Advanced Workshop on
FPGA-based Systems-On-Chip
for Scientific Instrumentation and
Reconfigurable Computing

Vivado Design Flow for SoC

Cristian Sisterna

Universidad Nacional de San Juan

Argentina

Why Vivado Design Suite?

Larger FPGAs lead to more difficult design issues

- Users integrating more functionality into the FPGA
  - Use of multiple hard logic objects (block RAMs, GTs, DSP slices, and microprocessors, for example)
- I/O and clock planning critical to FPGA performance
- Higher routing and utilization density
- Complex timing constraints with designs that have multiple clock domains

FPGA designs are now looking like ASIC platform designs

- Assembled from IP cores—commercial or developed in-house
  - Maintaining place and route solutions is very important (this is resolved with the use of partitions)
  - Bottom-up design methodology
- Team design flows becoming a necessity

***Vivado Design Suite** provides solution to all of the above*

Vivado IDE Solution

Interactive design and analysis

- Timing analysis, connectivity, resource utilization, timing constraint analysis

RTL development and analysis

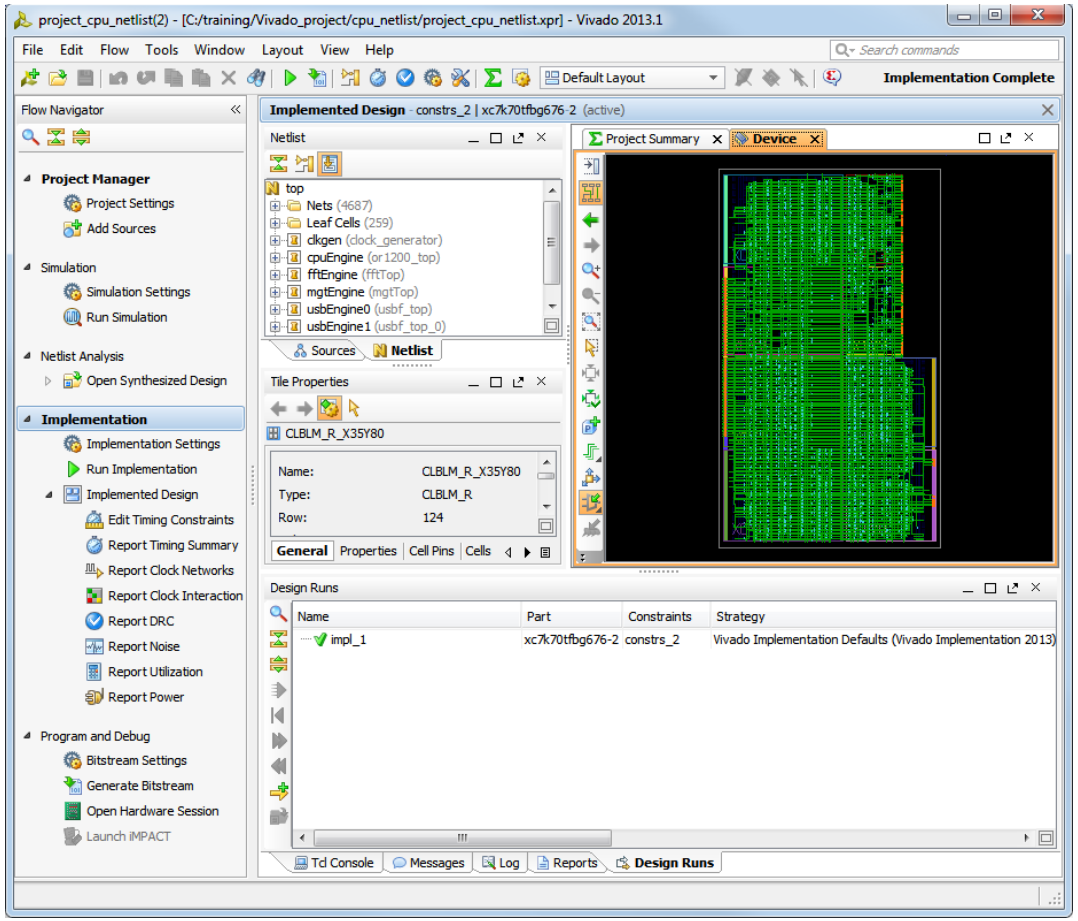
- Elaboration of HDL
- Hierarchical exploration
- Schematic generation

