **When Using FROM with Serialization Feature**
  - **User can select the target programmer**



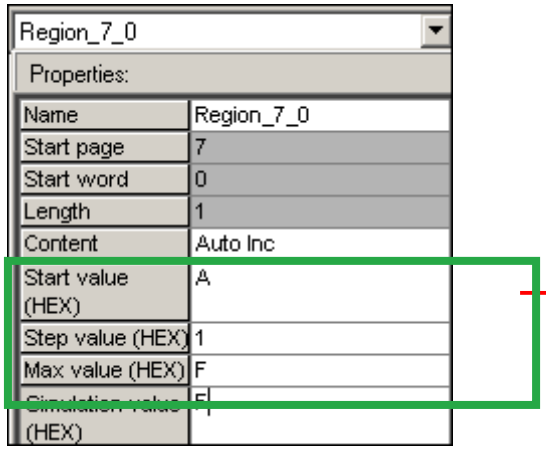
STAPL Serialization

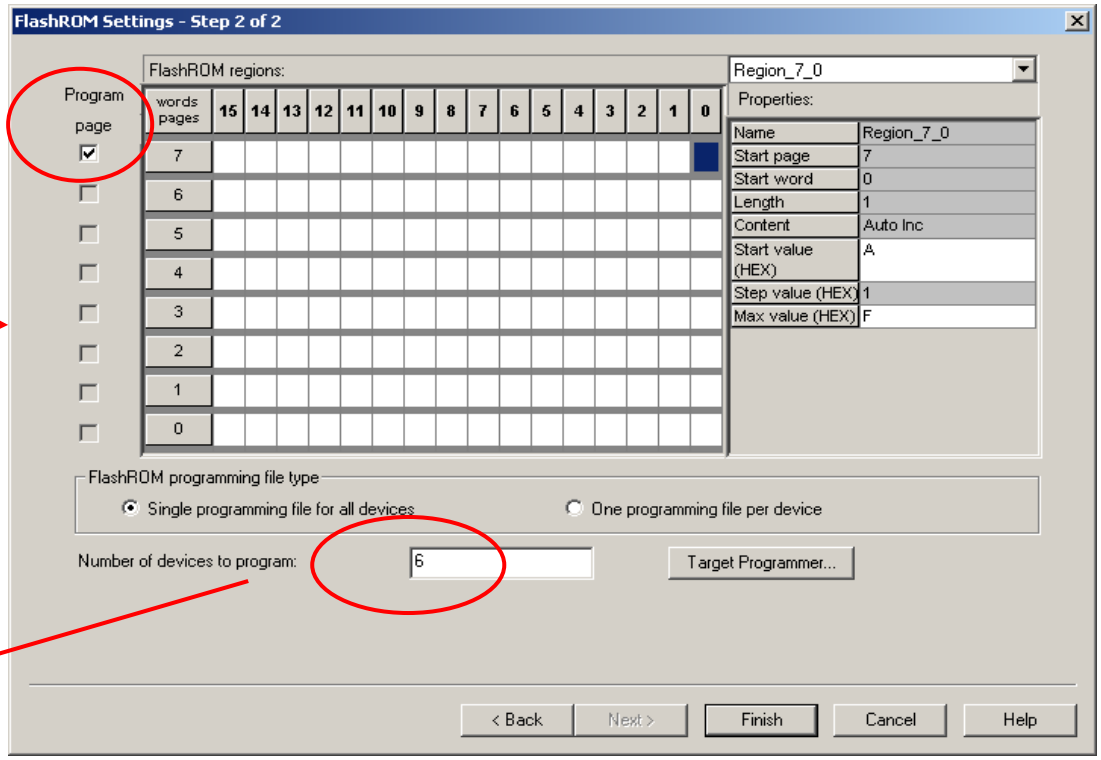
- **For Silicon Sculptor II or Sculptor 3, BP Auto Programmer or FlashPro3**
- **Actel and BP specific STAPL format to support device serialization**
- **Auto Serial Index Increment**

Generic STAPL Player

- **Each FROM and ARRAY content has its own action**

Device Serialization:
STAPL File Format





STAPL File

```
……..
DATA FROM_BITSTREAM_BLOCK;
BOOLEAN FRomStream_1[128]=
$0000000000000000000000000000000A;
BOOLEAN FRomStream_2[128]=
$0000000000000000000000000000000B;
BOOLEAN FRomStream_3[128]=
$0000000000000000000000000000000C;
……..
```

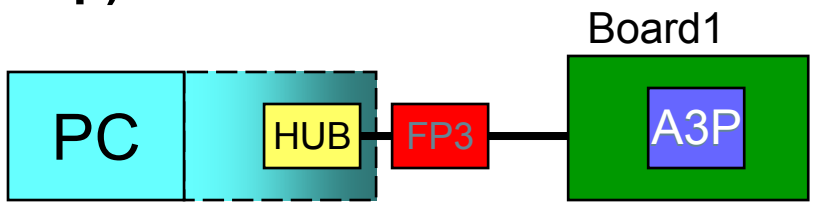
Device 1

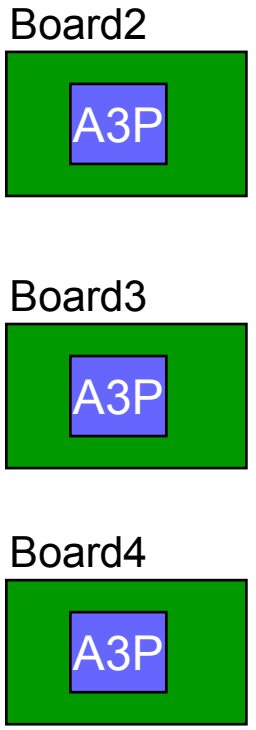
Device 2

Device 3

Serialization Options

- **What Should Happen when Programming fails on a Single Board with one Device?**
  - **User has GUI option in FlashPro to**
    - **Reprogram same serial number (Reuse)**
    - **Program with the next sequential number (Skip)**





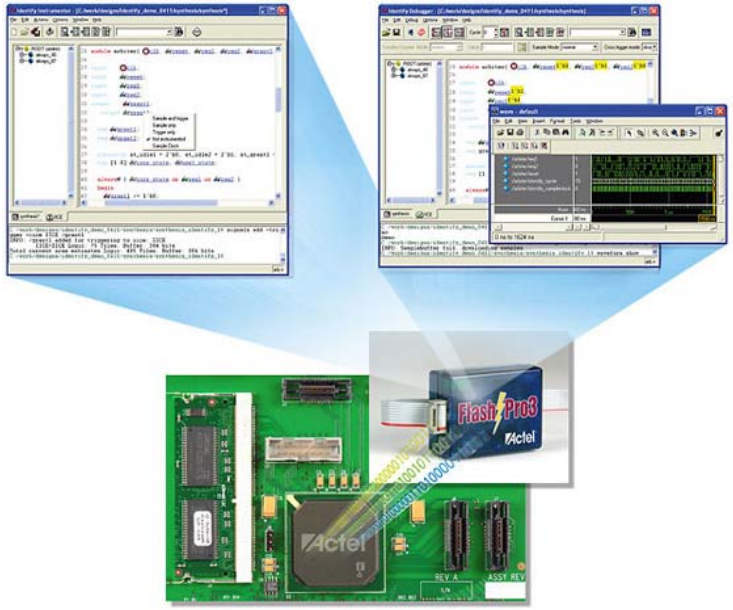
Debug Tools

FPGA Debug Tools

- Identify AE
- Silicon Explorer II

What is Identify AE?

- RTL Source Level Hardware Debugger from Synplicity
  - Allow internal visibility in the target system at full speed
  - Trigger on data path and control path
  - Instrument and Debug Design directly in RTL source code
  - View data in standard waveform display



Identify AE Product Specification

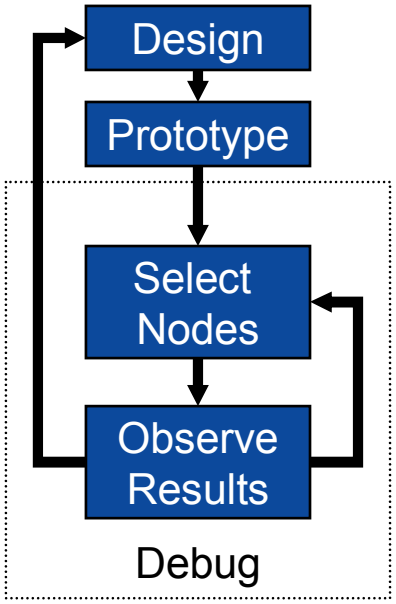
- Supported Devices
  - Identify AE v3.0.2A: ProASIC^{PLUS}, ProAsic3, Fusion and IGLOO
- Supported Programmers:
  - FlashPro, FlashPro Lite and FlashPro3
- HDL Language Support
  - VHDL, Verilog or mixed language designs
- FREE 1 year License

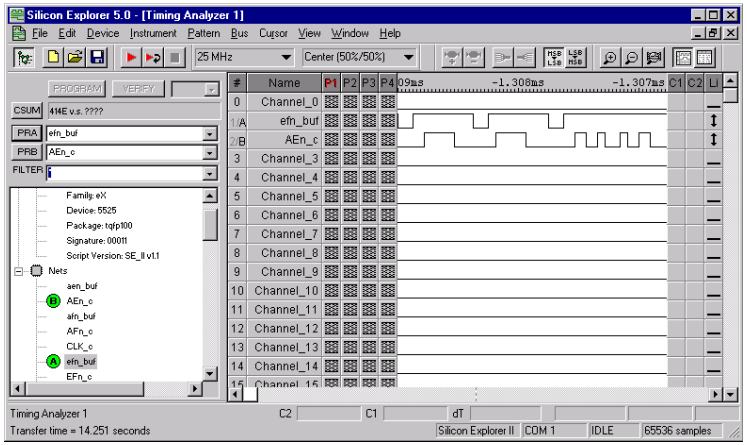
FPGA Debug Tools

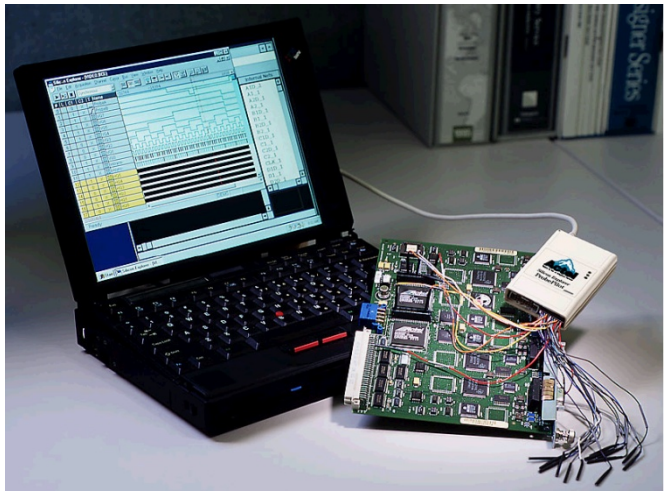
- Identify AE
- Silicon Explorer II

Silicon Explorer

- Debug Antifuse Designs in Real Time!
  - Select Internal FPGA Nodes on the Fly for Viewing while Device Runs at FULL Speed!
  - Reduce Debug Time and Decreases Time to Market!

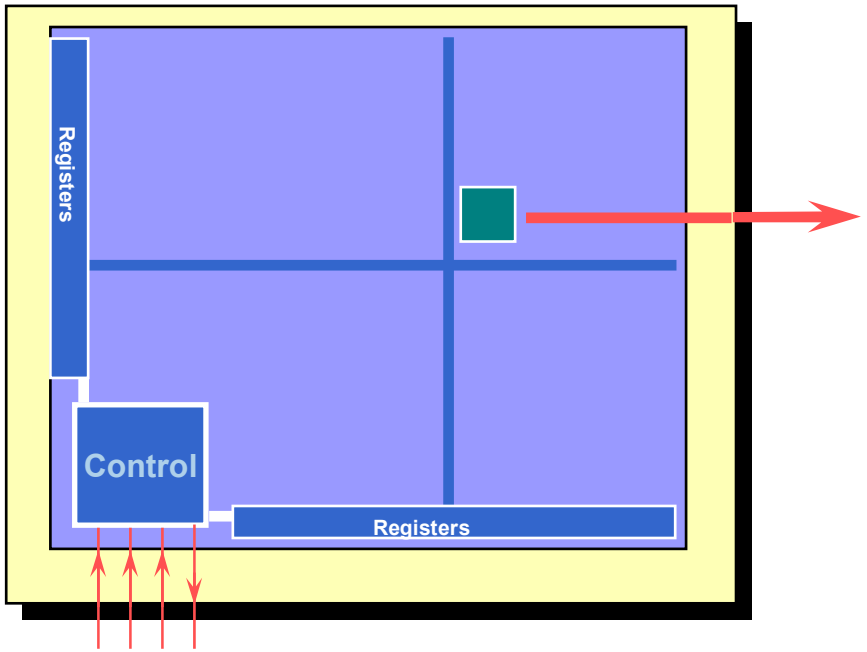






Action Probe Circuitry

- Dynamic Internal Node Access
  - No Changes to Timing Relationships
  - No Changes to Fan-out or Node Loading
- Patented Architectural Feature
  - Antifuse Devices Only
  - Unique to Actel
- No Silicon Overhead
  - Uses Zero Logic Resources
  - Always there if Needed



Summary

- Libero FPGA Design Suite Includes:
  - Design Entry
    - ViewDraw, HDL Editor, SmartGen, SmartDesign
  - Synthesis
    - Synplicity
  - Verification
    - ModelSim, WaveFormer Lite
  - Designer (P&R, Timing Analysis and Constraints)
- Actel Continues to Improve Libero IDE
  - Increased Quality of Results
  - Ease of Use
  - Additional Features

Reference Material

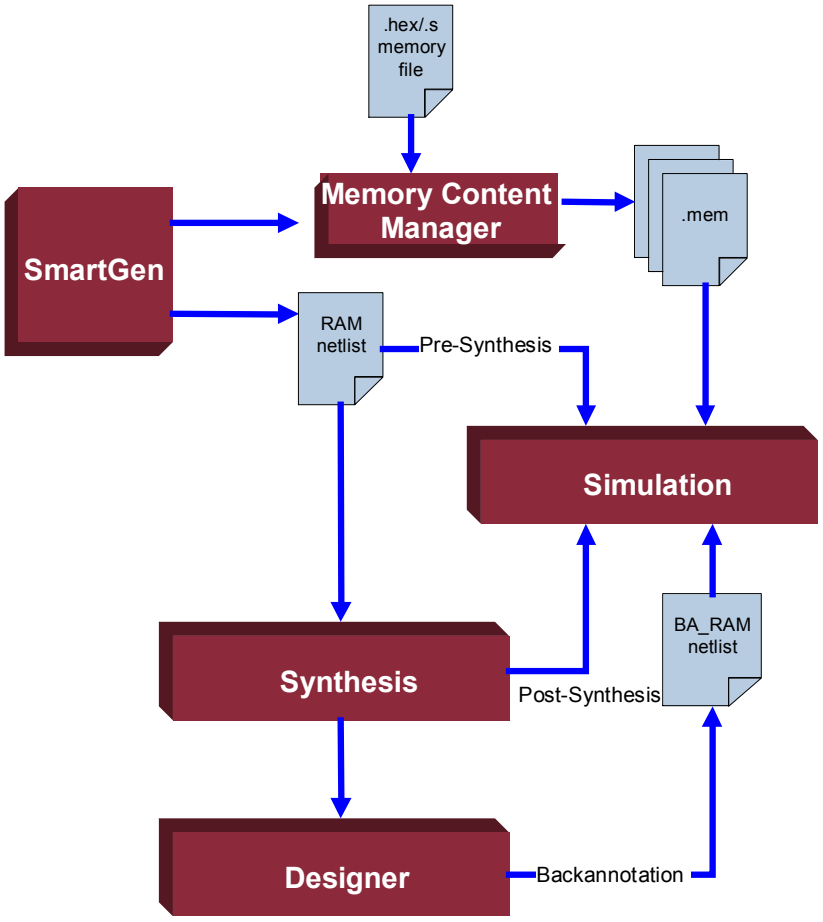
Reference Material

- SmartGen
- ViewDraw
- CoreConsole
- WaveFormer Lite
- ModelSim Dataflow and List Windows
- Synplify Attributes and Directives
- Chip Editor
- SDC Timing Constraints
- Silicon Sculptor
- Device Serialization