XSIM simulator integration

- Synthesis, implementation and simulation in one package

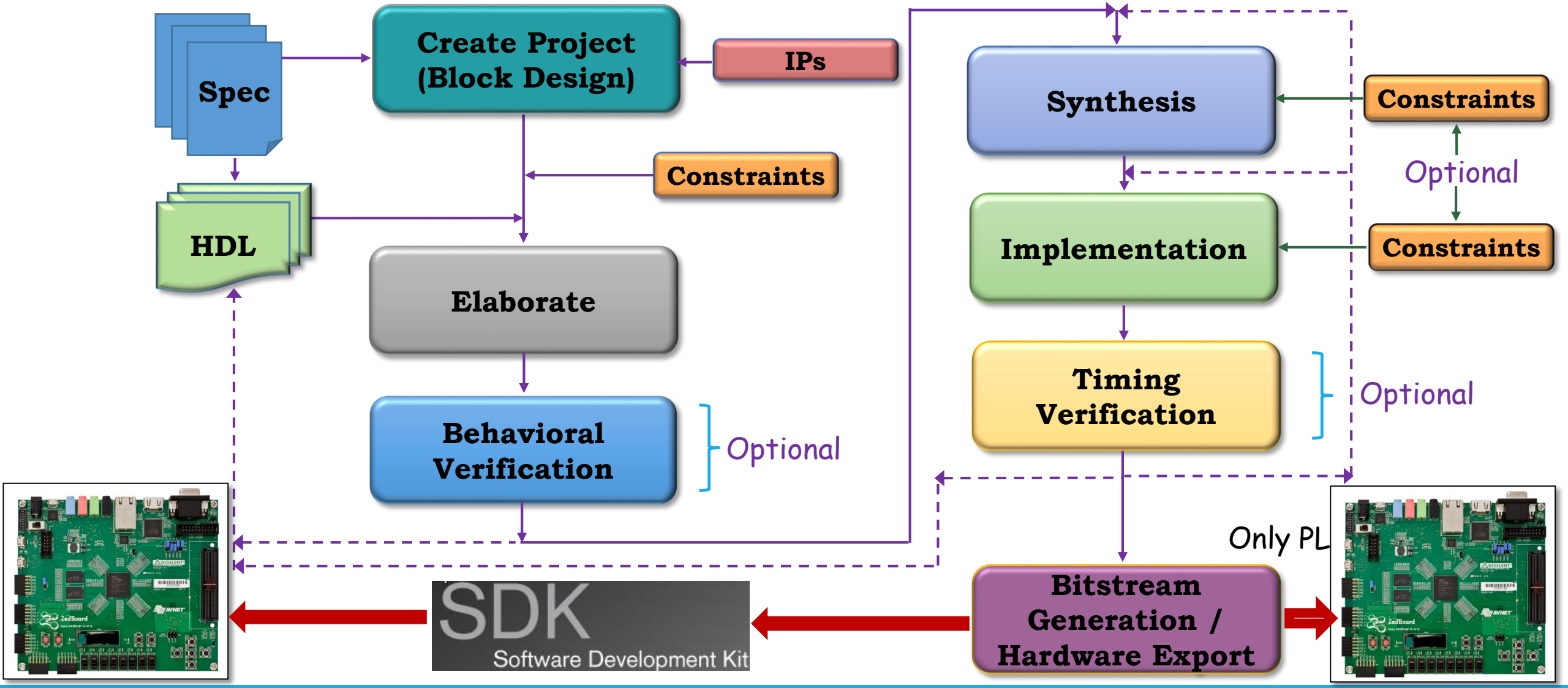
I/O pin planning

- Interactive rule-based I/O assignment



Hierarchical Design Analysis and Implementation Environment

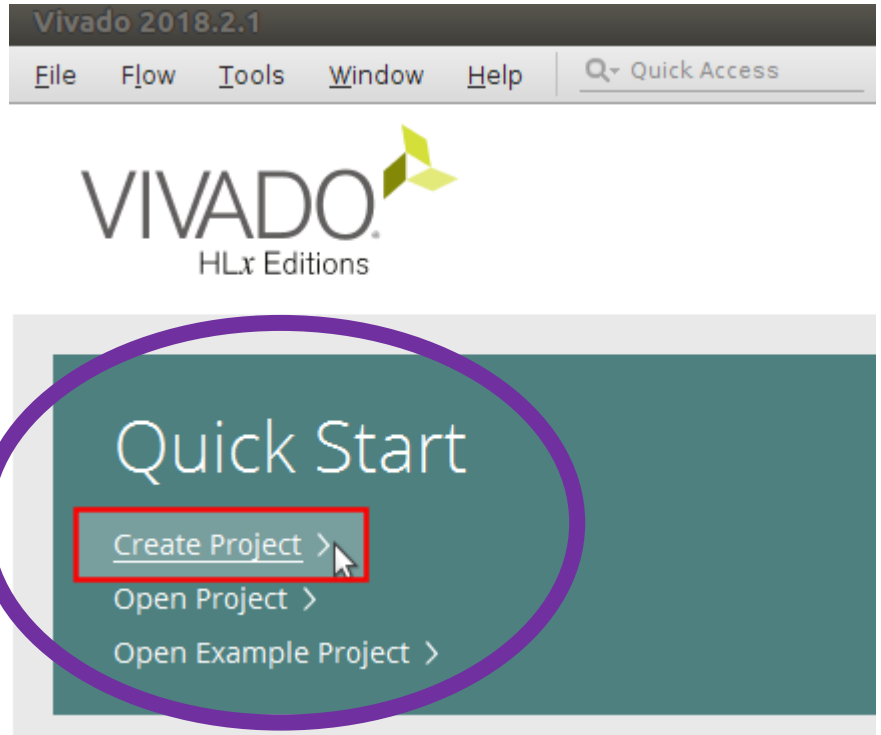
Embedded System Design – Vivado Flow



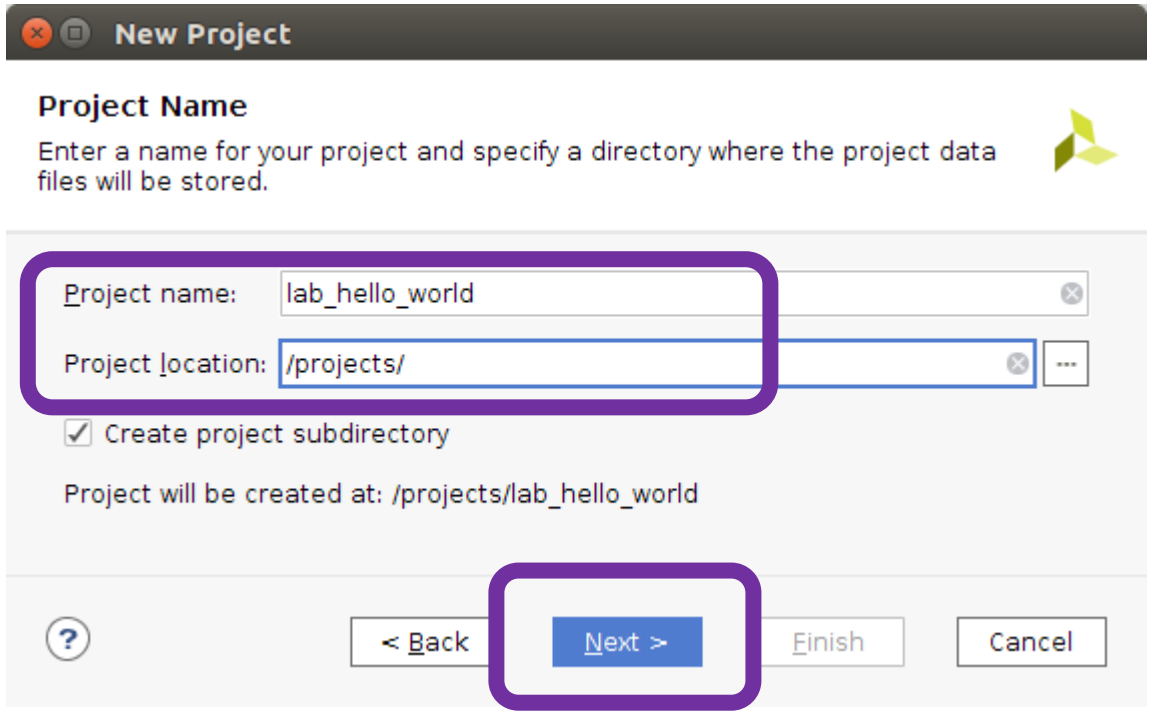
Typical vs Vivado Design Flow

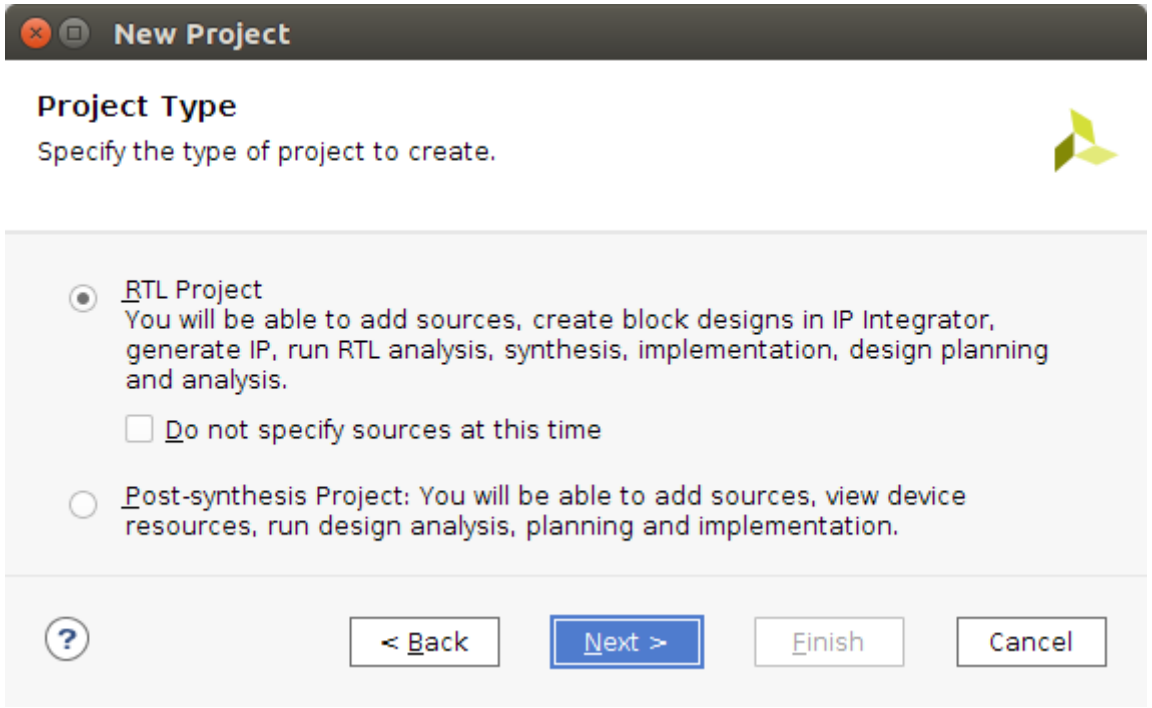
Vivado Flow Practical Steps

Creating a Project in Vivado

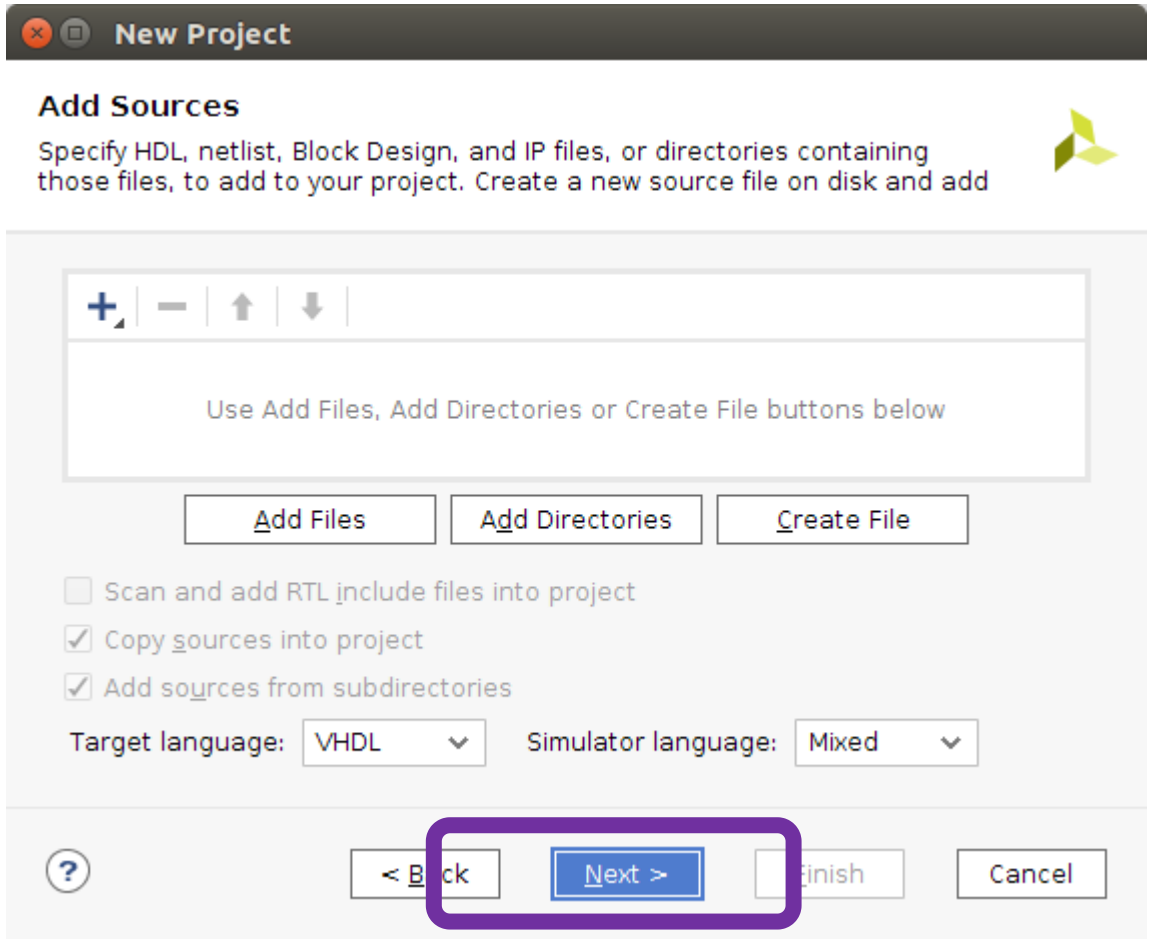


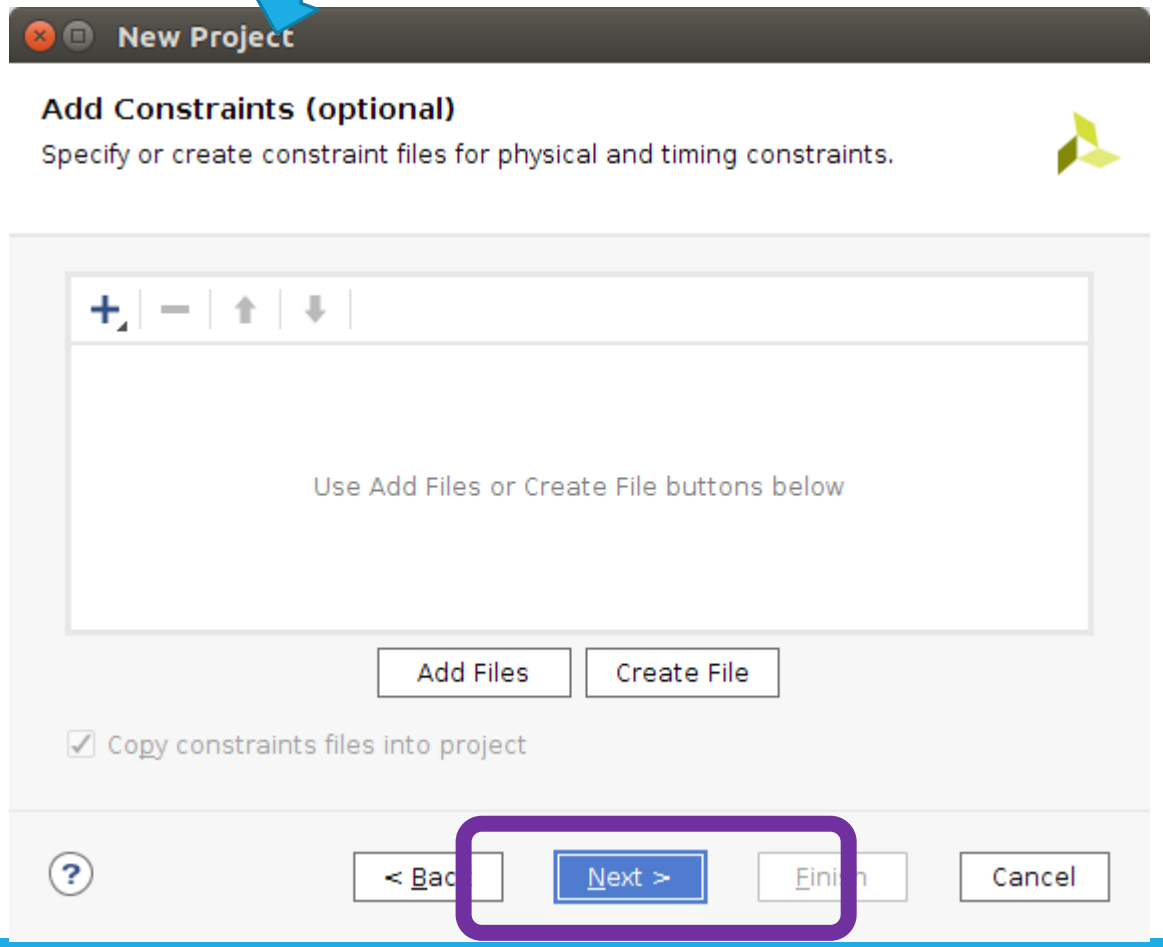
Creating a Project in Vivado



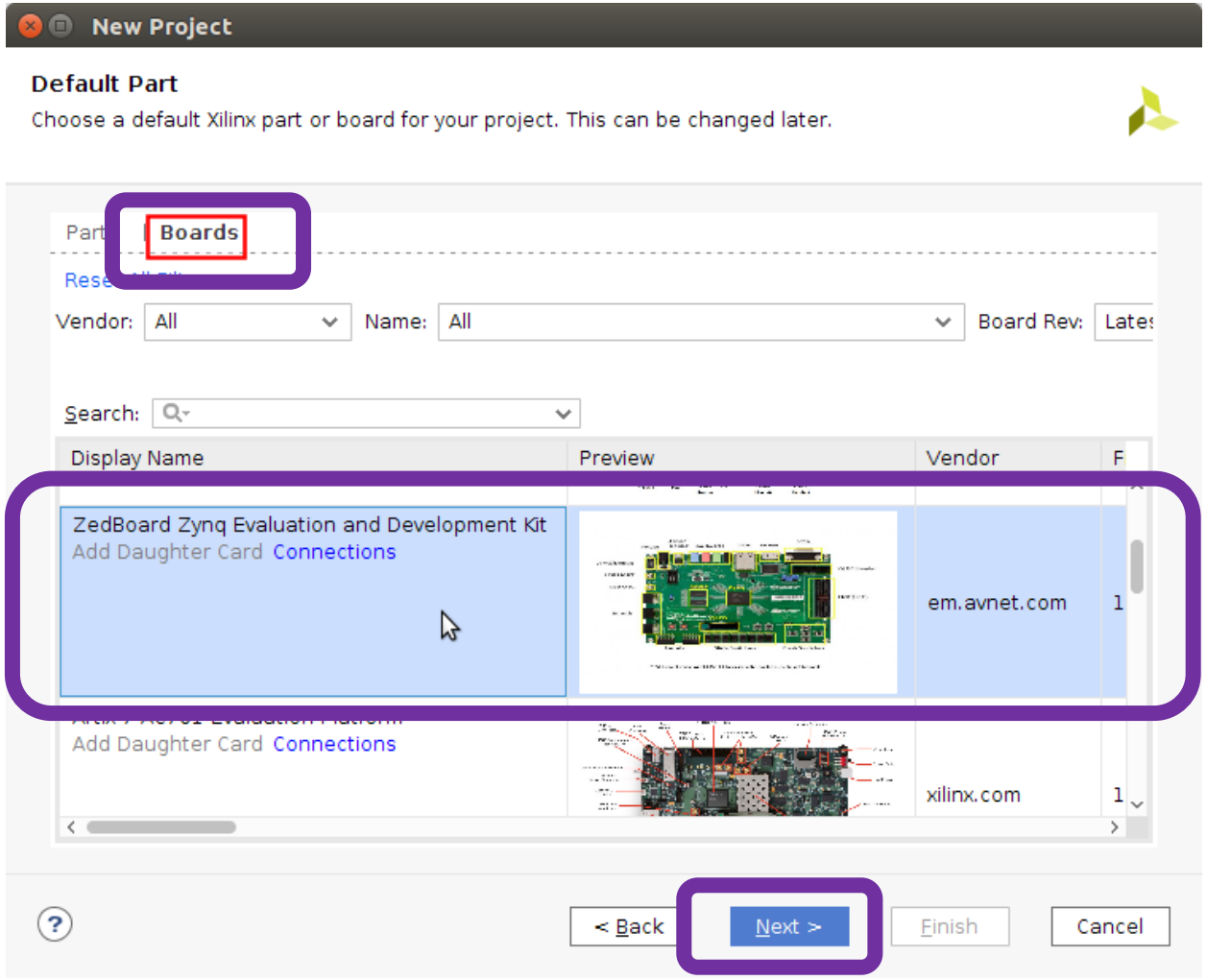


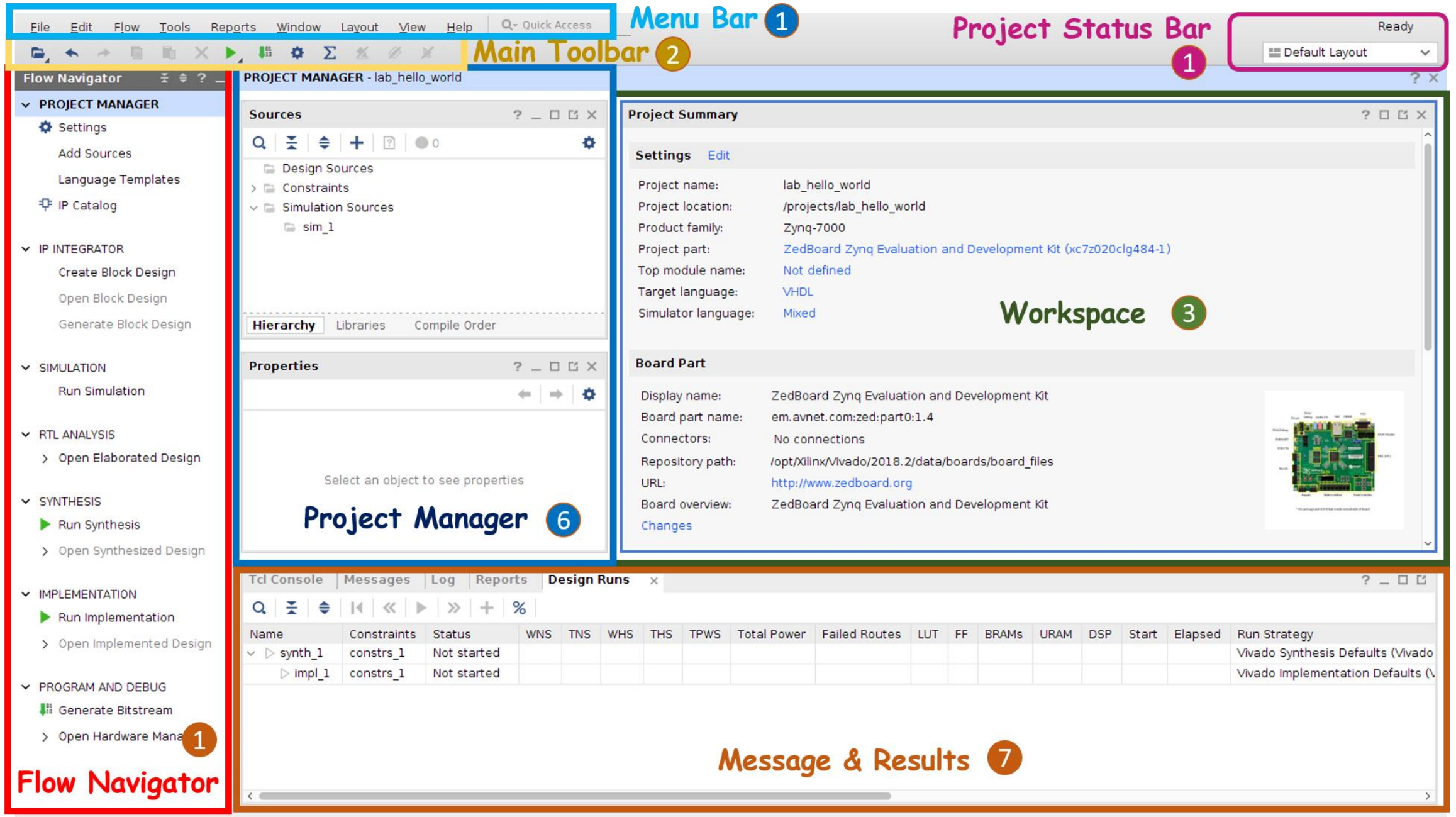
Creating a Project in Vivado





Creating a Project in Vivado

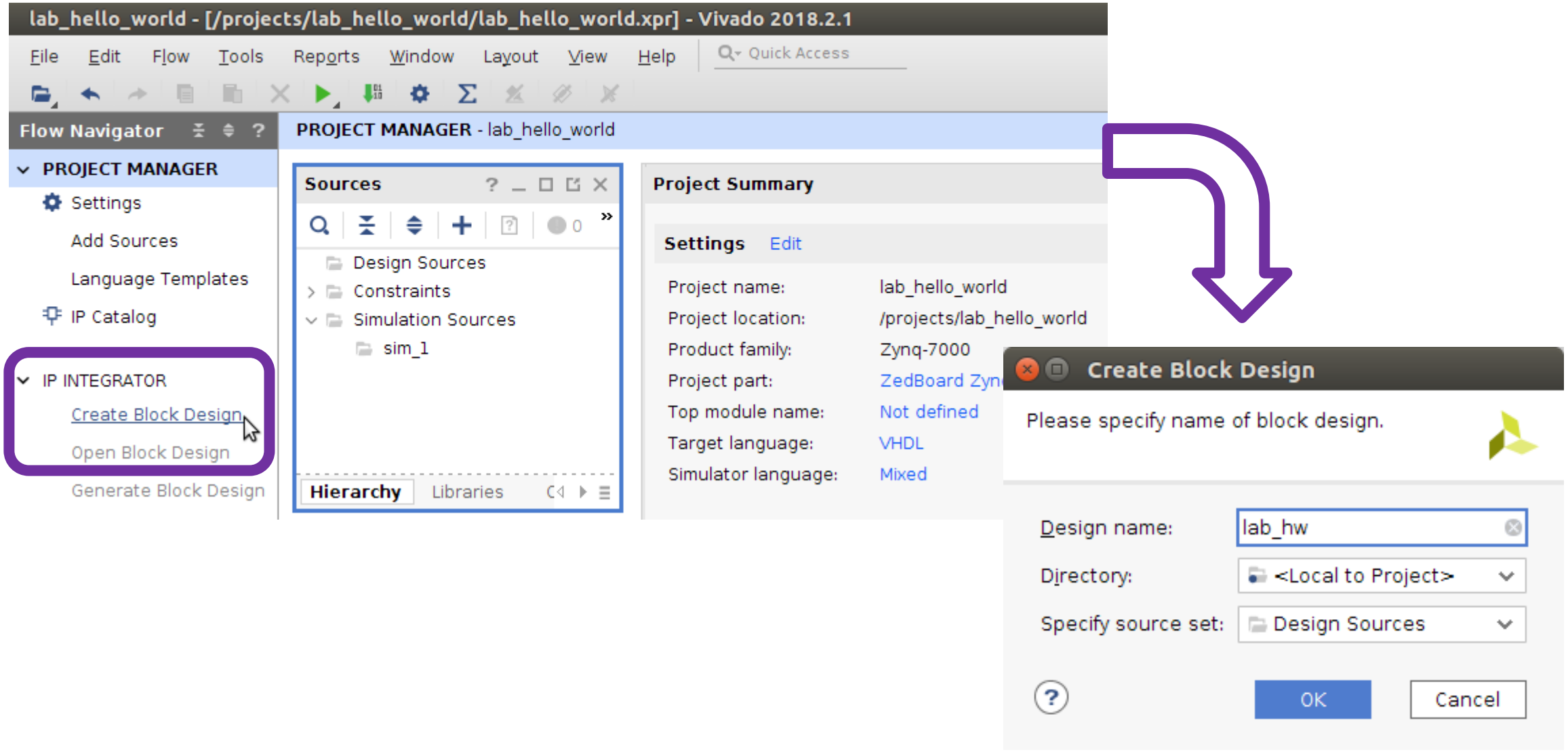



File Edit Flow Tools Reports Window Layout View Help
Quick Access
Menu Bar 1
Main Toolbar 2
Project Status Bar 1
Ready
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Mana
1
Flow Navigator
PROJECT MANAGER - lab_hello_world