SmartGen

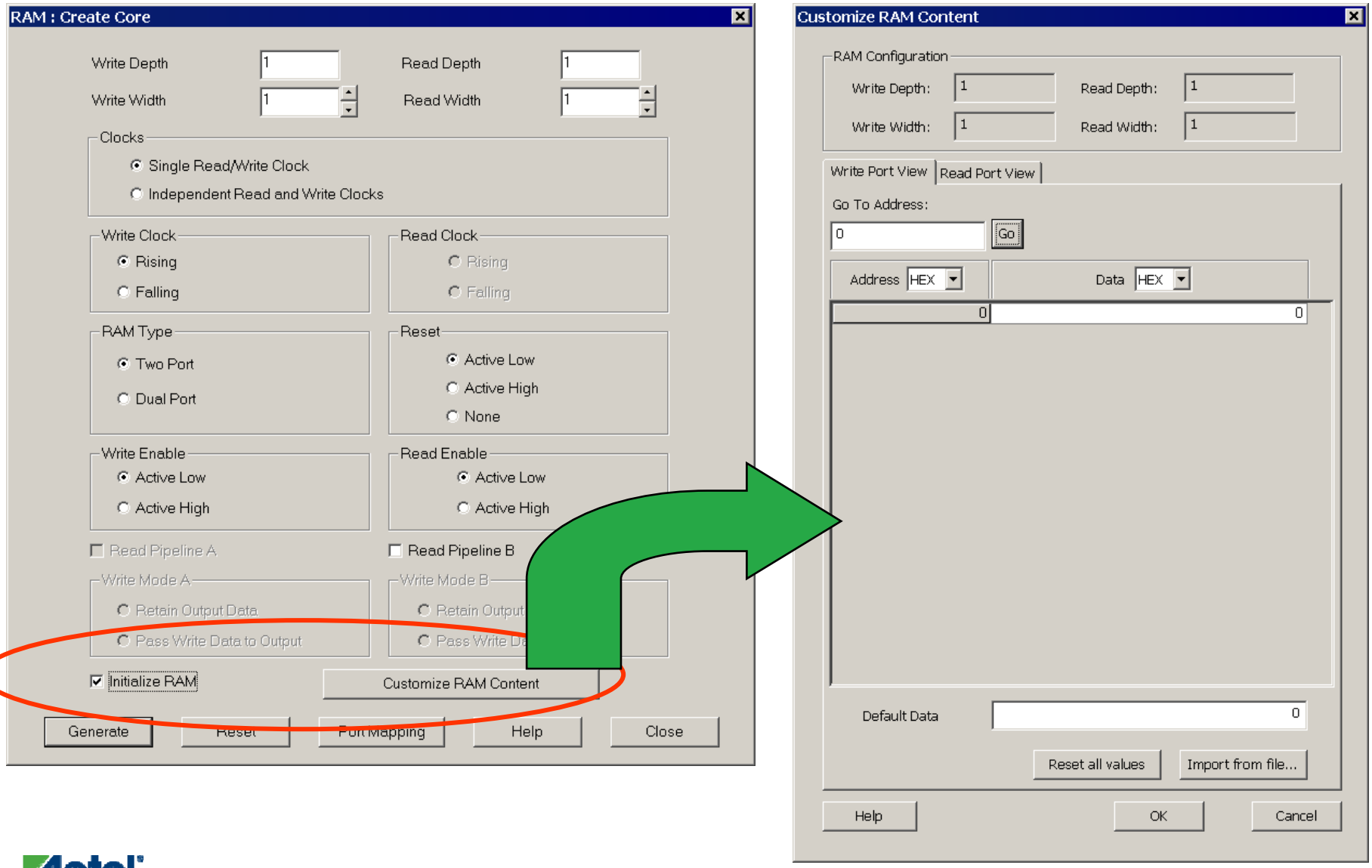
SmartGen RAM Initialization

- Speeds Simulation Time by Reducing the Initial Writes to Setup the Memory
- Completely Automated Simulation Flow
- MEMORYFILE Property Preserved Throughout the Design Flow
- Cascading of Multiple RAM Blocks Handled Automatically
- Supports Industry Standard Memory Content Specification Files



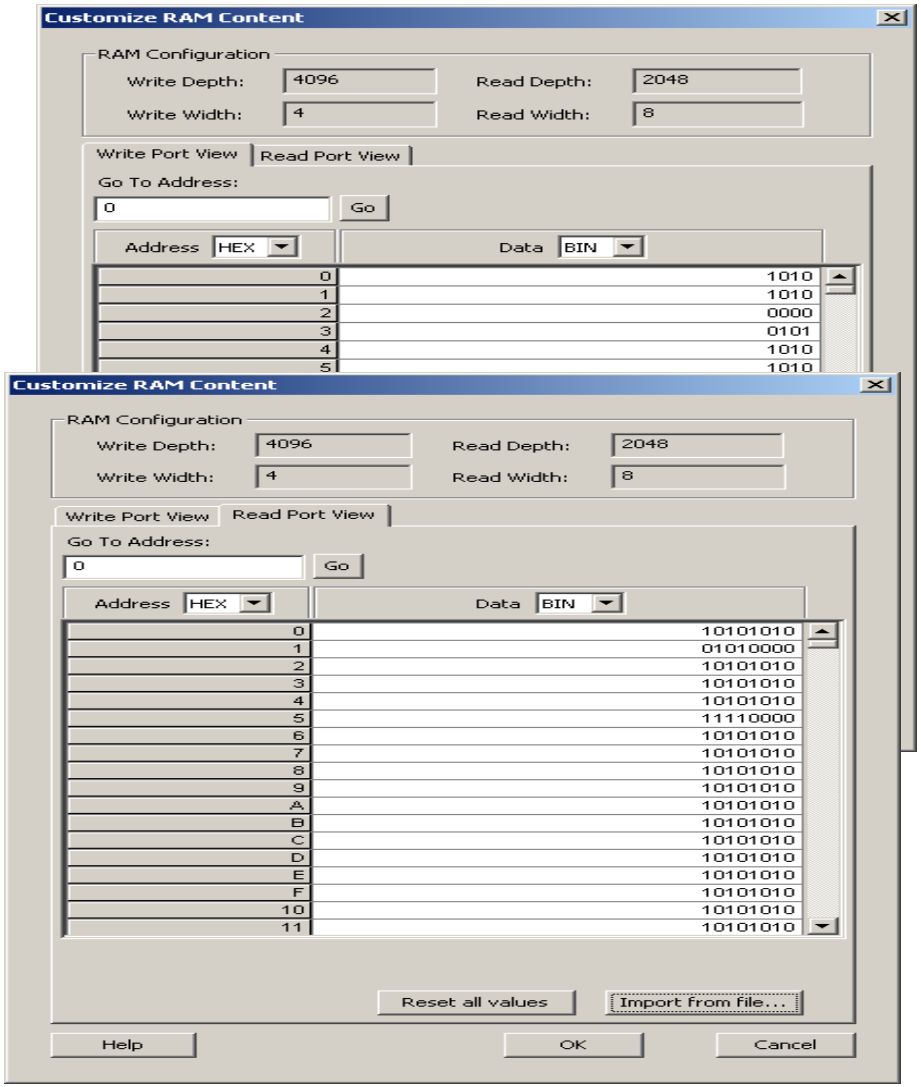
SmartGen RAM Initialization

ProASIC3 and Axcelerator (cont.)



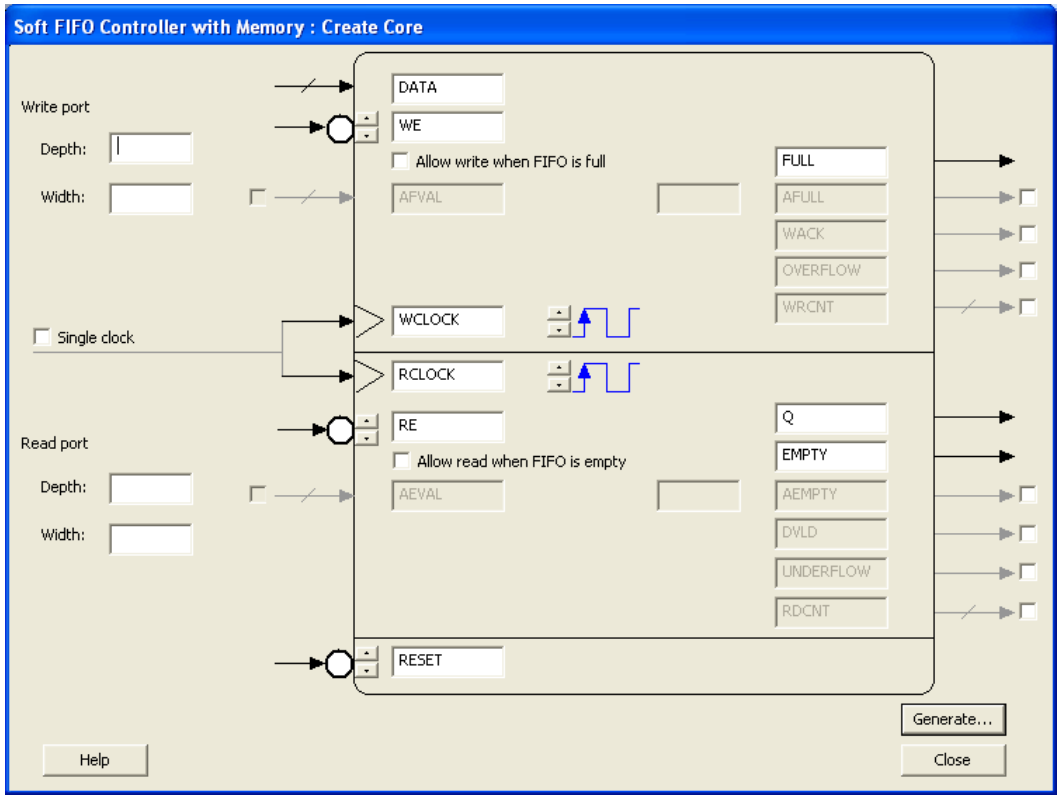
Memory Content Manager - Features

- Variable Aspect Ratio Support
  - Read View & Write View
- Import of User Memory File
  - Intel hex format
  - Motorola- S format
- Multiple Radix for Data and Address Display
  - Hexadecimal
  - Binary
  - Decimal



SmartGen Soft FIFO Controller

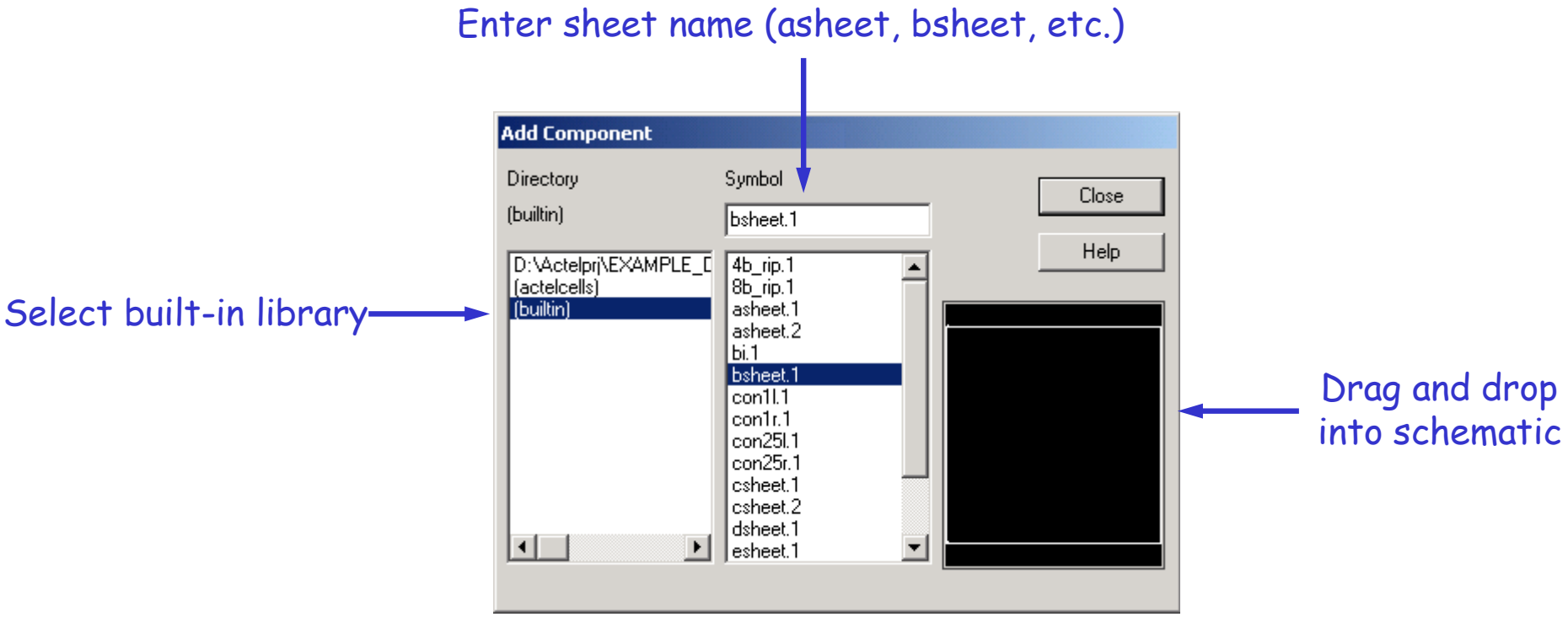
- Create Controller With or Without Memory



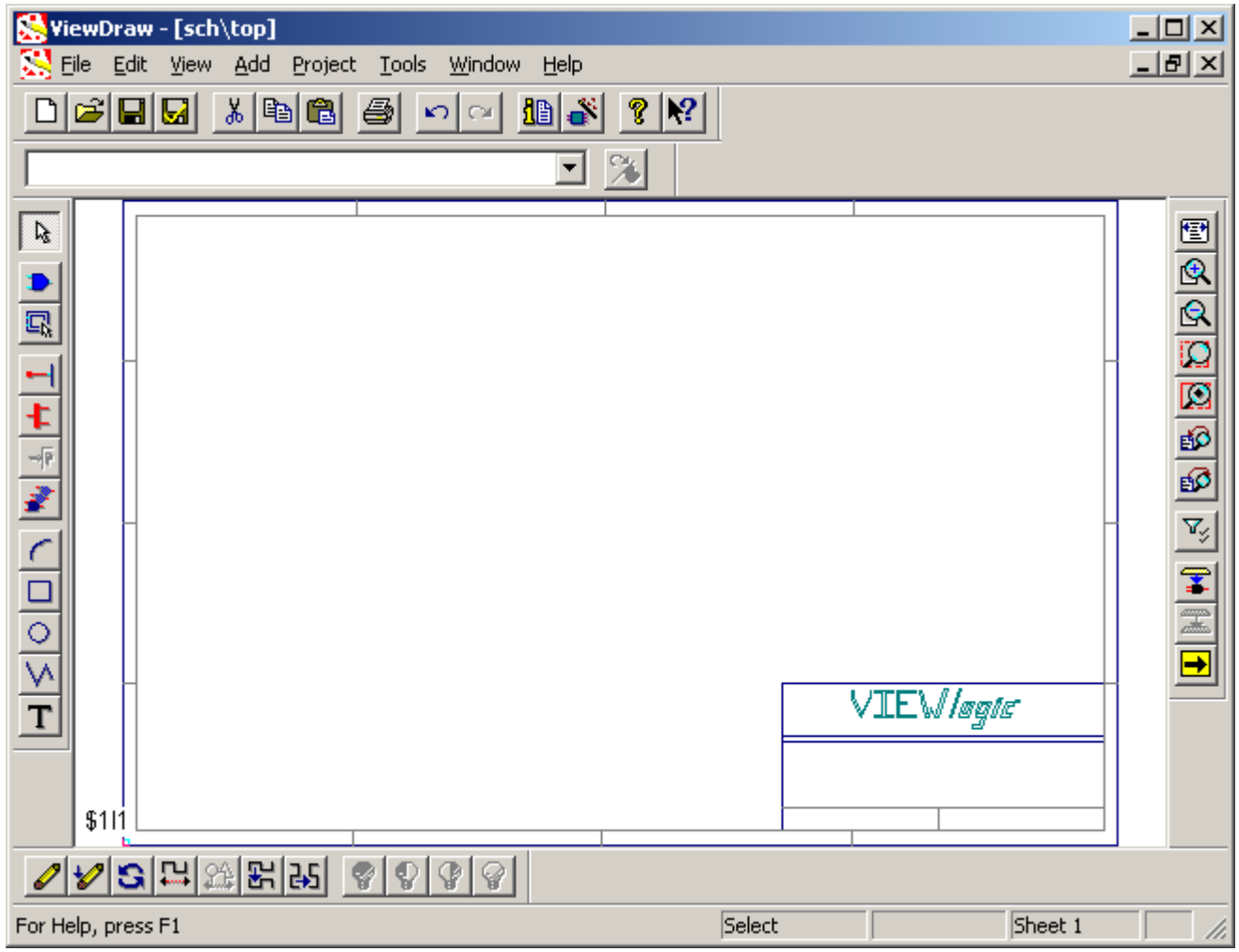
ViewDraw

Adding Schematic Border

- Built-in Library Contains Several Sheet Border Templates
  - Templates Can Be Modified