Sources
Design Sources
Constraints
Simulation Sources
sim_1
Hierarchy
Libraries
Compile Order
Properties
Select an object to see properties
Project Manager 6
Project Summary
Settings Edit
Project name: lab_hello_world
Project location: /projects/lab_hello_world
Product family: Zynq-7000
Project part: ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1)
Top module name: Not defined
Target language: VHDL
Simulator language: Mixed
Workspace 3
Board Part
Display name: ZedBoard Zynq Evaluation and Development Kit
Board part name: em.avnet.com:zed:part0:1.4
Connectors: No connections
Repository path: /opt/Xilinx/Vivado/2018.2/data/boards/board_files
URL: http://www.zedboard.org
Board overview: ZedBoard Zynq Evaluation and Development Kit
Changes
Tcl Console
Messages
Log
Reports
Design Runs
Name Constraints Status WNS TNS WHS THS TPWS Total Power Failed Routes LUT FF BRAMs URAM DSP Start Elapsed Run Strategy
synth_1 constrs_1 Not started Vivado Synthesis Defaults (Vivado
impl_1 constrs_1 Not started Vivado Implementation Defaults (\
Message & Results 7

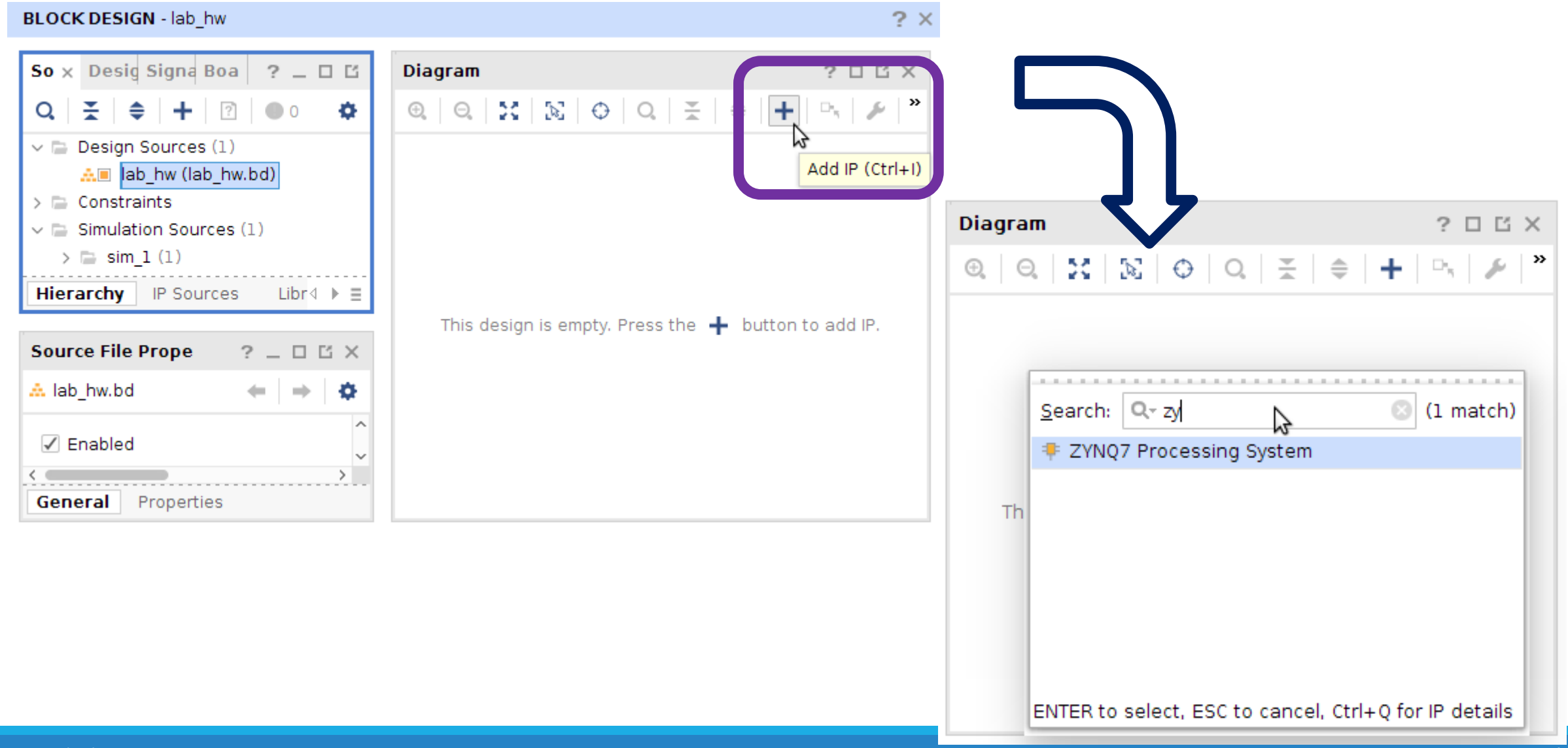
Project Navigator Main Components

1. **Menu Bar**: Vivado IDE commands
2. **Main Toolbar**: Access to the most commonly used Vivado IDE commands
3. **Workspace**: area for schematic panel, device panel, package panel, text editor panel.
4. **Project Status Bar**: displays the status of the currently active design
5. **Flow Navigator**: provide easy access to the tools and commands necessary to guide the design from start to finish.
6. **Project Manager Pane**: by default displays information related to design data and sources, such as Property Window, Netlist Window, and Source Window
7. **Project Status Bar**: displays information about menu bar and toolbar commands; task progresses
8. **Results Window Area**: there are a set of windows, such as Messages, showing message for each process, Tcl Console, Tcl commands of each activity, Reports, reports generated throughout the design flow, Desing Runs, display the different run for the current project