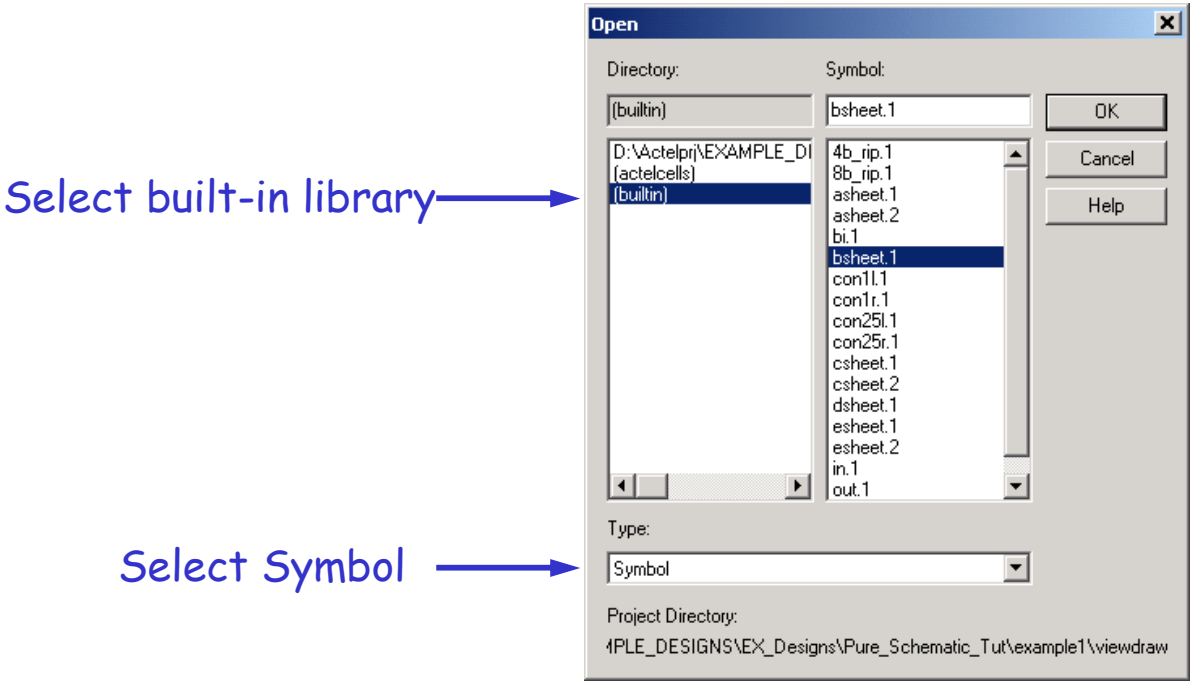


ViewDraw Border in Schematic



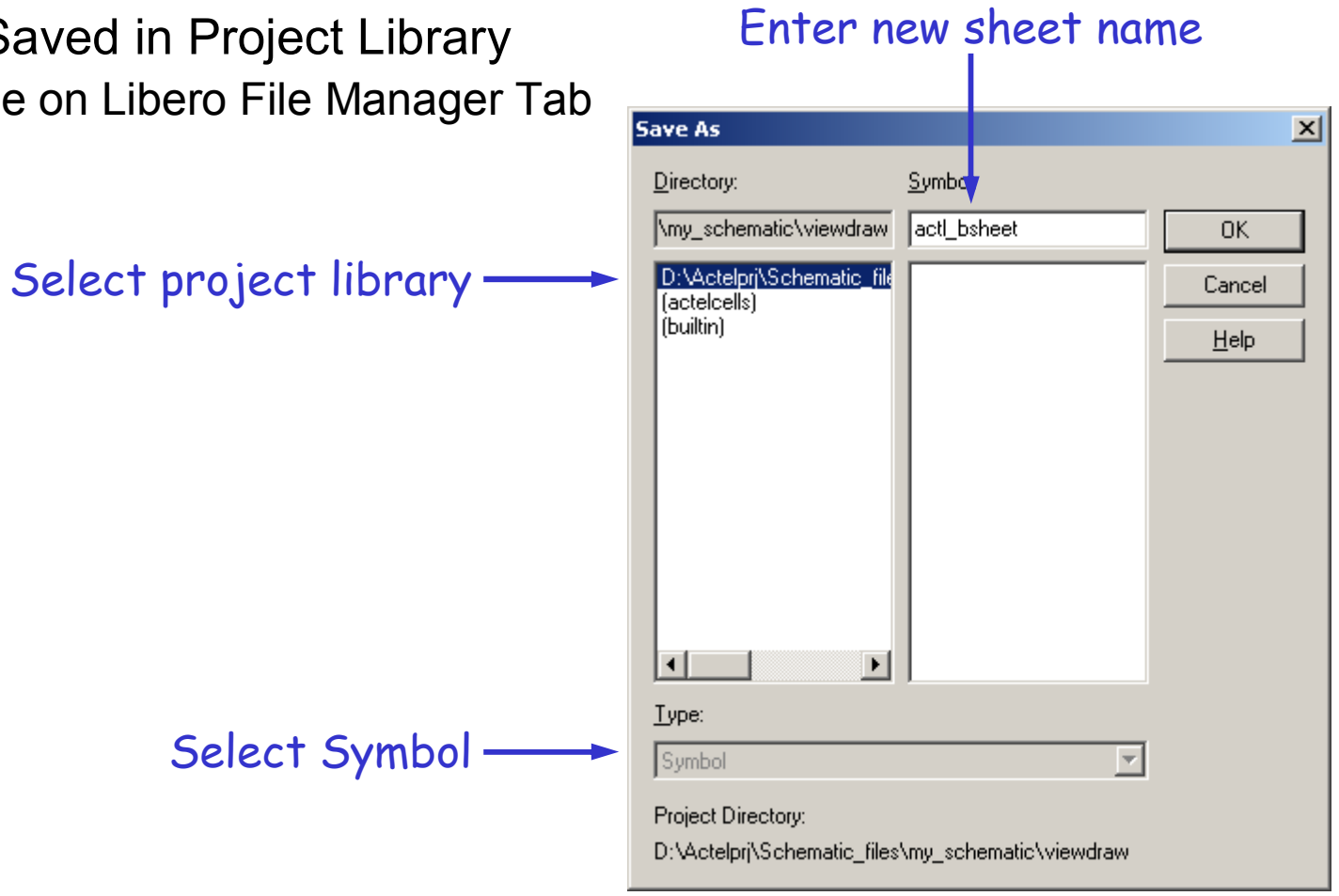
Customizing a Schematic Border

- Border Template Can Be Customized
- Open Border (File > Open)
  - Select Symbol from “Type” Menu



Customizing a Schematic Border (cont.)

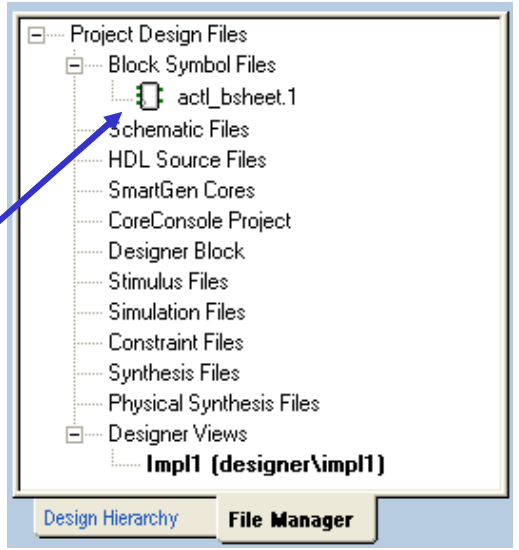
- Save File to New Name
  - (File > Save Copy As <name>)
  - Border Saved in Project Library
    - Visible on Libero File Manager Tab

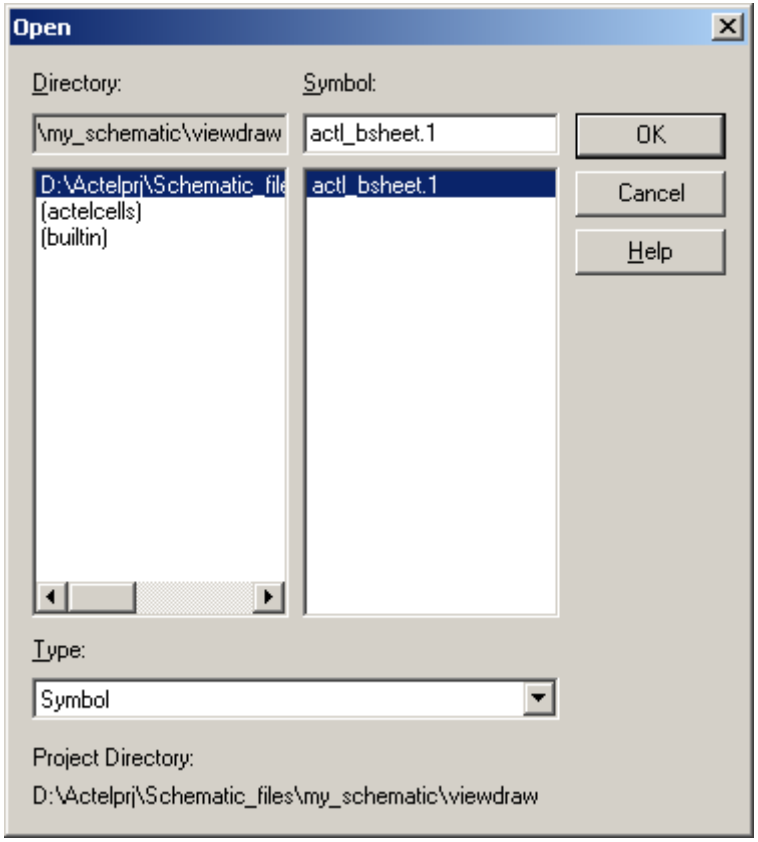


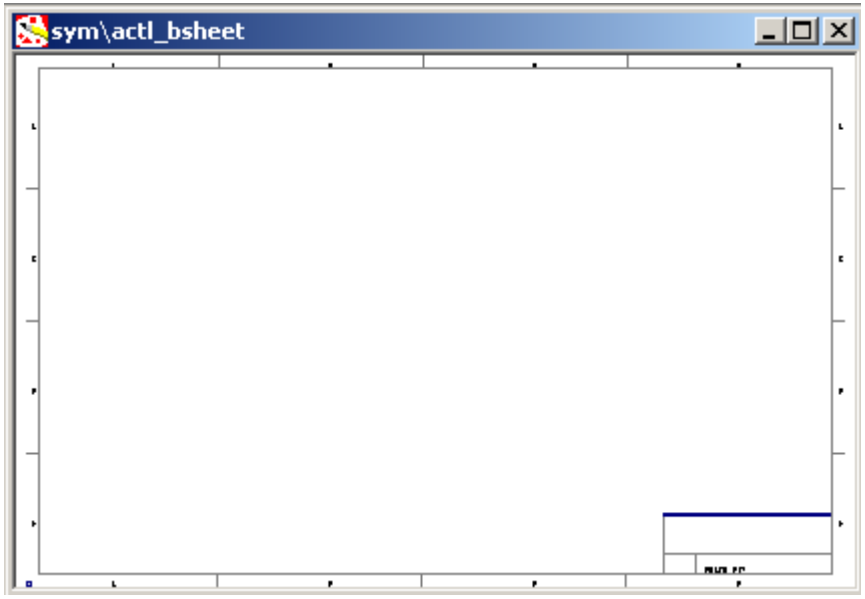
Customizing a Schematic Border (cont.)

- Open Saved Border and Edit (File > Open)
  - Add Lines, Arcs, Text, etc. as Necessary

Modified border visible on File Manager tab

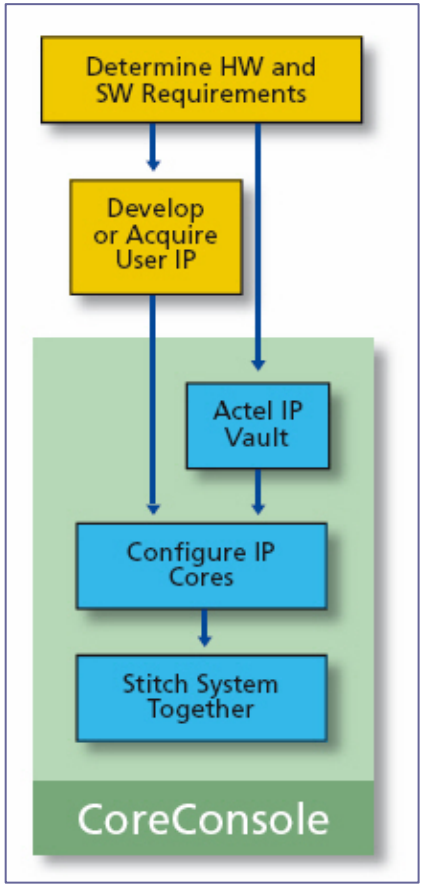






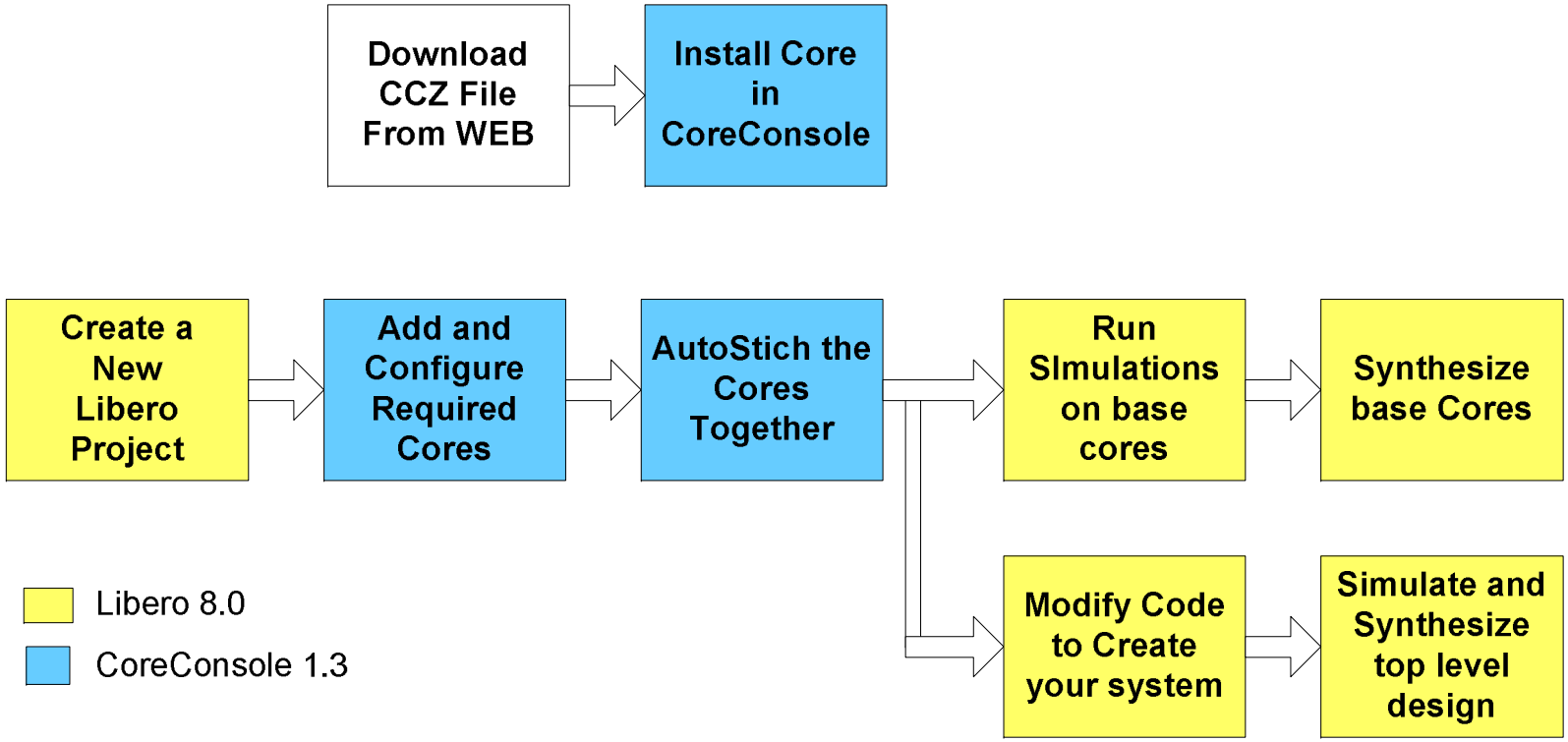
CoreConsole

CoreConsole IP Deployment Platform



- System-level Design Builder
  - Fast assembly and configuration of user designs
  - Easy to use graphical user interface
- Incorporated IP Delivery
  - Users can access IP modules in the IP Vault
    - Some modules require a separate license
- System Output
  - Configured RTL can be either plain text or obfuscated
  - CoreMP7 and other protected IP output as a blackbox
  - Output includes
    - System and IP test benches
    - Memory map generation with software driver export

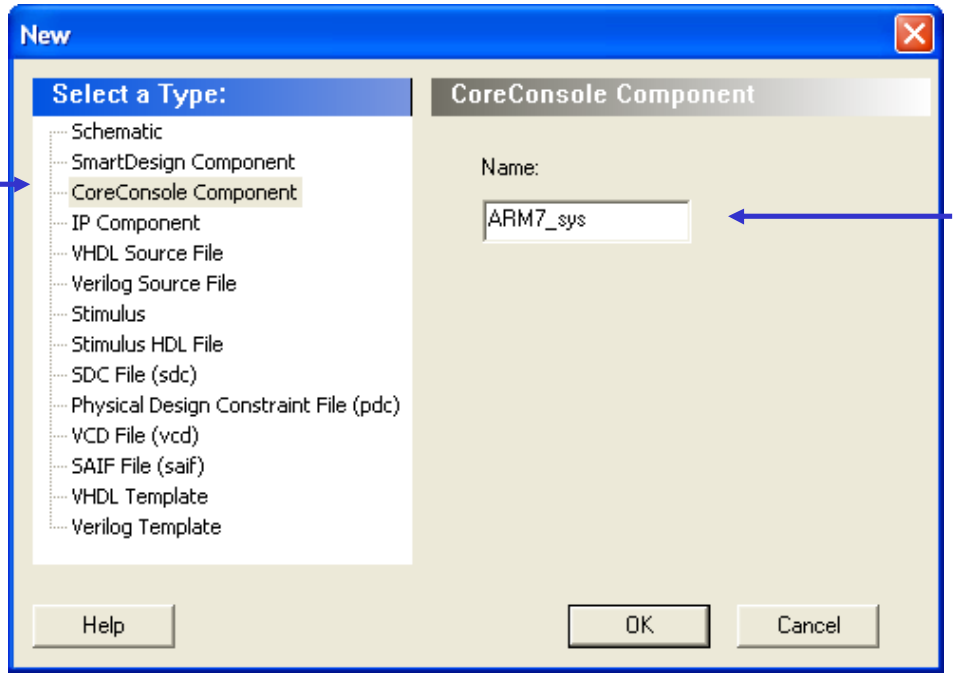
CoreConsole/Libero IDE Design Flow



Opening CoreConsole

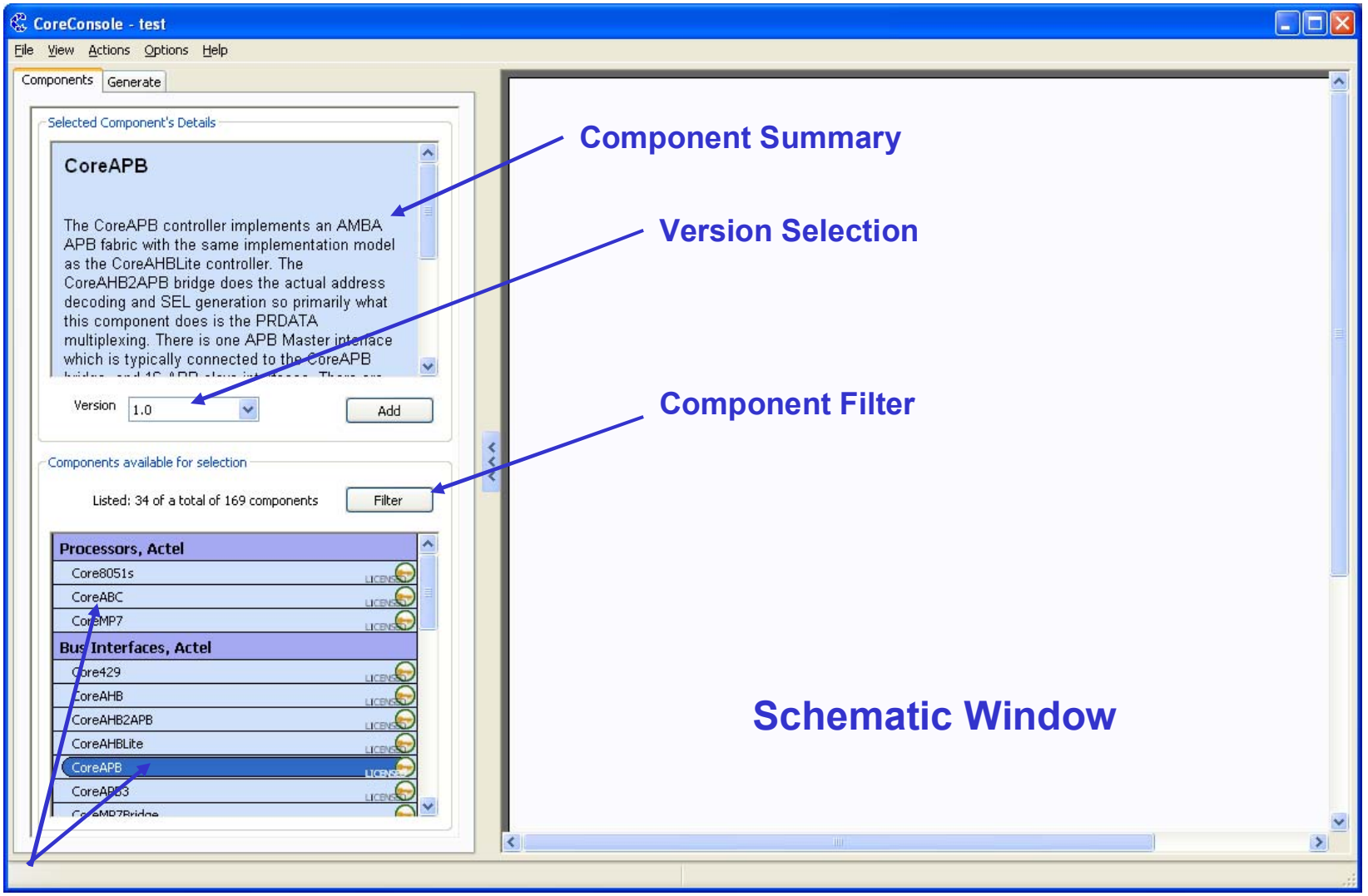
- Click CoreConsole Button in Project Flow Window
- Add and Stitch Components
- Generate IP

Select CoreConsole



Enter name

CoreConsole v1.4
User Interface

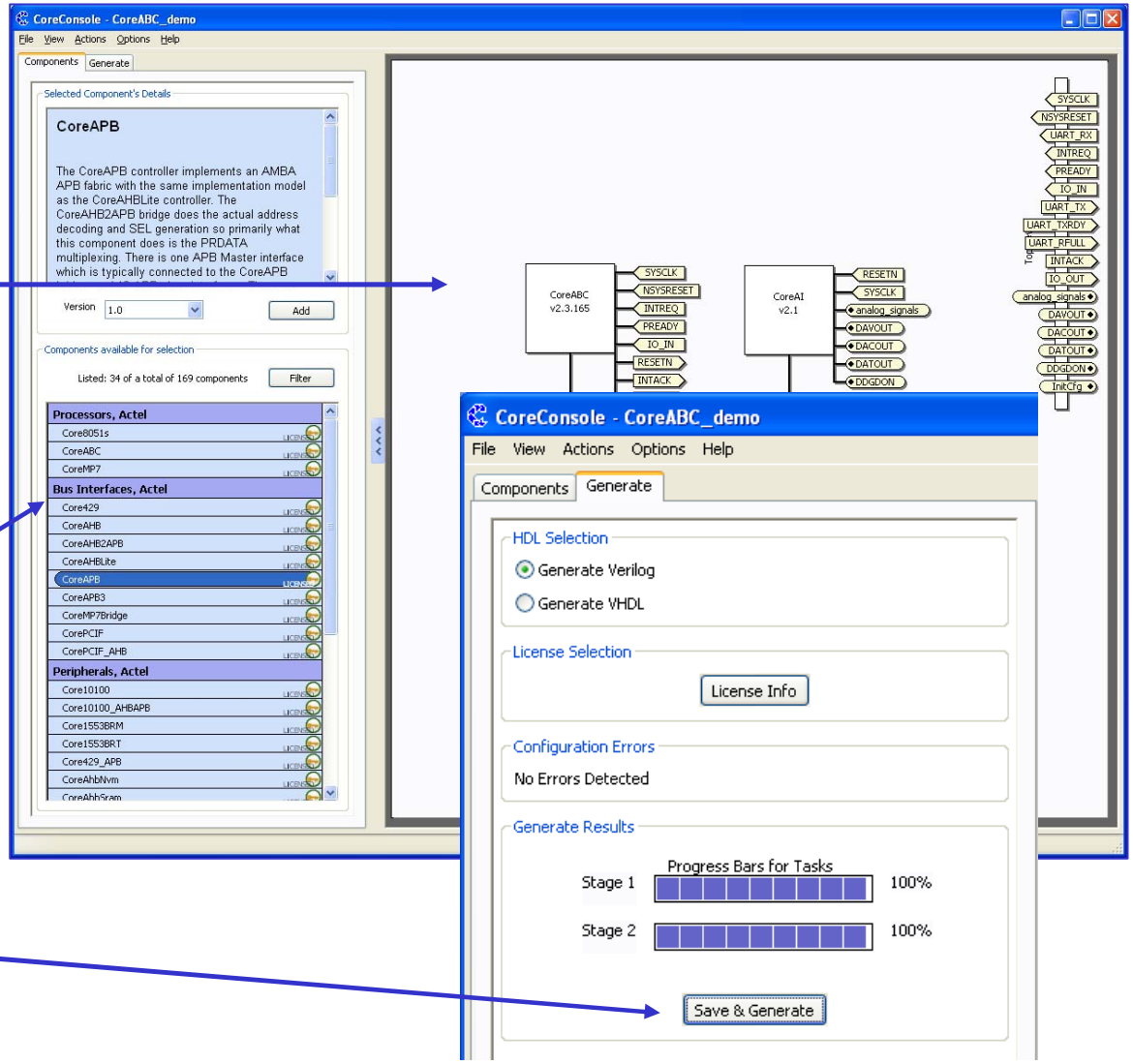


CoreConsole v1.4
GUI Tabs

Design
Configure and Stitch Each Component in Design (Done Directly in Schematic Window)

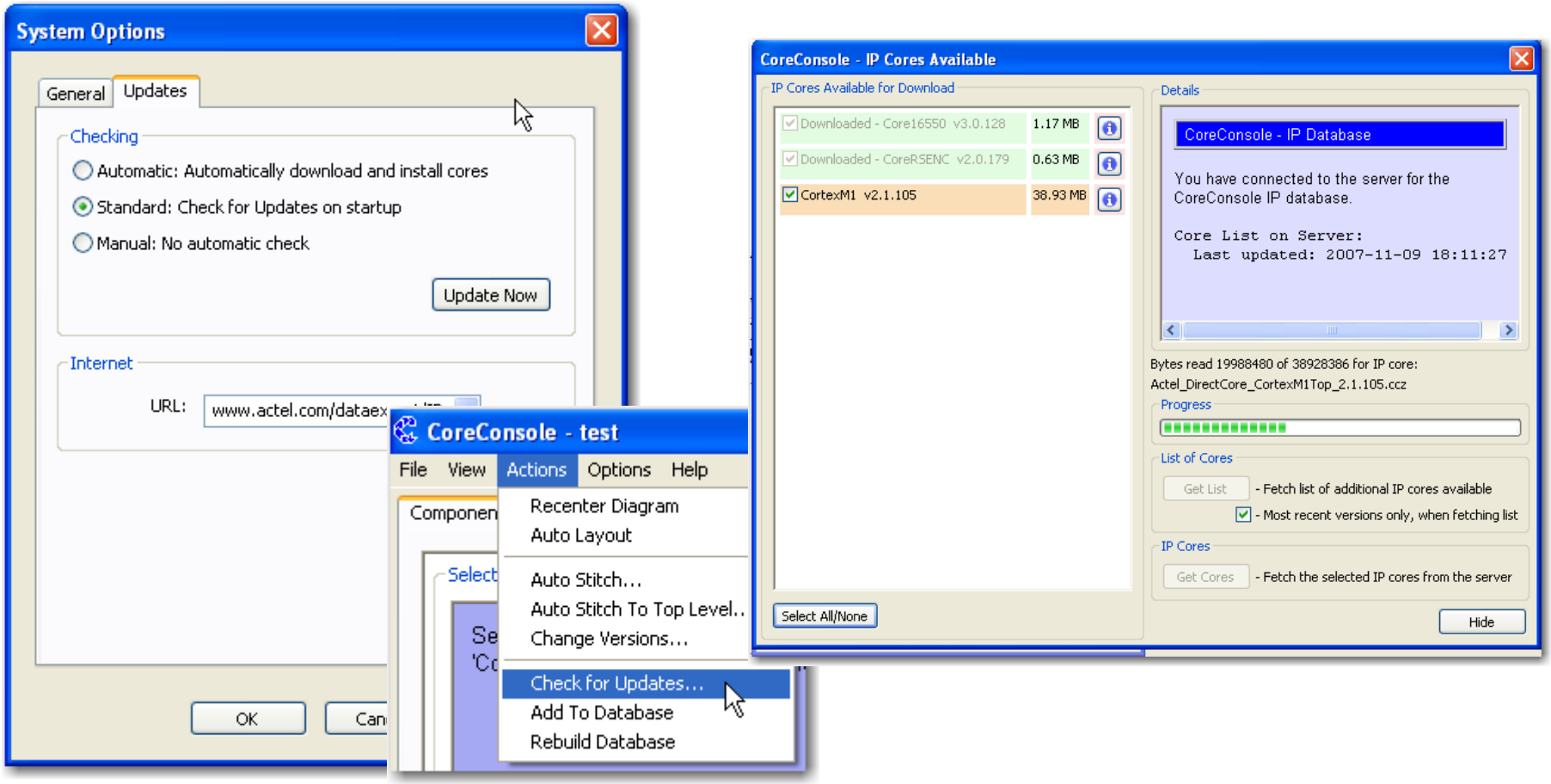
Add Components
List of Available IP, Click to Add to Design

Generate
Generate Design RTL and Other Items



CoreConsole v1.4

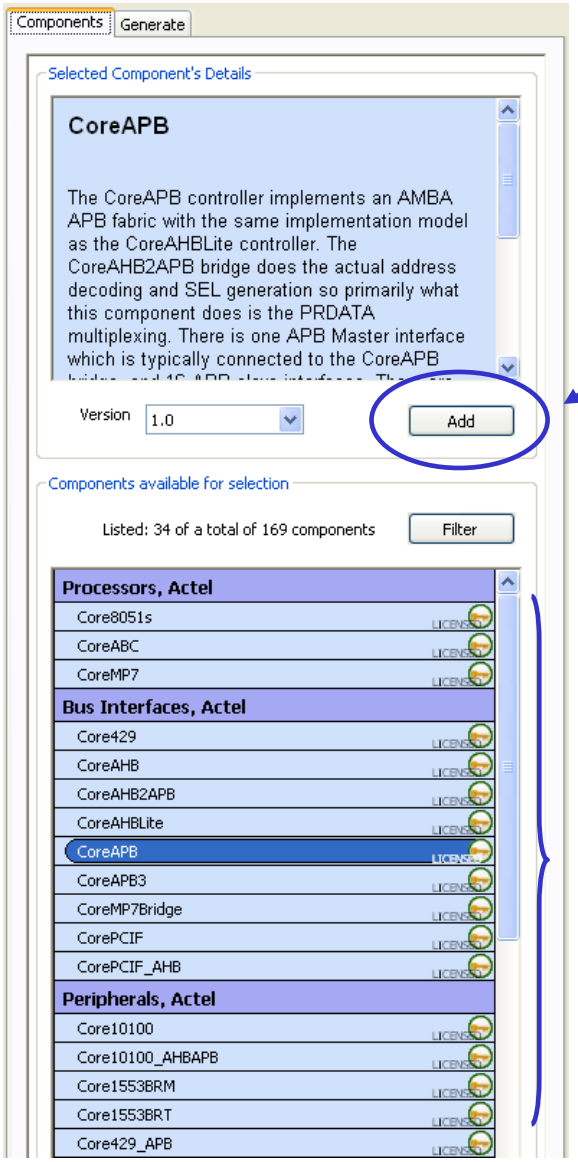
- Automatic Detection or Manual Check For New/Updated Cores



CoreConsole v1.4

Add Components

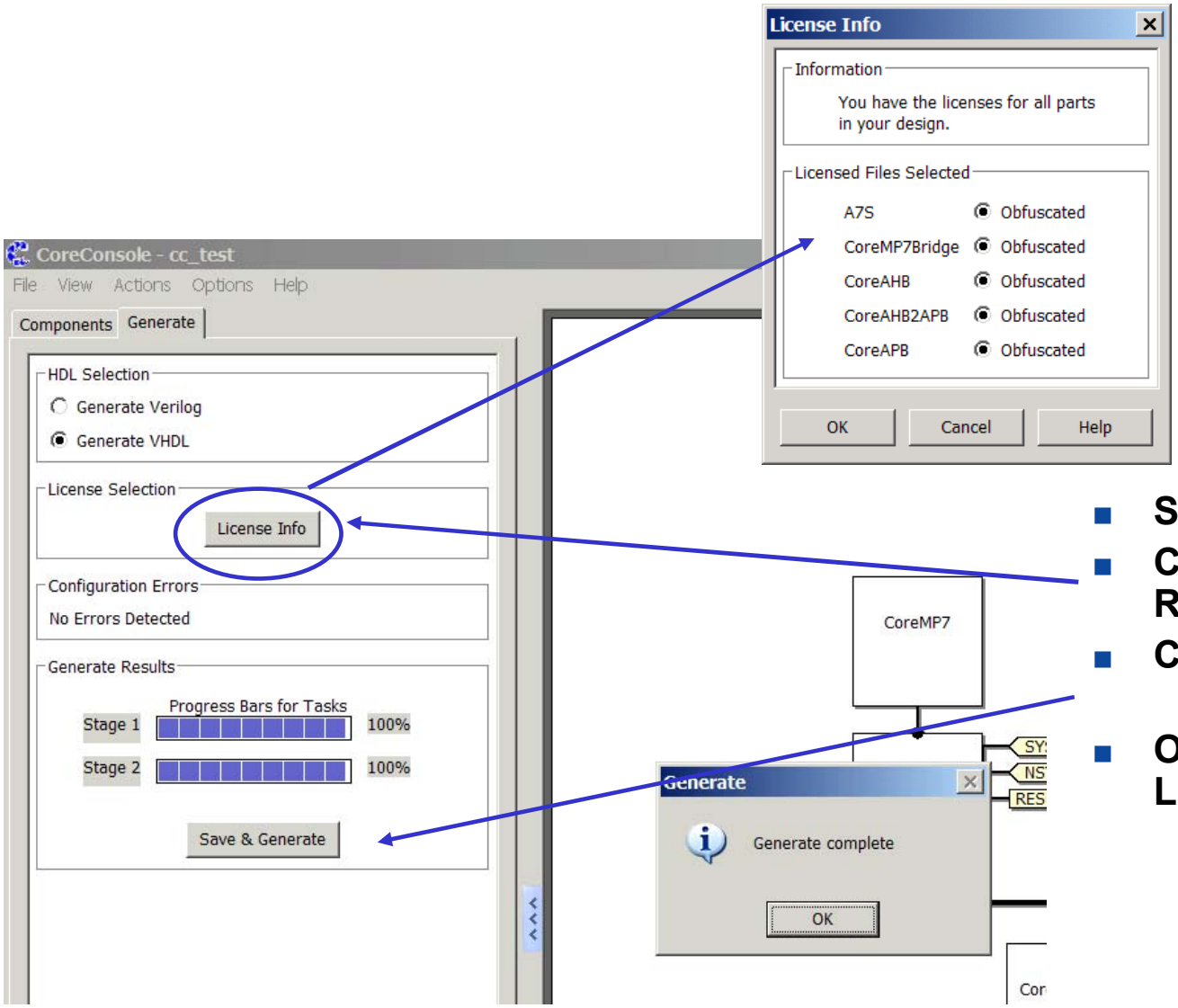
- **Components Tab Lists Available Components**
- **Push Button to Add Component**
- **Multiple Instances of the Same Core Are Given Unique Names in Schematic**