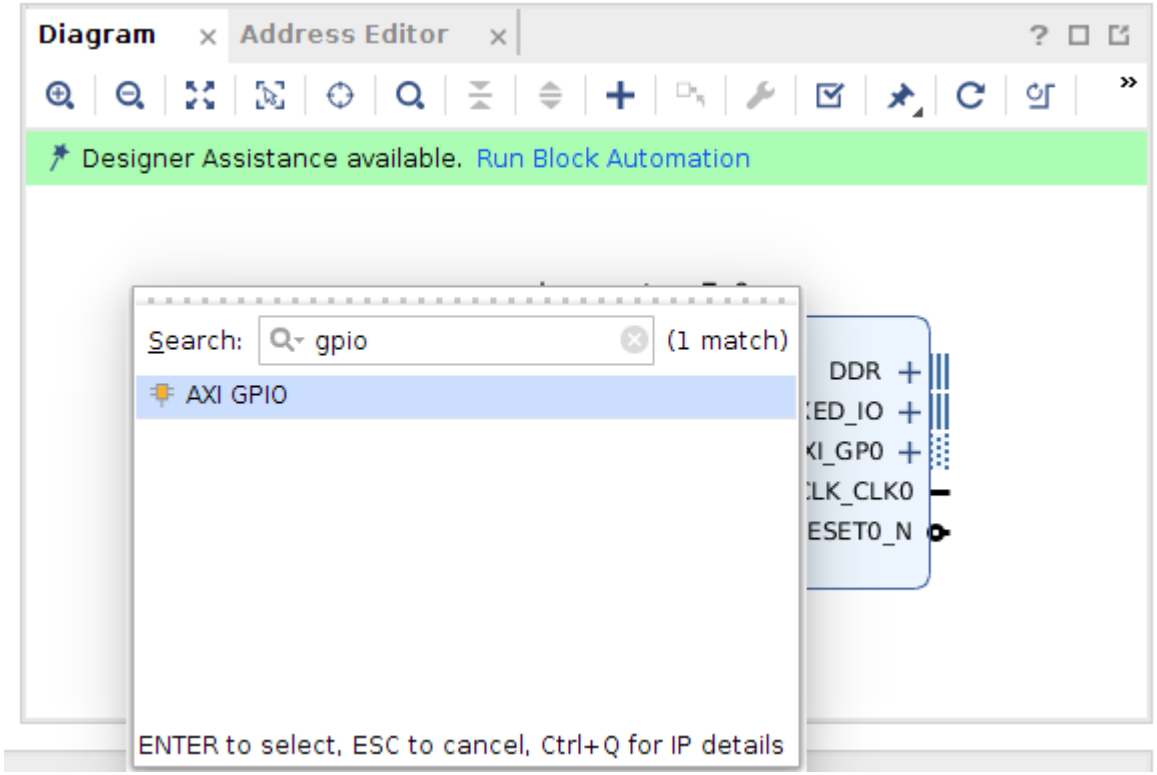
Create a Block design

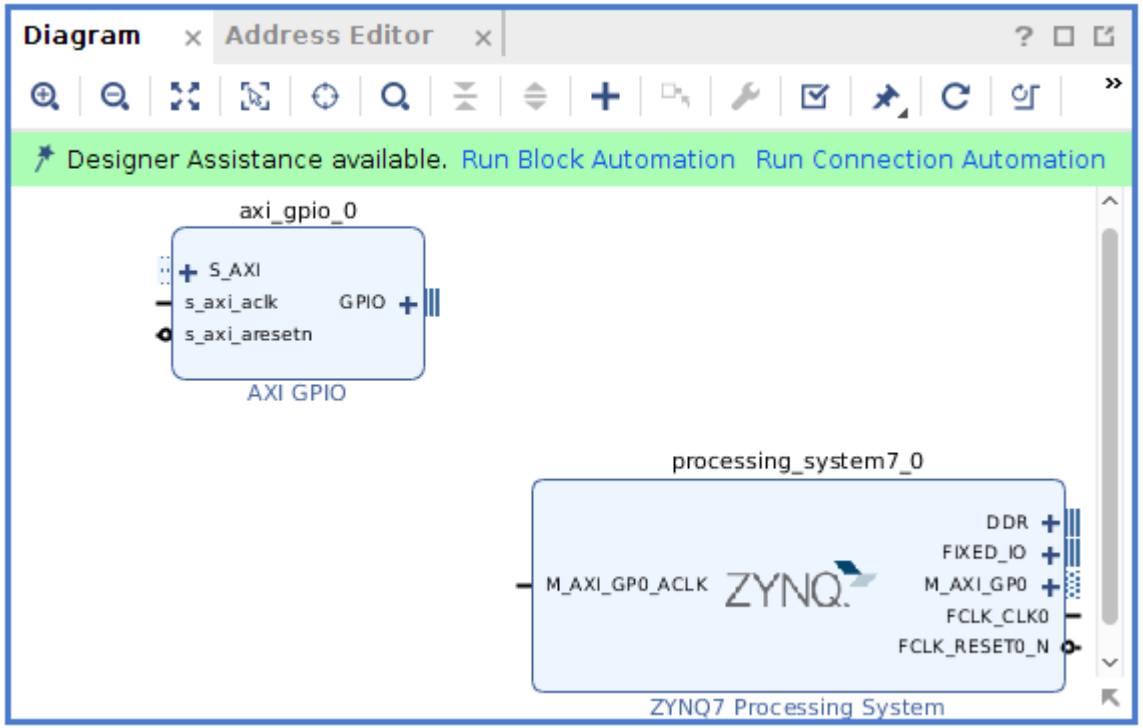


Adding IP Modules to the Design Canvas

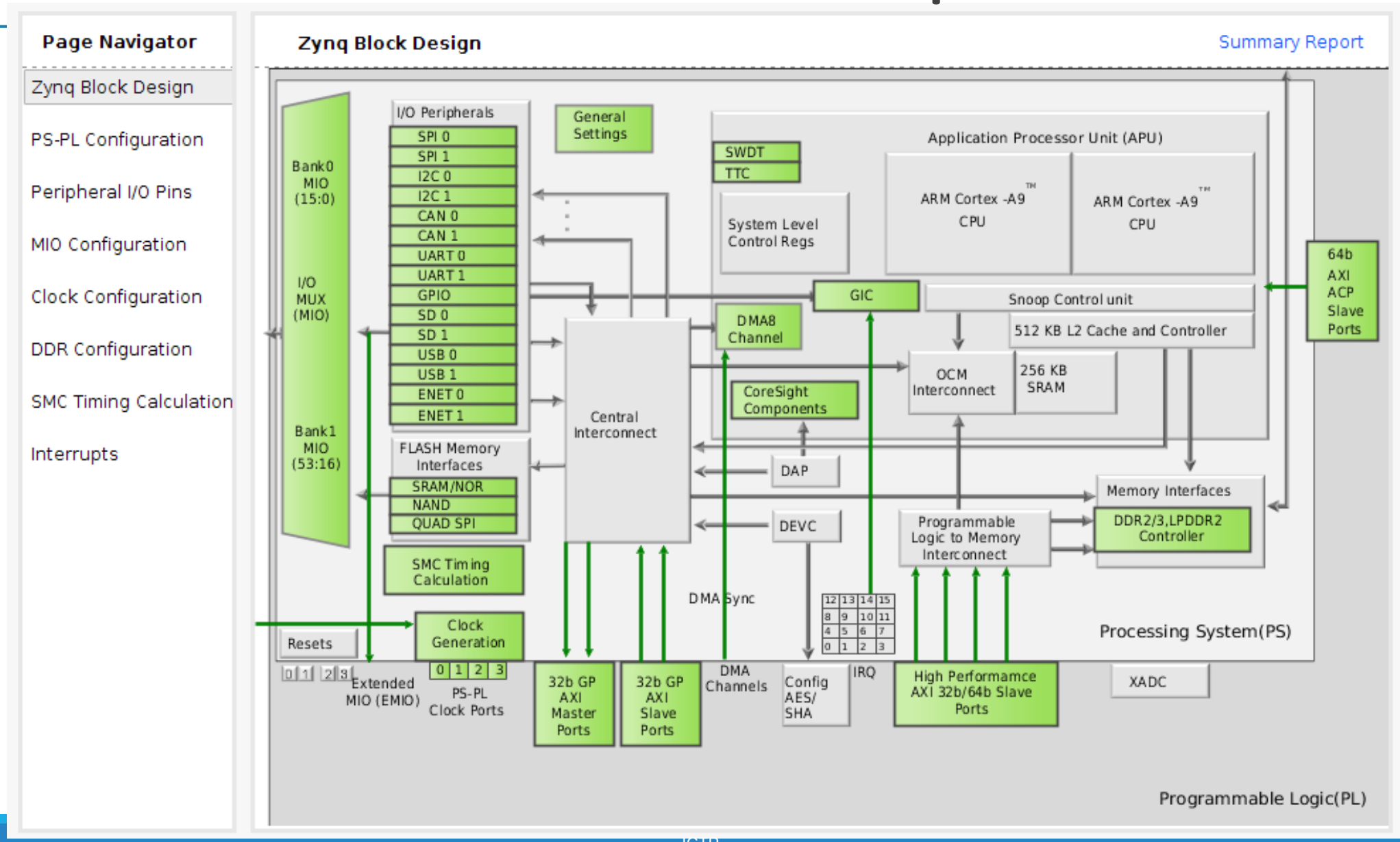