Push button to ADD

List of available cores

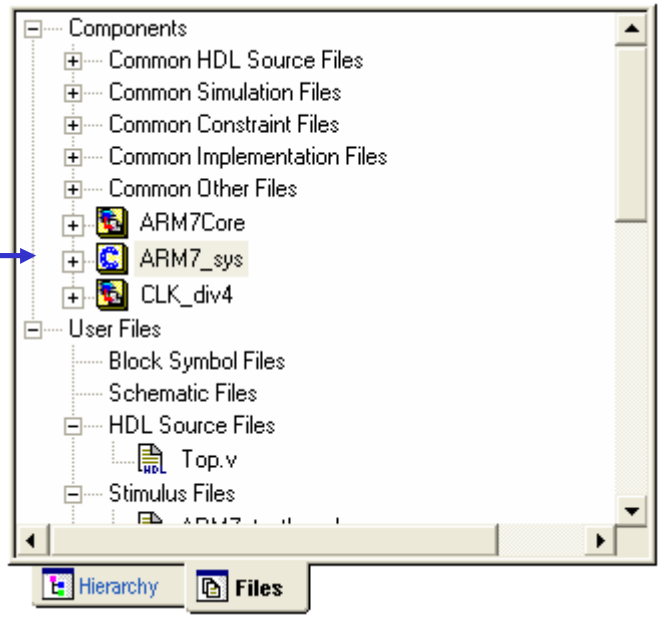
Generate



- **Select VHDL or Verilog**
- **Check License Info if Required**
- **Click *Save & Generate***
  - It May Take a Few Minutes!
- **Output Project Appears on Libero Files tab**

CoreConsole Project in Libero

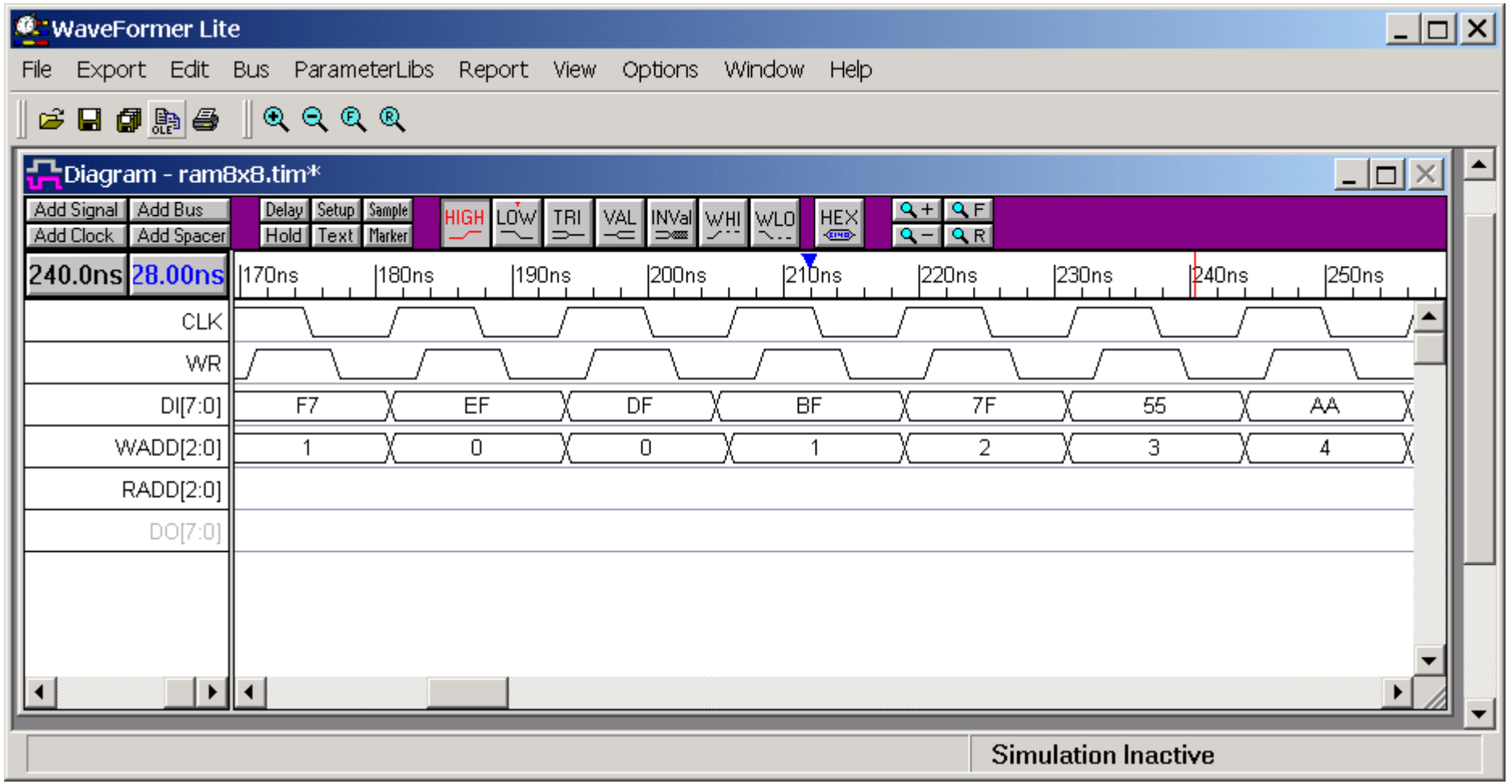
CoreConsole Project



WaveFormer Lite

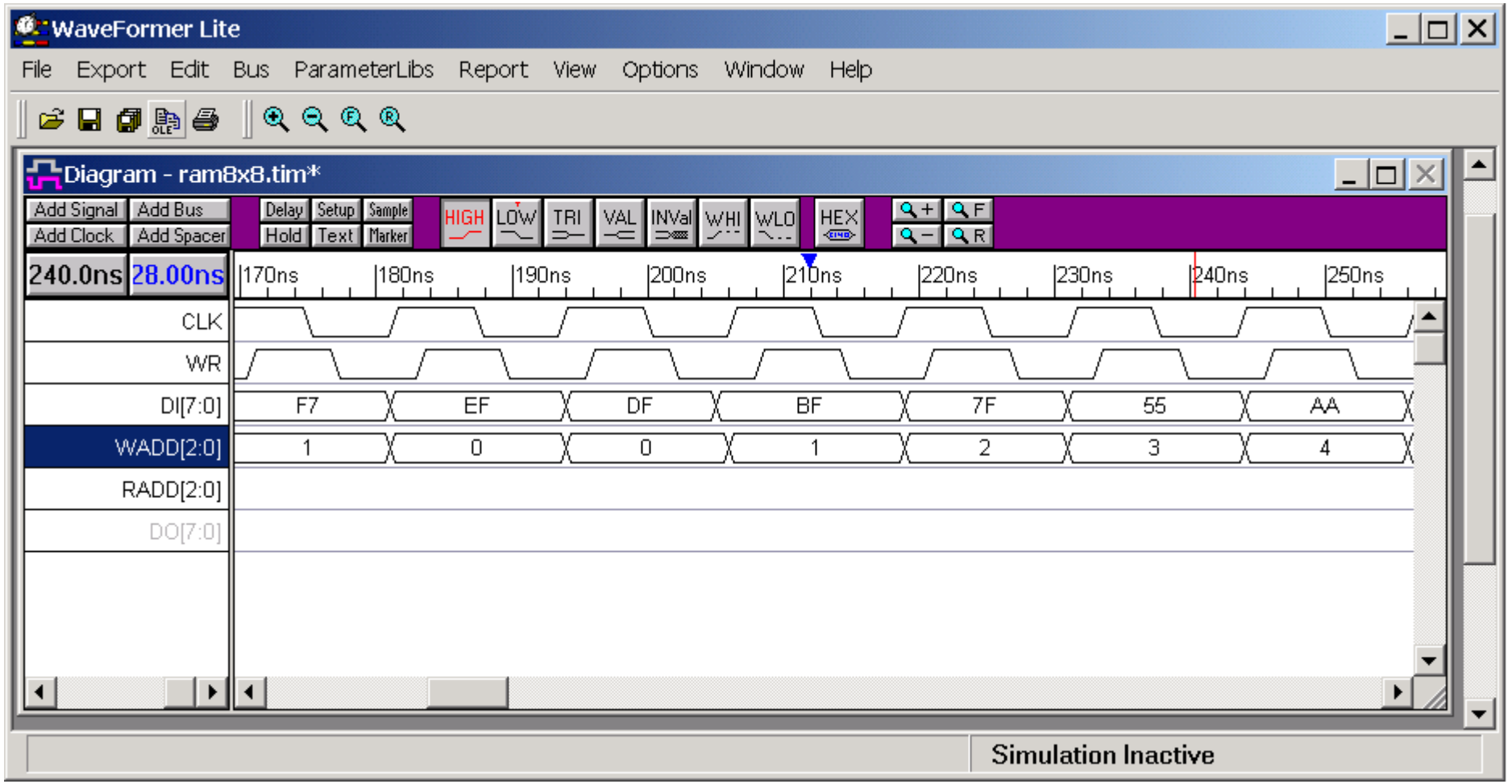
Copying to a Different Signal

- Signal Can Be Copied



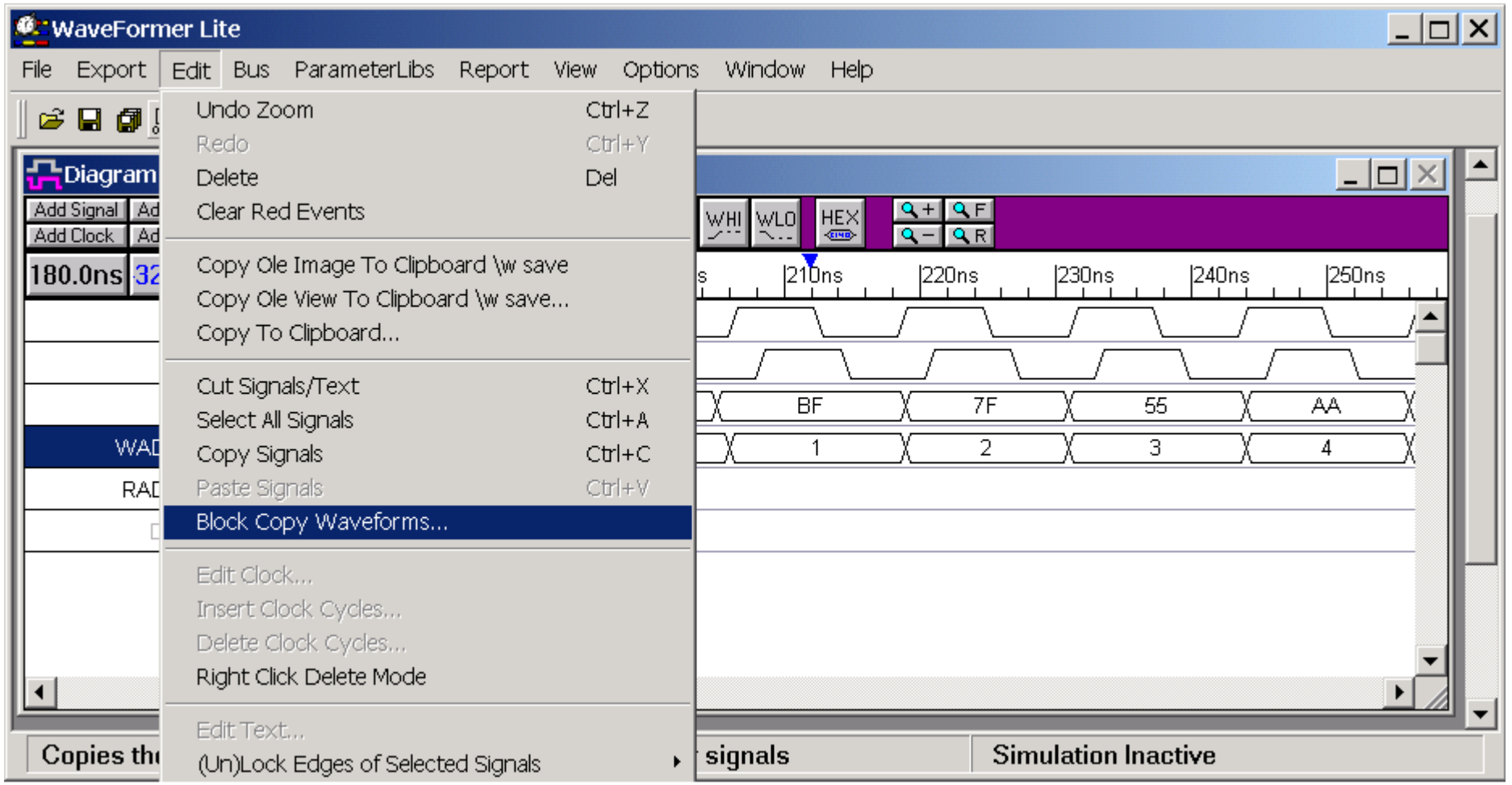
Copying to a Different Signal

- Signal Can Be Copied



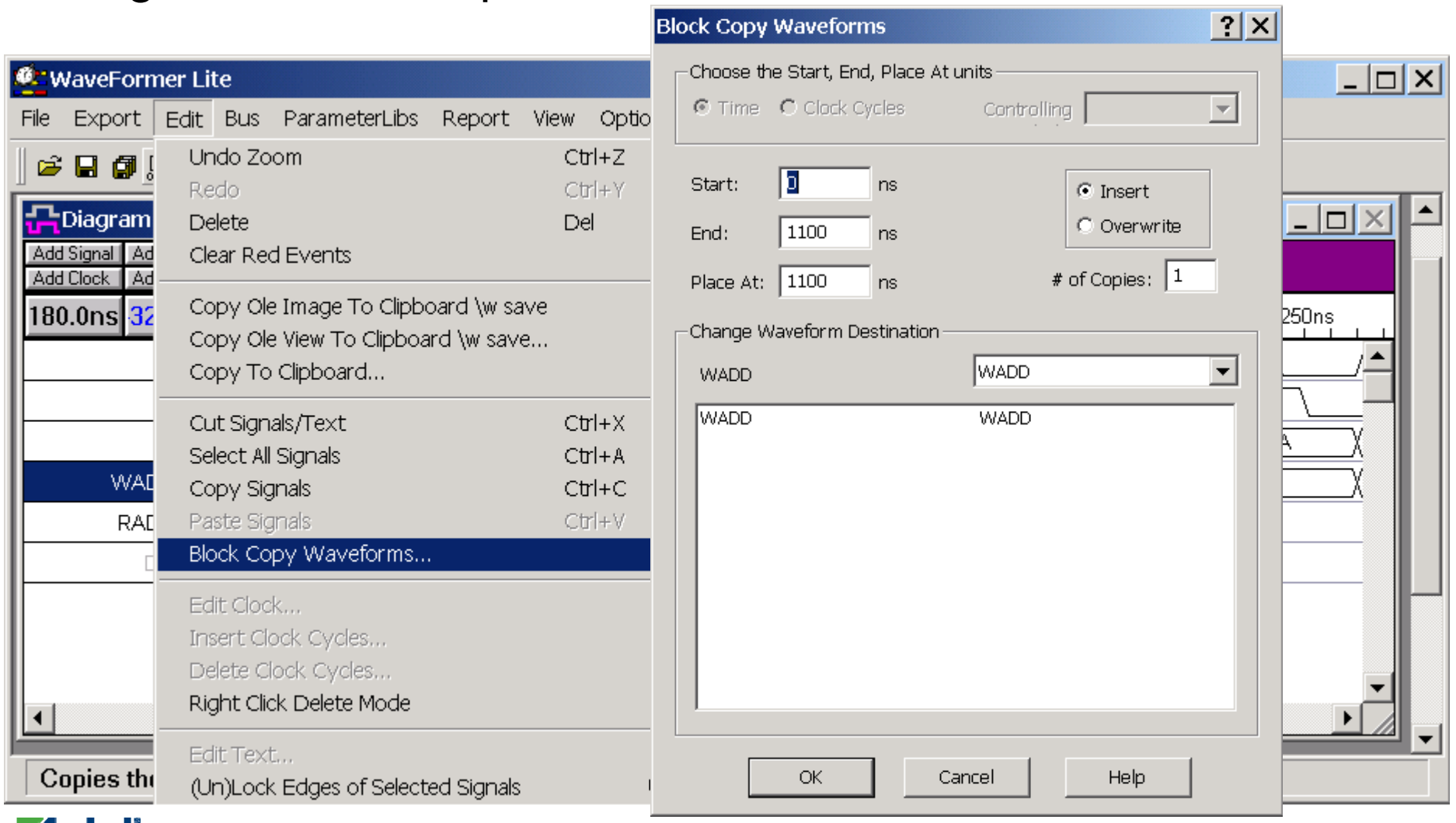
Copying to a Different Signal

- Signal Can Be Copied



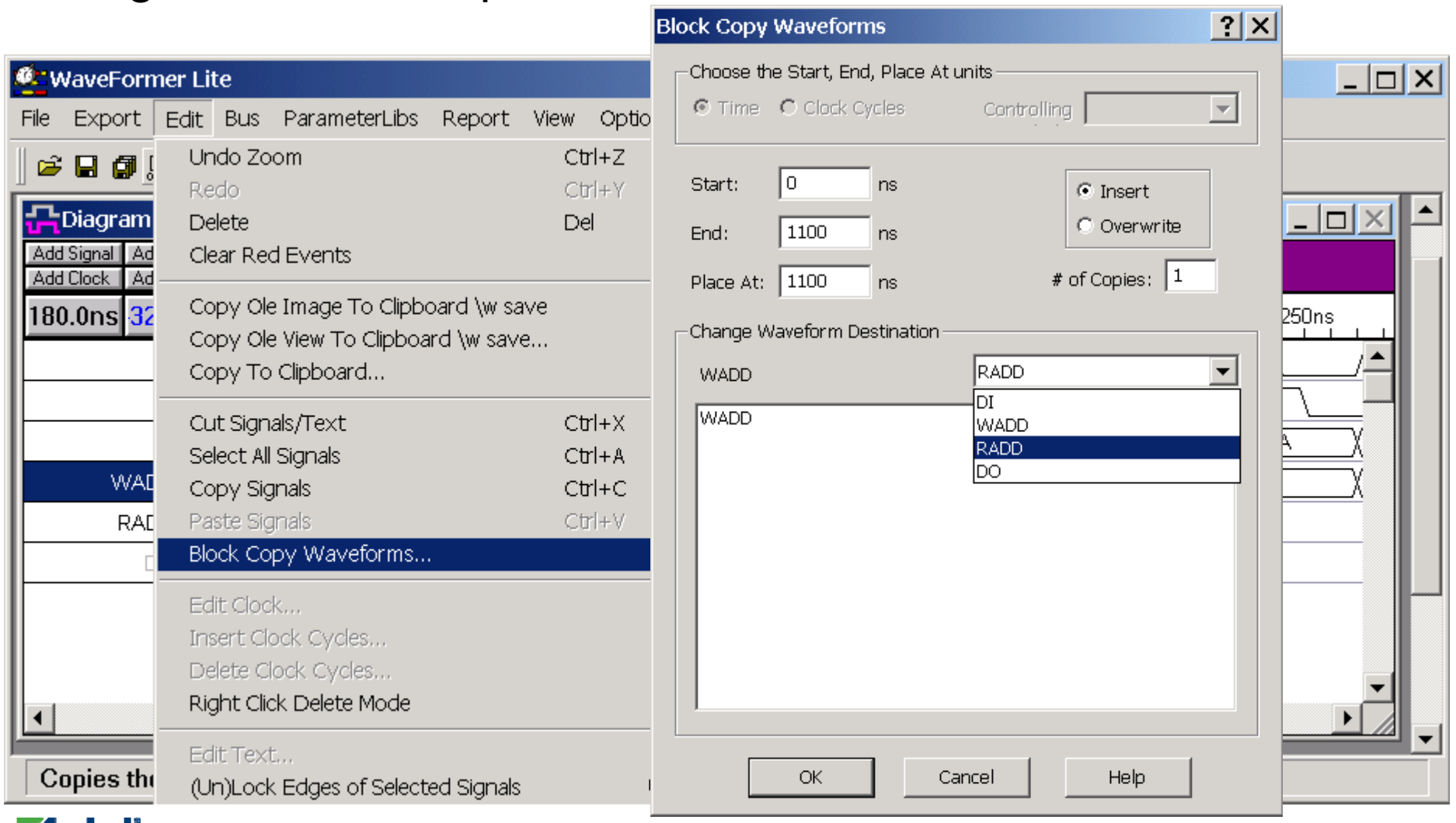
Copying to a Different Signal

- Signal Can Be Copied



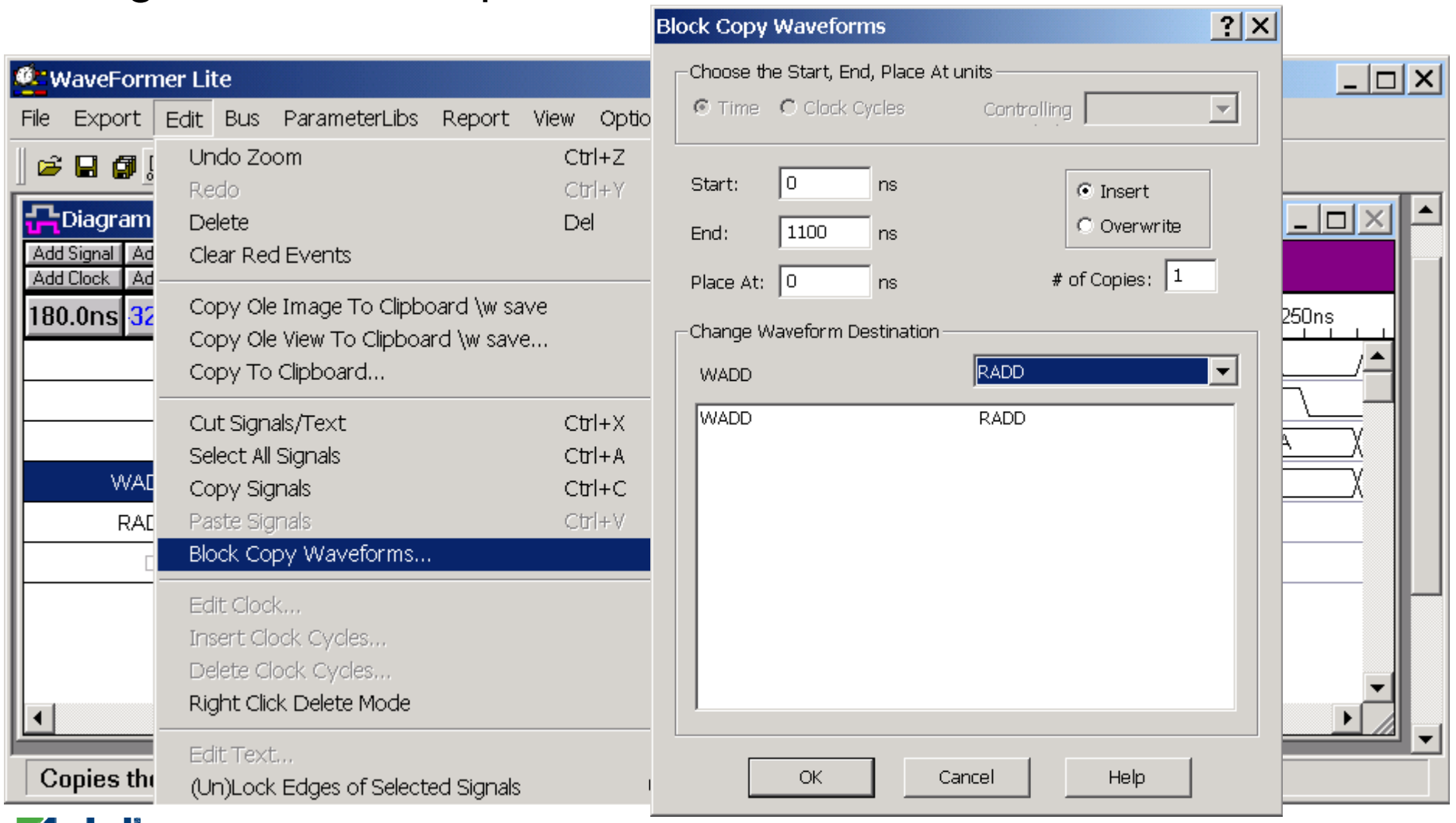
Copying to a Different Signal

- Signal Can Be Copied



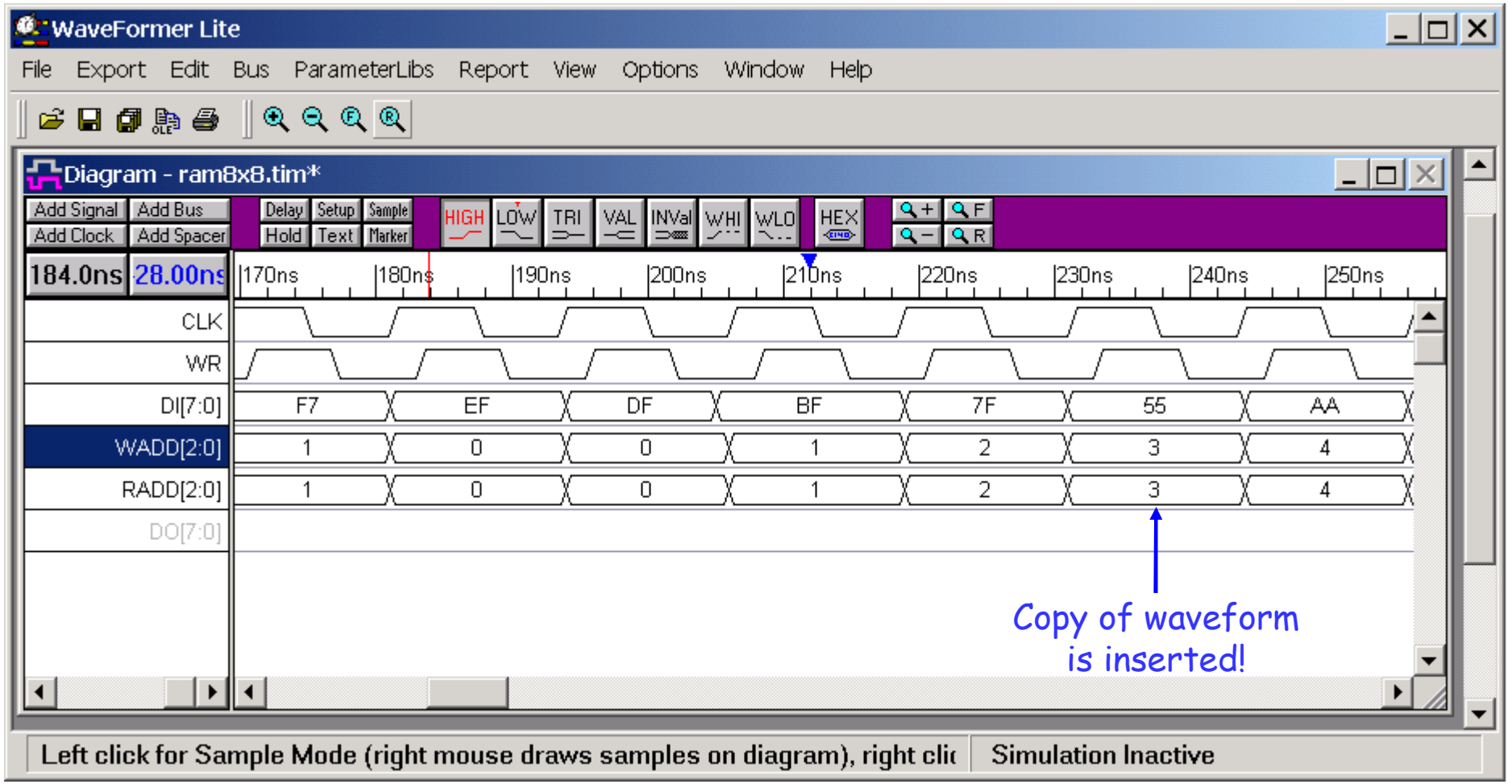
Copying to a Different Signal

- Signal Can Be Copied



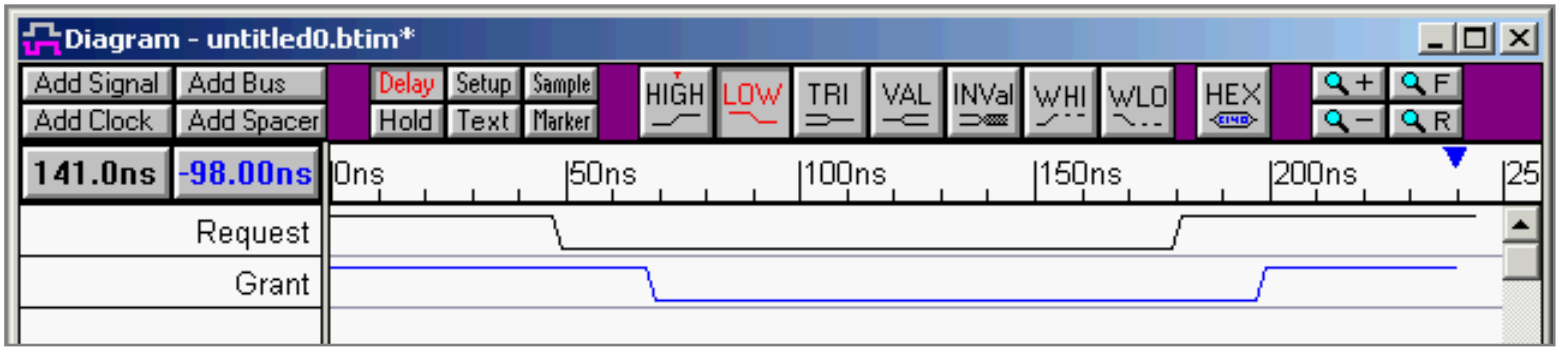
Copying to a Different Signal

- Signal Can Be Copied



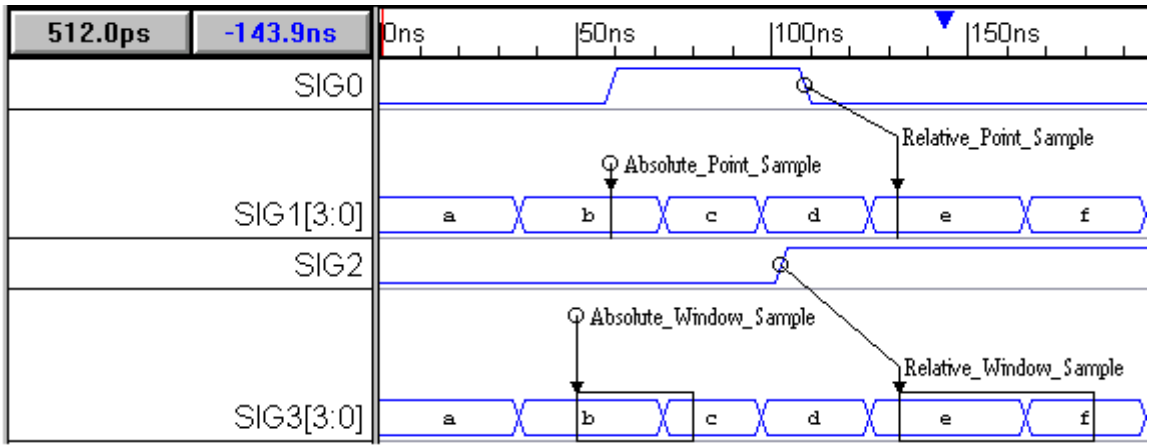
Reactive Test Bench:
Stimulus and Expected Response

- Draw stimulus waveforms on the input ports of the model under test.
- Draw expected response waveforms on the output ports of the model under test



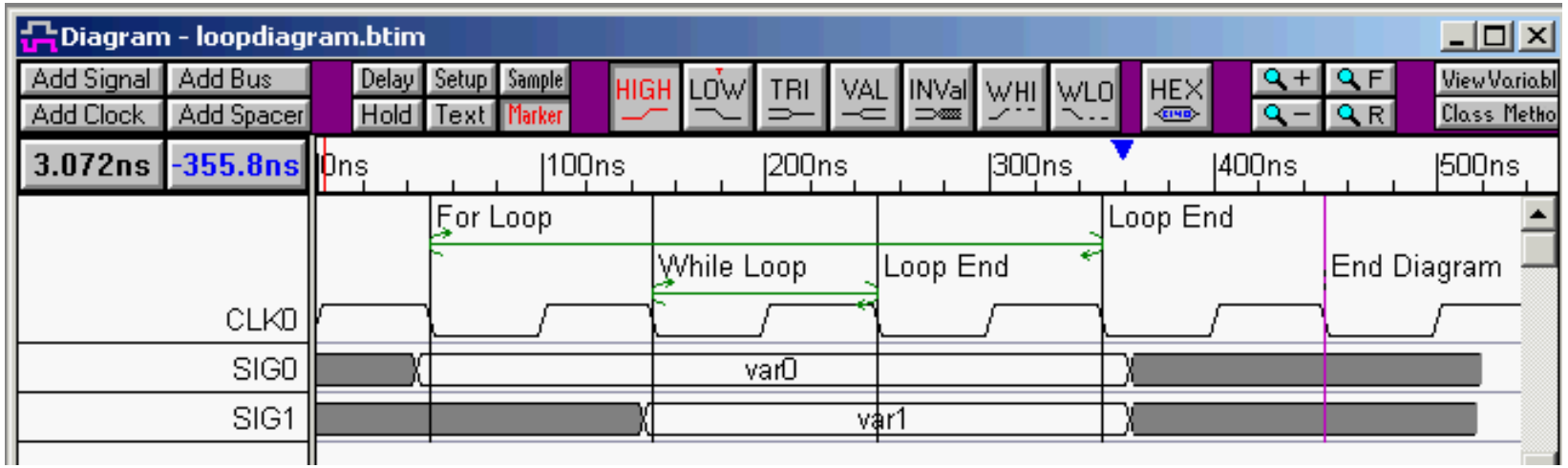
Reactive Test Bench:
Samples

- Samples Verify MUT Output
  - Sample constructs can monitor and perform actions based on the data sampled
  - Sample can work at a single point or over a windowed area
  - Sample can perform relative to the beginning of the transaction or relative to another event in the diagram.