Adding More IPs

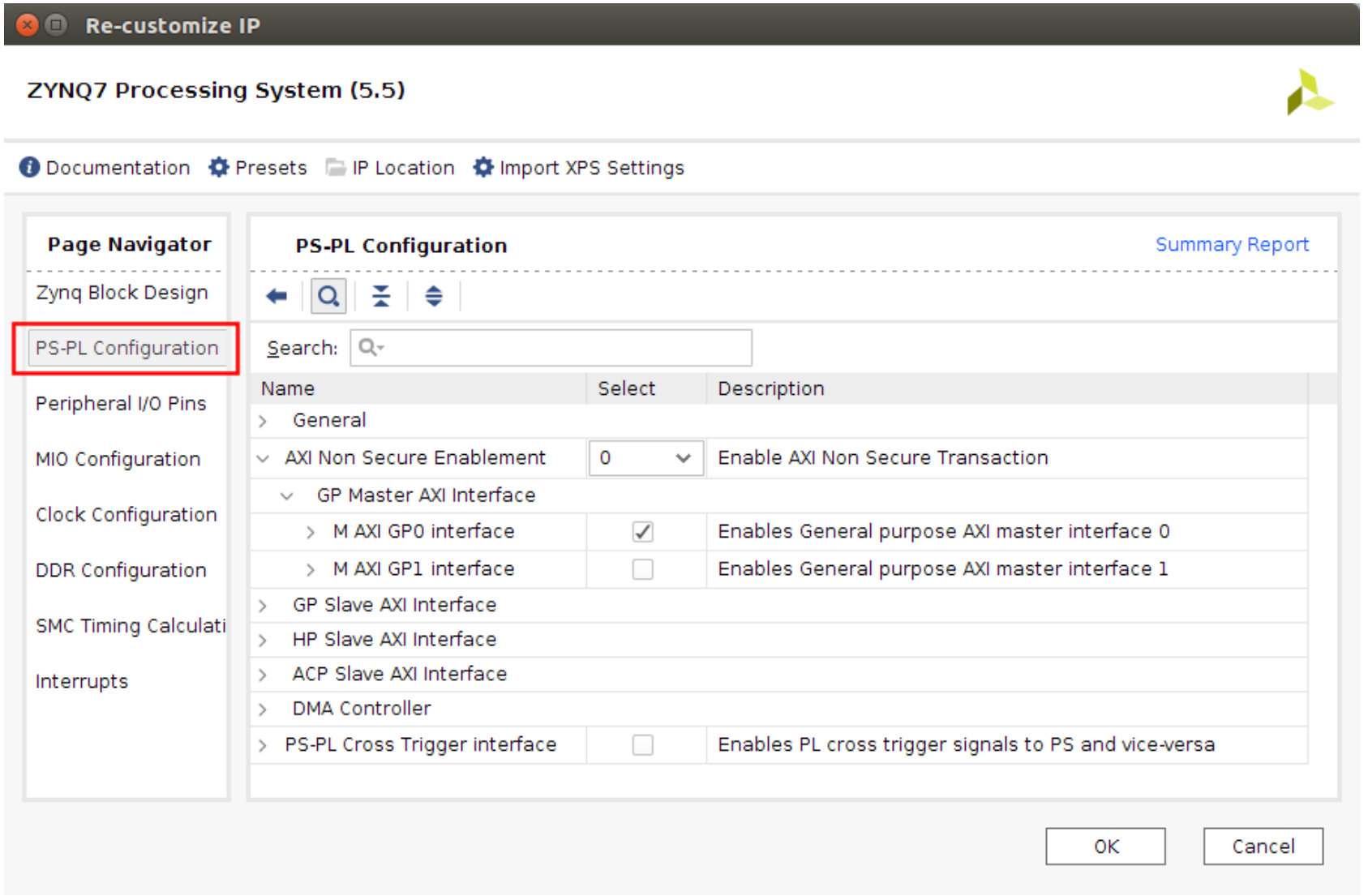




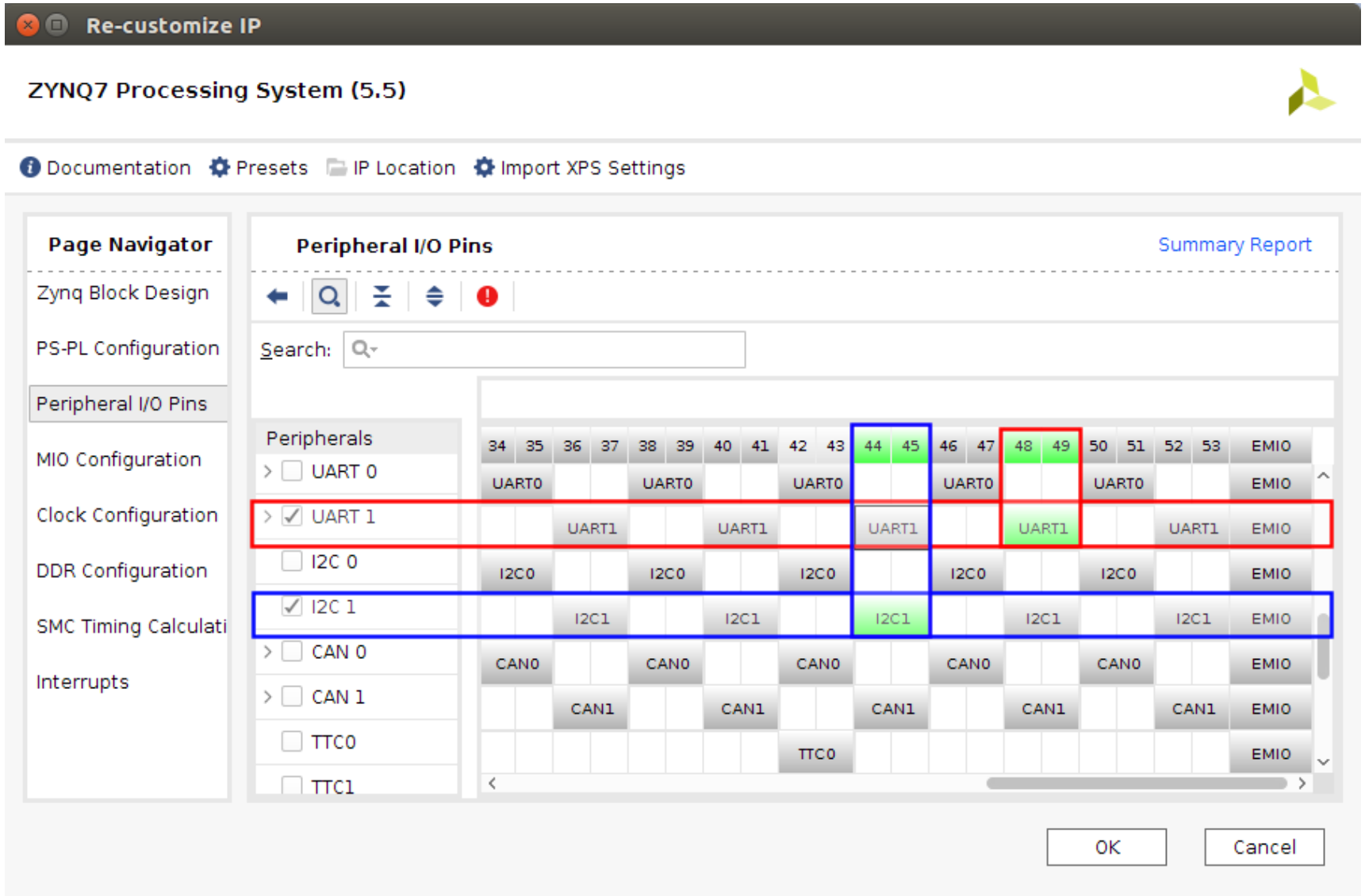
PS Customization Options



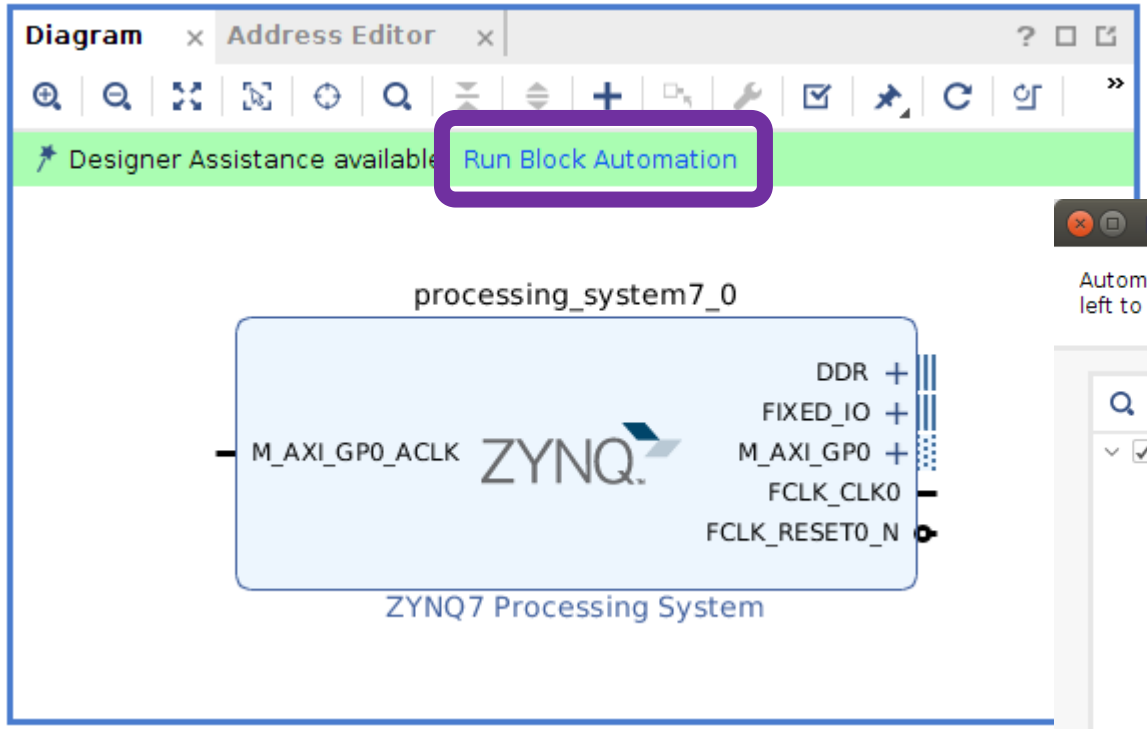