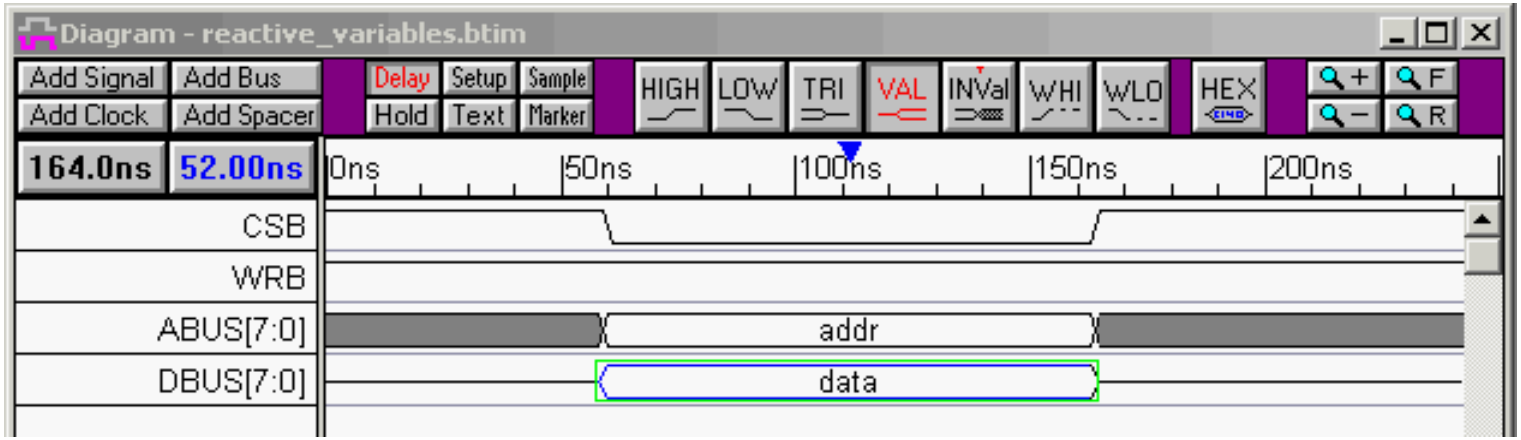
Reactive Test Bench:
Control & Looping

- Markers used for Control & Looping Sections of Transactions
  - Specify the end of the transaction
  - Create loops using for, while, and repeat loop markers
  - Insert HDL code



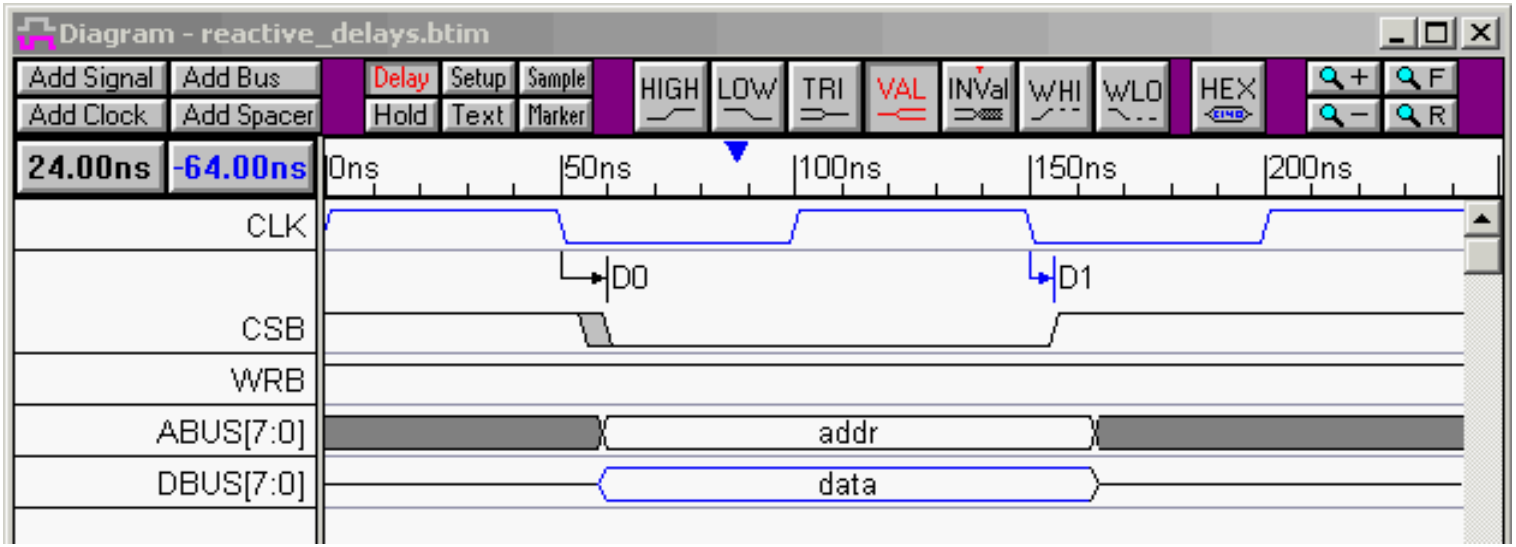
Reactive Test Bench:
Variables

- Variables Parameterize State Values
  - Variables can drive values on stimulus waveforms
  - Variables can store values on expected waveforms
  - Waveform states can be expressed as conditional expressions using variables



Reactive Test Bench:
Delays

- Delays Parameterize Time Values
  - Delays represent the time between two edges in the diagram
    - Specify min and max values
  - Delay values can be time or cycle-based
  - Conditionally control when edges occur



Reactive Test Bench:
Help Resources

- Online Manual:
  - Choose Reactive Test Bench Generation Under the Help menu
- PDF Manual:
  - Reactive_testbench_Generation_Option.pdf under the Help folder in the SynaptiCAD install directory
- SynaptiCAD's website: www.syncad.com

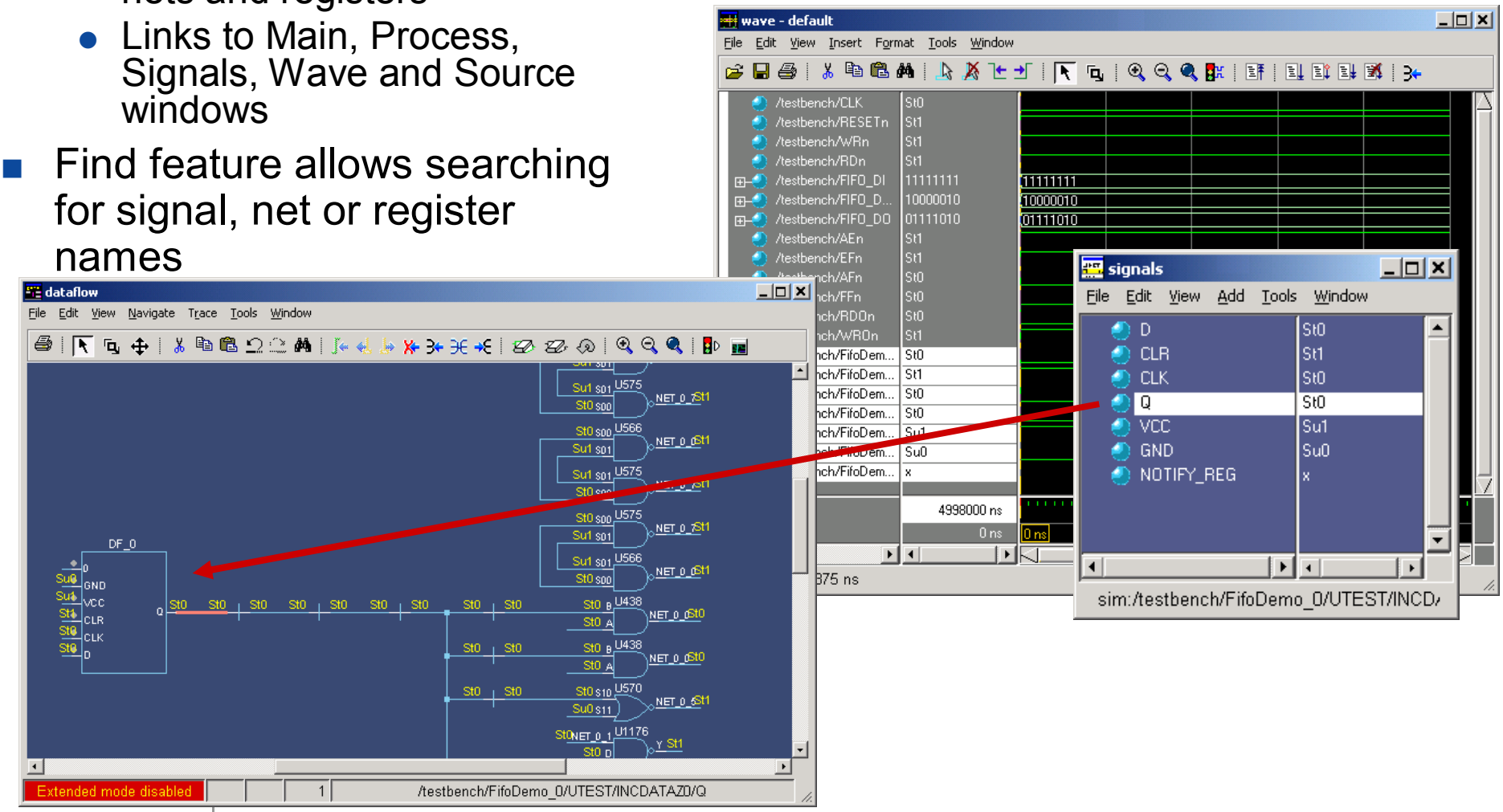
ModelSim AE

Dataflow Window

Explore physical connectivity of design

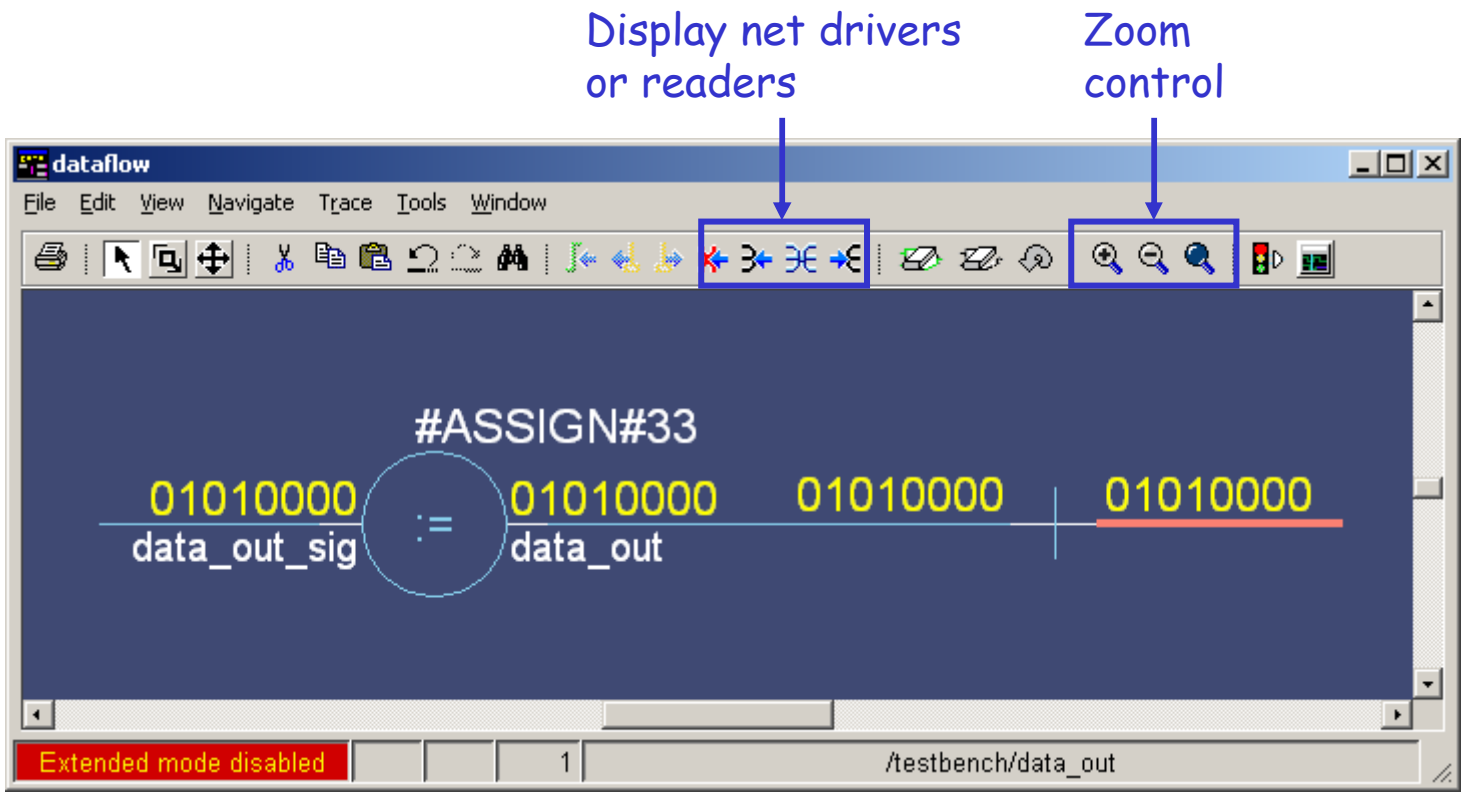
- Displays processes, signals, nets and registers
- Links to Main, Process, Signals, Wave and Source windows

- Find feature allows searching for signal, net or register names



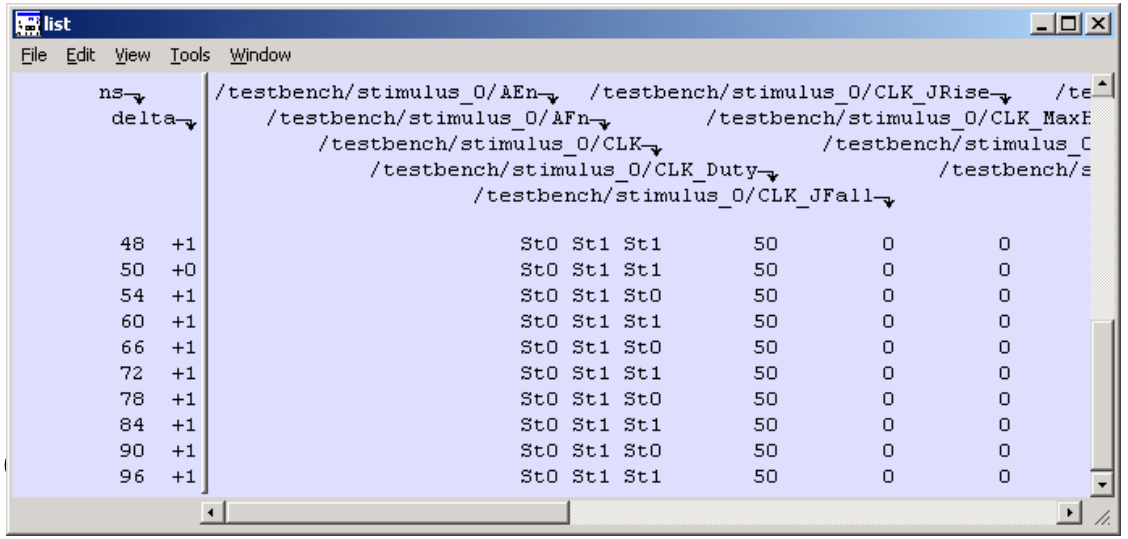
Dataflow Window
ModelSim AE Simulator

- The ModelSim AE simulator has a limited Dataflow functionality
  - Only one process and it's attached signals or one signal and it's attached processes are displayed

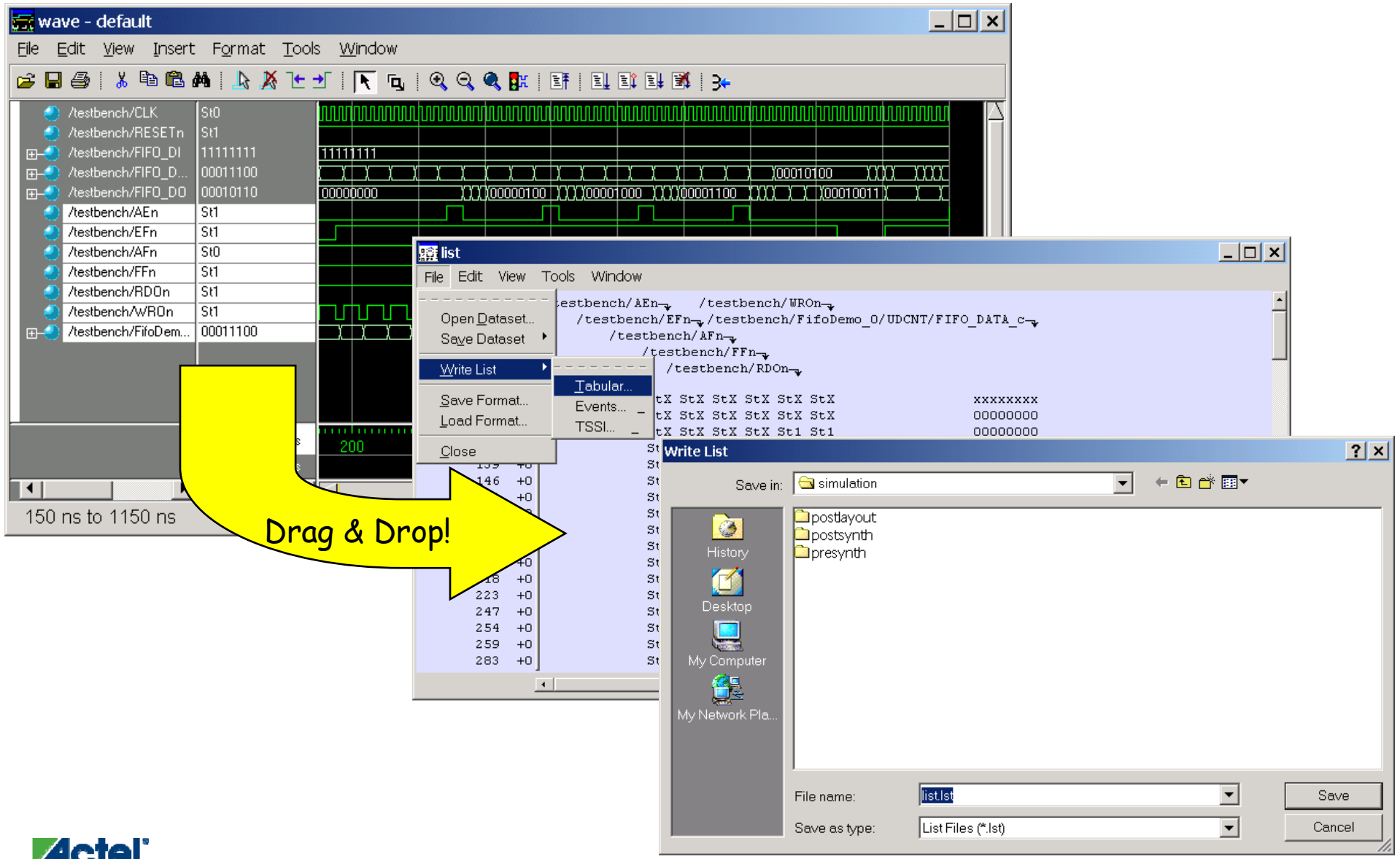


List Window

- Simulation Results in Tabular Format
  - VHDL - Signals and Process Variables
  - Verilog - Nets and Registers
- “Drag & Drop”
- Find Function
- Trigger / Strobe Properties
- Write List
  - Tabular
  - Event
  - TSSI
- Markers - Add, Delete or G



Saving Tabular Output



Synplify Attributes and Directives

syn_keep
Directive

- Maintains Net Throughout Synthesis
  - During Synthesis, Nets May Not Be Maintained in order to Create Optimized Circuit
- Applied to **`wires`** in Verilog and **`signals`** in VHDL
- Nets Can Be Preserved to
  - Probe their Values during Simulation
  - Prevent Certain Optimizations, such as Clock Enable Optimization

Example:

```
signal q_tmp : std_logic;
 attribute syn_keep : boolean;
 attribute syn_keep of q_tmp : signal is true;
```

`alspreserve`

Actel-Specific Attribute

- Keeps Net from Being Collapsed in Designer (Back-end) Tools
  - Must also Add **`syn_keep`** to Ensure Synplicity Retains Net
  - Synplicity Adds **`alspreserve`** Attribute to EDIF Netlist

alspreserve

Syntax

- Verilog Syntax

```
object /* synthesis alspreserve = 1 */ ;
```

  - Example:

```
module foo ( in, out);
  input [6:0] in;    output    out;    wire      out;
  wire      or_out1    /* synthesis syn_keep=1 alspreserve=1 */;
  wire      and_out1;    wire      and_out2;
  wire      and_out3  /* synthesis syn_keep=1 alspreserve=1 */;
```

- VHDL Syntax

```
attribute alspreserve of object : signal is true ;
```

  - Example:

```
architecture comb of foo is
    signal and_out1, and_out2, and_out3, or_out1 : std_logic;

    attribute syn_keep of and_out3 : signal is true;
    attribute syn_keep of or_out1  : signal is true;
    attribute alspreserve: boolean;
    attribute alspreserve of and_out3 :signal is true;
    attribute alspreserve of or_out1 : signal is true;
```

syn_encoding

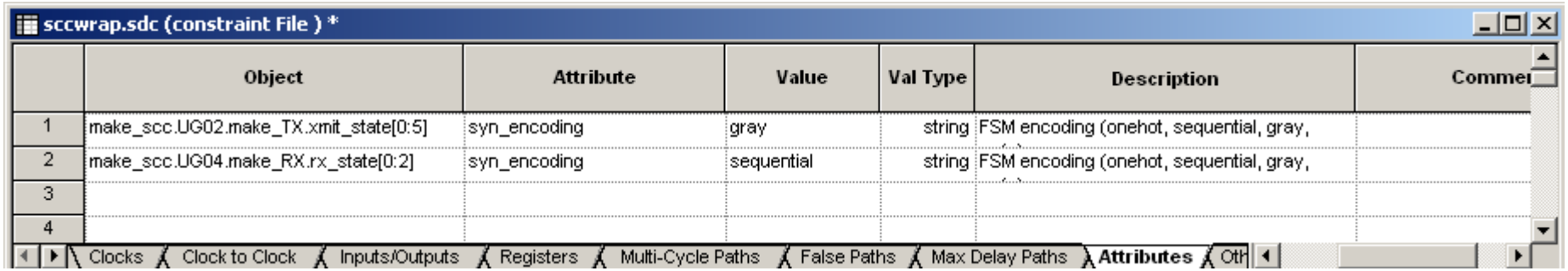
Attribute

- Sets Encoding Style for State Machines
  - Overrides Default Style
- Default Style - Compiler Selects Encoding Style Based on Number of States as Follows:
  - 1 - 4 States: Sequential
  - 5 - 24 States: One-hot
  - > 24 States: Gray
- **syn_encoding** Can Have the Following Values:
  - onehot
  - gray
  - sequential
  - safe

syn_encoding

Attribute Usage

- SCOPE Constraint Editor Usage
  - Select Individual State Machines and Assign Attribute



syn_encoding

Attribute Usage (cont.)

- Verilog Syntax

```
object /* synthesis syn_encoding = "value" */ ;
```

  - Example:

```
module prep3 (CLK, RST, IN, OUT);
input CLK, RST;
input [7:0] IN;
output [7:0] OUT;
reg [7:0] OUT;
reg [7:0] current_state /* synthesis syn_encoding="gray" */;
// Other code
```

- VHDL Syntax

```
attribute syn_encoding of object : object_type is "string" ;
```

  - Example:

```
library synplify;
use synplify.attributes.all;
package my_states is
type state is (st0, st1, st2, st3, st4, st5);
signal s1 : state;
attribute syn_encoding of s1 : signal is "gray";
end my_states;
```

syn_maxfan

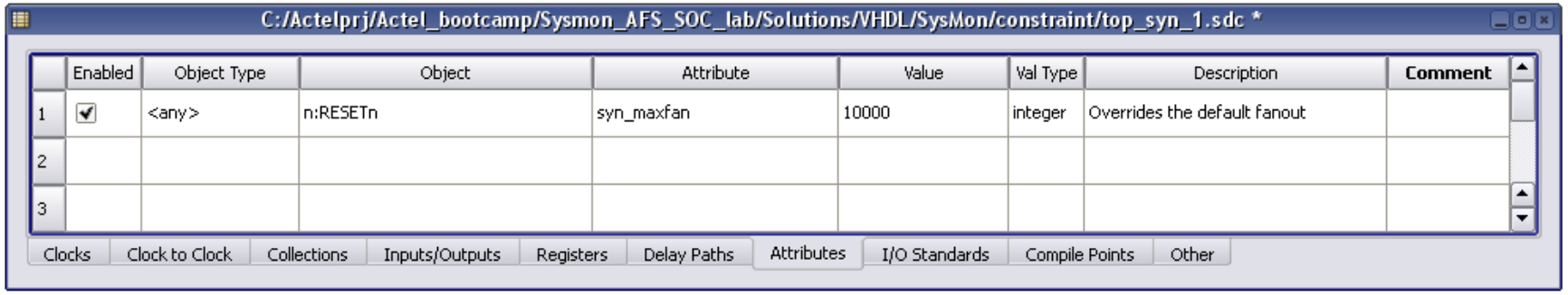
Attribute

- Controls Maximum Fanout of Instance, Net, or Port
- Limit Specified by this Attribute May Be Treated as Hard or Soft Depending on Where It Was Specified
  - Soft Limit May Not Be Honored if it Degrades Performance
- You Can Apply **syn_maxfan** Attribute to Module, Register, Instance, Port, or Net
    - For ProASIC and APA Designs Only – You Can also Apply to Module or Entity

syn_maxfan

Usage

- SCOPE Constraint Editor Usage



- SDC File Syntax

  `define_attribute { object } syn_maxfan { integer }`

  - Example – Limit Fanout for Signal **`clk`** to 200:

```
. . .
define_attribute {clk} syn_maxfan {200}
. . .
```

syn_maxfan

Syntax

- Verilog Syntax

```
object /* synthesis syn_maxfan = "value" */ ;
```

  - Example:

```
module test (registered_data_out, clock, reset, data_in);
output [31:0] registered_data_out;  input clock, reset;
input [31:0] data_in;
input reset /* synthesis syn_maxfan=1000 */;
// Other code
```

- VHDL Syntax

```
attribute syn_maxfan of object: object_type is "value“;
```

  - Example:

```
entity test is
        port(clock, reset : in bit;
             data_in : in bit_vector(31 downto 0);
             registered_data_out: out bit_vector(31 downto 0))
attribute syn_maxfan : integer;
attribute syn_maxfan of reset : signal is 1000;
-- Other code
```

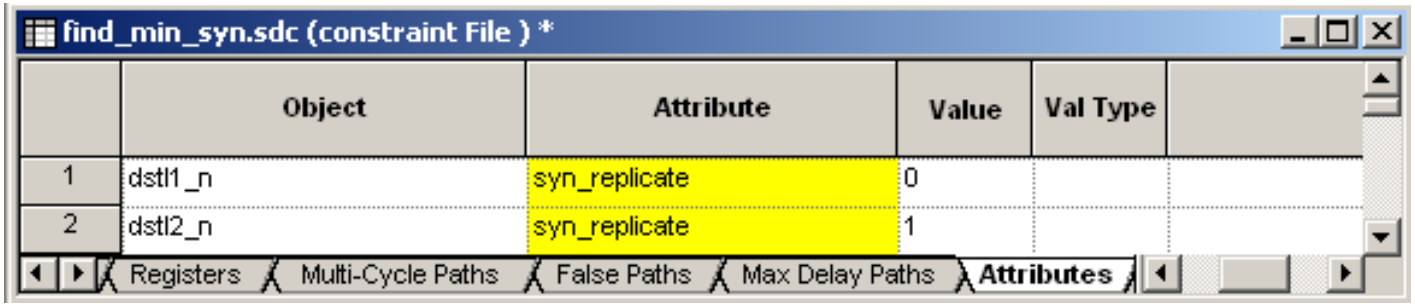
`syn_replicate`
Attribute

- Prevents Replication of Register
  - **`assign syn_replicate = 0`** Turns Off Register Replication
  - Cannot Force Tool to Replicate
  - Works along with **`max_fanout`** Value
  - Only Supported on Individual Register
- When Will Synplify Replicate or Buffer?
  - Generally Flip-flops Are Replicated to Achieve Fan-out Control
  - Combinatorial Cells May be Replicated to Control Fanout

syn_replicate

Usage

- SCOPE Constraint Editor Usage



- SDC File Syntax

```
define_global_attribute syn_replicate = { 1 | 0 }
```

Example - Disables All Replication in Design:

1 enables replication
0 disables replication

```
 .   .   .
define_global_attribute syn_replicate {0}
 .   .   .
```

syn_replicate
Usage (cont.)

- Verilog Syntax
  - **object /* synthesis syn_replicate = 1 | 0 */;**
  - Example:

0 disables replication

```
module norep (Reset, Clk, Drive, OK, ADPad, IPad, ADOut);

 .   .   .
reg [31:0] IPad;
reg DriveA /* synthesis syn_replicate = 0 */;
assign ADPad = DriveA ? ADOut : 32'bz;
always @(posedge Clk or negedge Reset)
        if (!Reset) begin
           DriveA <= 0;
           IPad   <= 0;  end
        else        begin
           DriveA <= Drive & OK;
           IPad   <= ADPad;  end
endmodule
```

syn_replicate

Usage (cont.)

- VHDL Syntax
  - attribute syn_replicate of object : object_type is true | false ;
  - Example:

false disables replication

```
entity norep is port (
                  Reset : in std_logic;
                  Clk : in std_logic;
                  Drive : in std_logic;
                  OK : in std_logic;
                  ADPad : inout std_logic_vector (31 downto 0);
                  IPad : out std_logic_vector (31 downto 0);
                  ADOut : in std_logic_vector (31 downto 0));
end norep;
architecture archnorep of norep is
signal DriveA : std_logic;
attribute syn_replicate : boolean;
attribute syn_replicate of DriveA : signal is false;

begin

-- Other code
```

syn_sharing
Directive

- Enables/Disables Resource Sharing of Operators inside Module during Synthesis
  - By Default, Directive Is Enabled (Value 1 for Verilog, true for VHDL).
  - If Resource Sharing Check Box in Project View is Disabled, You Can Still Enable Resource Sharing Using syn_sharing Directive

syn_sharing

Usage

- Verilog Syntax
  - object /* synthesis syn_sharing = 1 | 0 */ ;
  - Example:

```
module my_design(out,in,clk_in) /* synthesis syn_sharing=0 */;


// Other code
```

- VHDL Syntax
  - attribute syn_sharing of object : object_type is " true | false";
    object can be architecture name
  - Example:

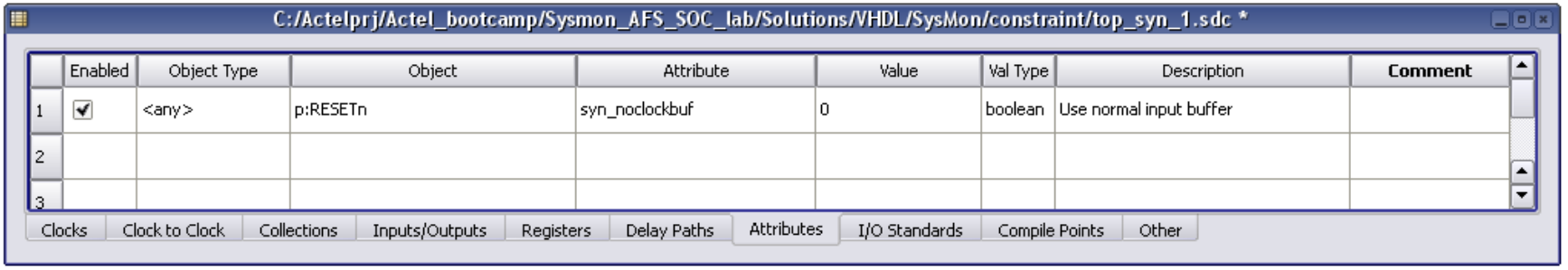
```
entity alu is
        port ( a, b : in std_logic_vector (7 downto 0);
               . . . );
end alu;
architecture behave of alu is
--  Turn on resource sharing for the architecture.
attribute syn_sharing of behave : architecture is "true";
begin
-- Other code
```

syn_noclockbuf

Synthesis Attribute

- Selects/Deselects Automatic Clock Buffering
  - Value of '1' (or Boolean TRUE) Turns OFF Automatic Clock Buffering
  - You Can Apply syn_noclockbuf Attribute to Module, Register, Instance, Port, or Net

- Synplicity SCOPE Constraint Editor Usage



syn_noclockbuf

Attribute Usage (cont.)

- Verilog Syntax
  - object /* synthesis syn_noclockbuf = "value" */ ;
  - Example:

```
module test (registered_data_out, clock, reset, data_in);
output [31:0] registered_data_out;  input clock;
input [31:0] data_in;
input reset /* synthesis syn_noclockbuf=1 */;
// Other code
```

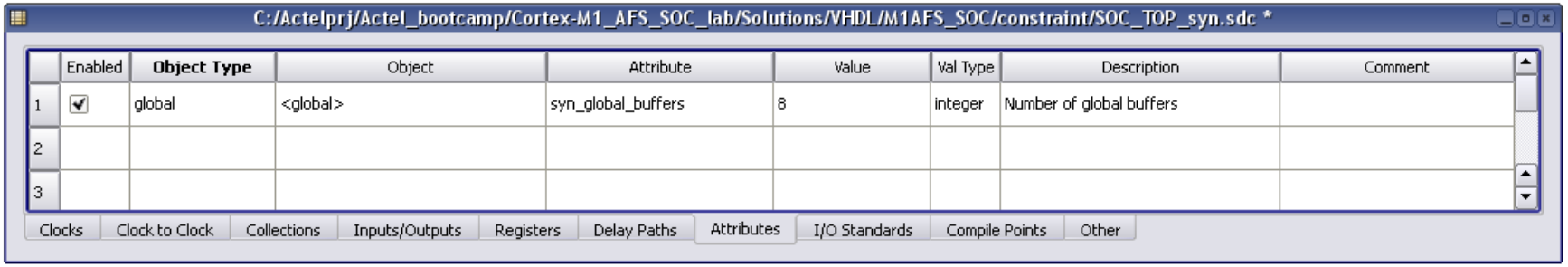
- VHDL Syntax
  - attribute syn_noclockbuf of object : object_type is "value" ;
  - Example:

```
entity test is
        port(clock, reset : in bit;
             data_in : in bit_vector(31 downto 0);
             registered_data_out: out bit_vector(31 downto 0))
attribute syn_noclockbuf : boolean;
attribute syn_noclockbuf of reset : signal is true;
-- Other code
```

syn_global_buffers

Synthesis Attribute (ProASIC3/E, Fusion, IGLOO/e)

- Specifies The Number Of Global Buffers To Be Used
  - Use This Attribute To Restrict The Number Of Global Buffer Resources Used by Synplicity
  - Attribute Is Applied Globally On The Top-level Module Or Entity
  - Value Can Be Any Integer Between 6 And 18
- Synplicity SCOPE Constraint Editor Usage



syn_global_buffers

Attribute Usage

- Verilog Syntax

```
object /* synthesis syn_global_buffers = <maximum> */;
```

  - Example:

```
module top (clk1, clk2, clk3, clk4, clk5, clk6,
clk7,clk8,clk9,  clk10, clk11, clk12, clk13, clk14, clk15,
d1, d2, d3, d4, d5, q1, q2, reset)

/* synthesis syn_global_buffers = 10 */;
```

- VHDL Syntax

```
attribute syn_global_buffers of object : object_type is
  <maximum>;
```

  - Example:

```
entity top is

   port(      . . .       );

end top;

 architecture behave of top is

attribute syn_global_buffers : integer;

attribute syn_global_buffers of behave : architecture is 10;
```

syn_ramstyle
Synthesis Attribute

- Manually Indicates How Inferred RAM is Implemented
- Values
  - **block_ram** (default) – Inferred RAM Will Use Block RAM if Possible
    - RAM Larger than One Block Will Be Cascaded
    - By Default, Software Uses Deep Block RAM Configurations instead of Wide Configurations for Better Timing Results
      - Example: 4Kx8 Built from 512x8s instead of 4Kx1s
    - Deeper RAMS Reduces Output Data Delay Timing by Reducing the MUX Logic at RAM Outputs of the RAMS
  - **registers** – Inferred RAM Will Be Implemented as Registers
  - **no_rw_check** – Implements RAM without Glue Logic
    - Glue Logic to Bypass RAM Is Added by Default because Simultaneous Read and Write at the Same Address Gives Indeterminate Data
      - Possible Simulation Mismatch
- **syn_ramstyle** Attribute Is Applied to Memory Array
- Value of **select_ram** (Indicated in Documentation) Does Not Apply to Actel RAMs