PS-PL Configuration Options

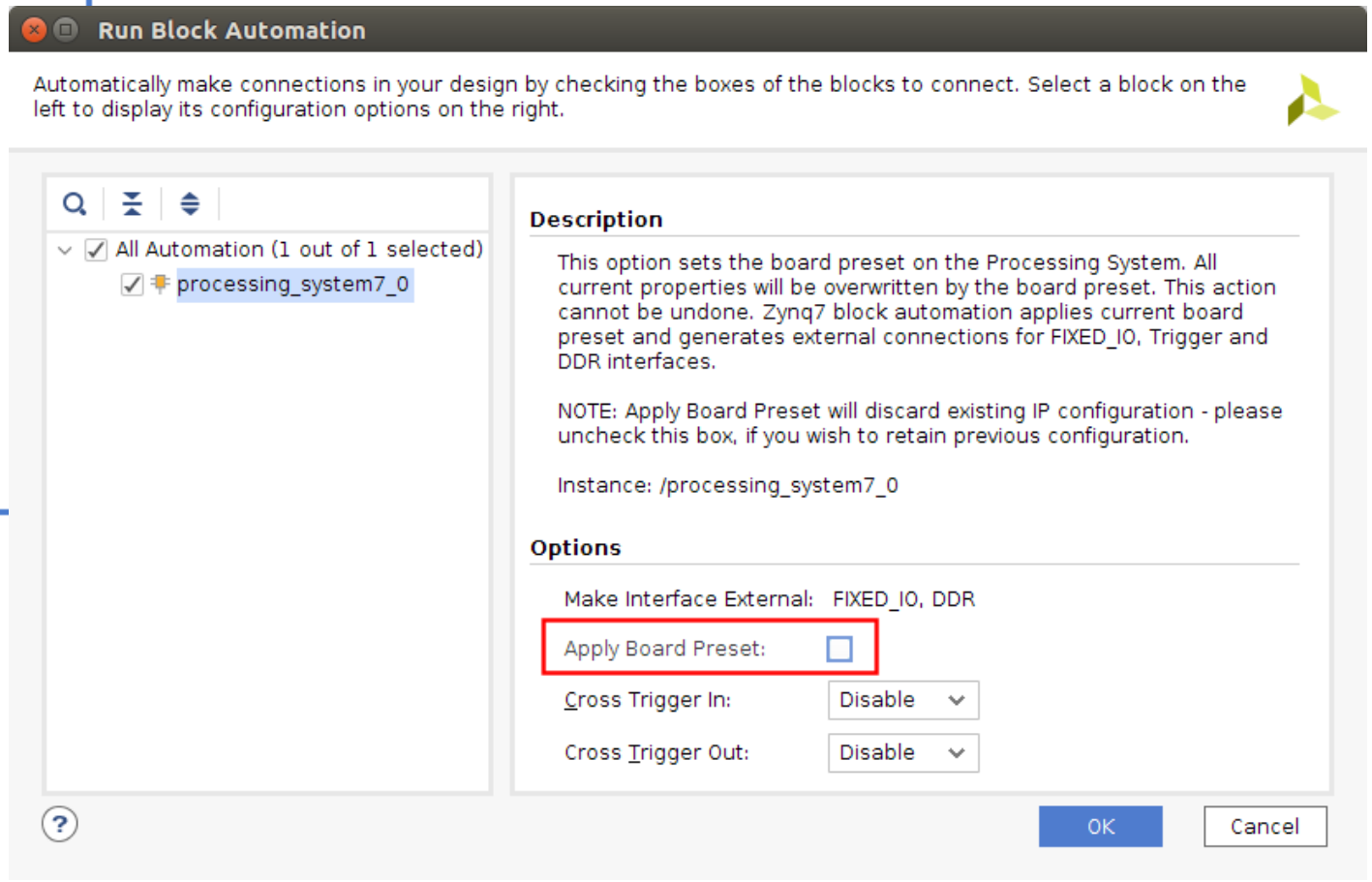


MIO and EMIO Configuration

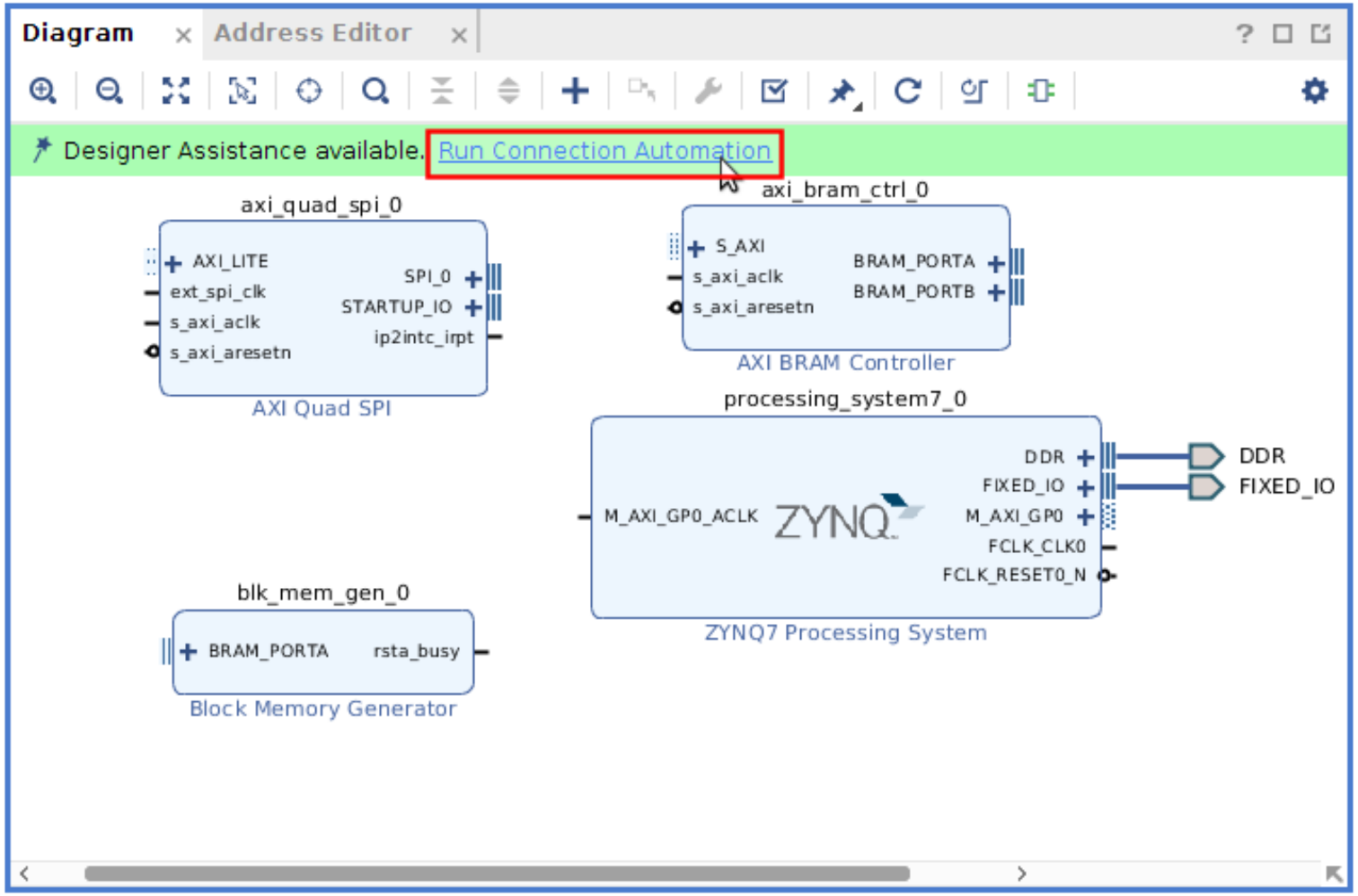


Running Block Automation

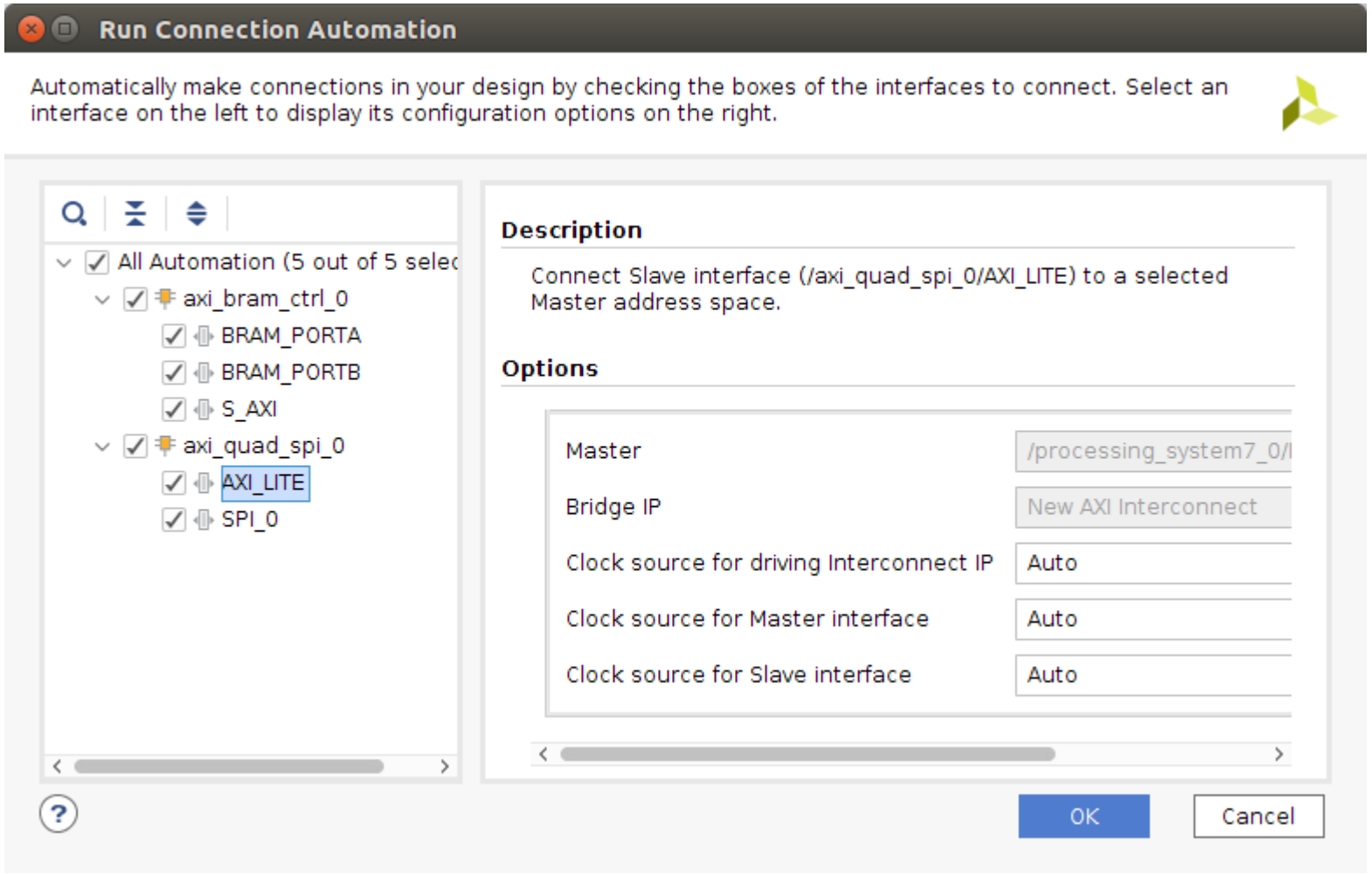




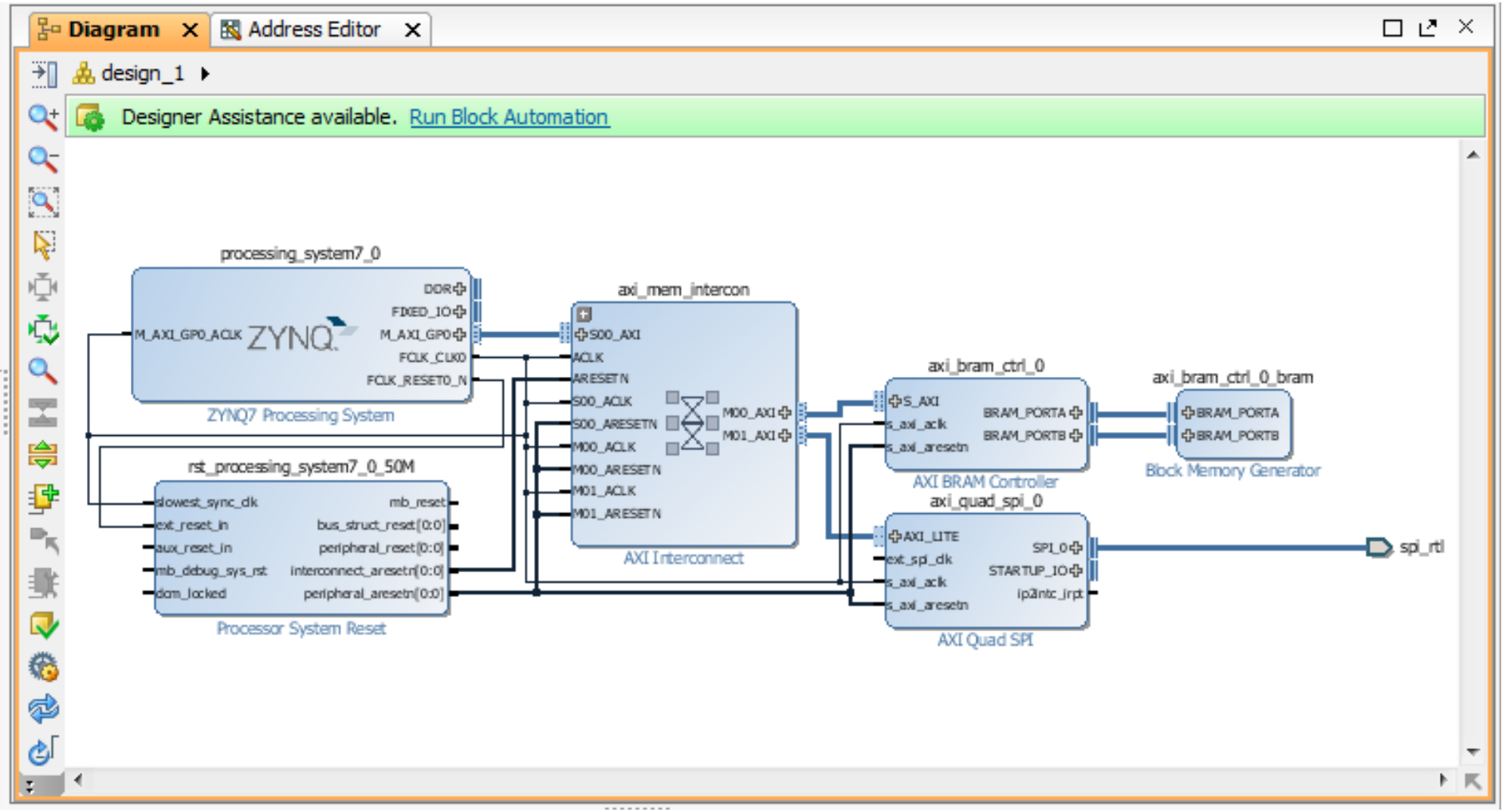
Run Connection Automation



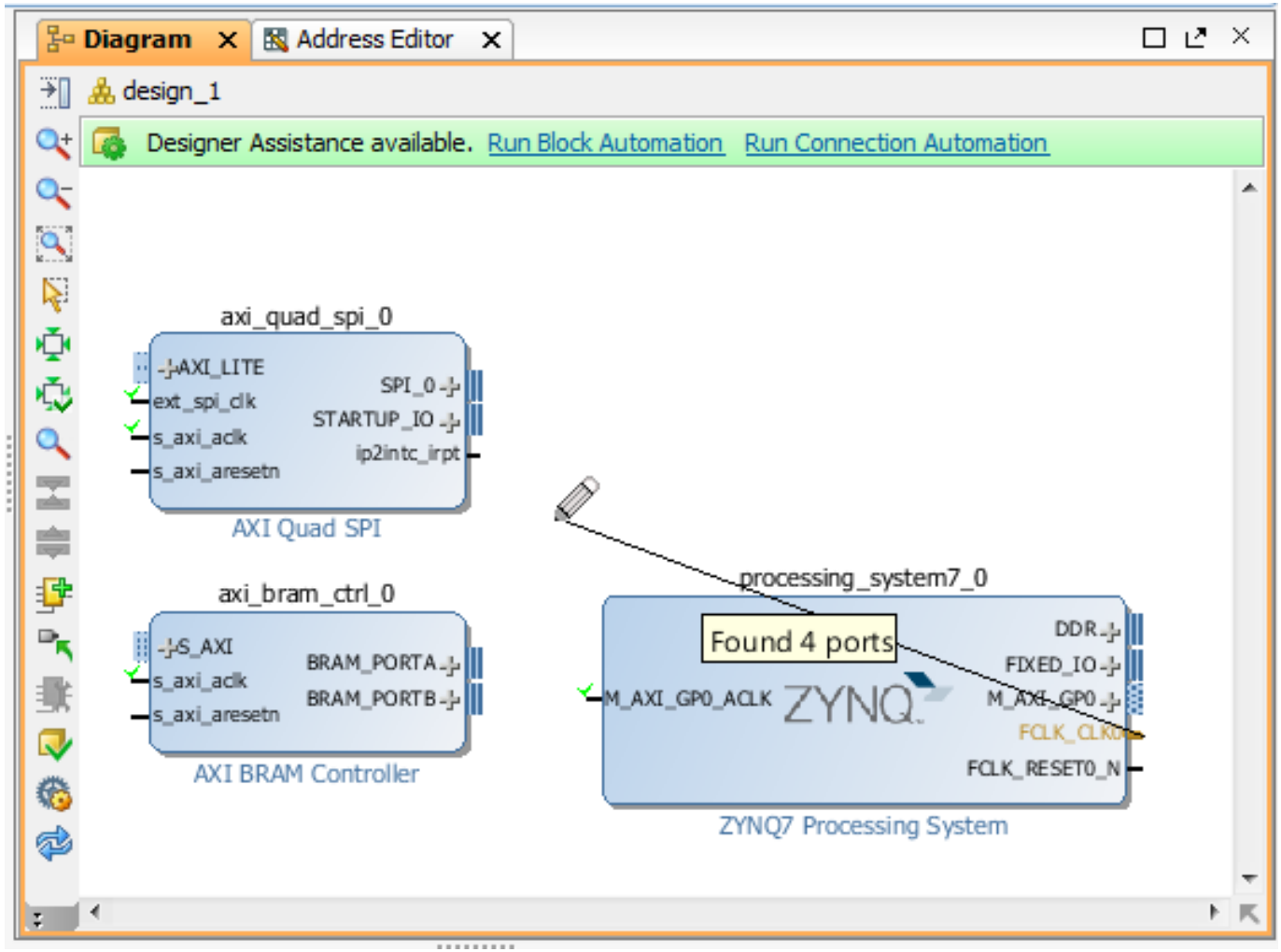
Connecting IPs – Making Connections With the Tool



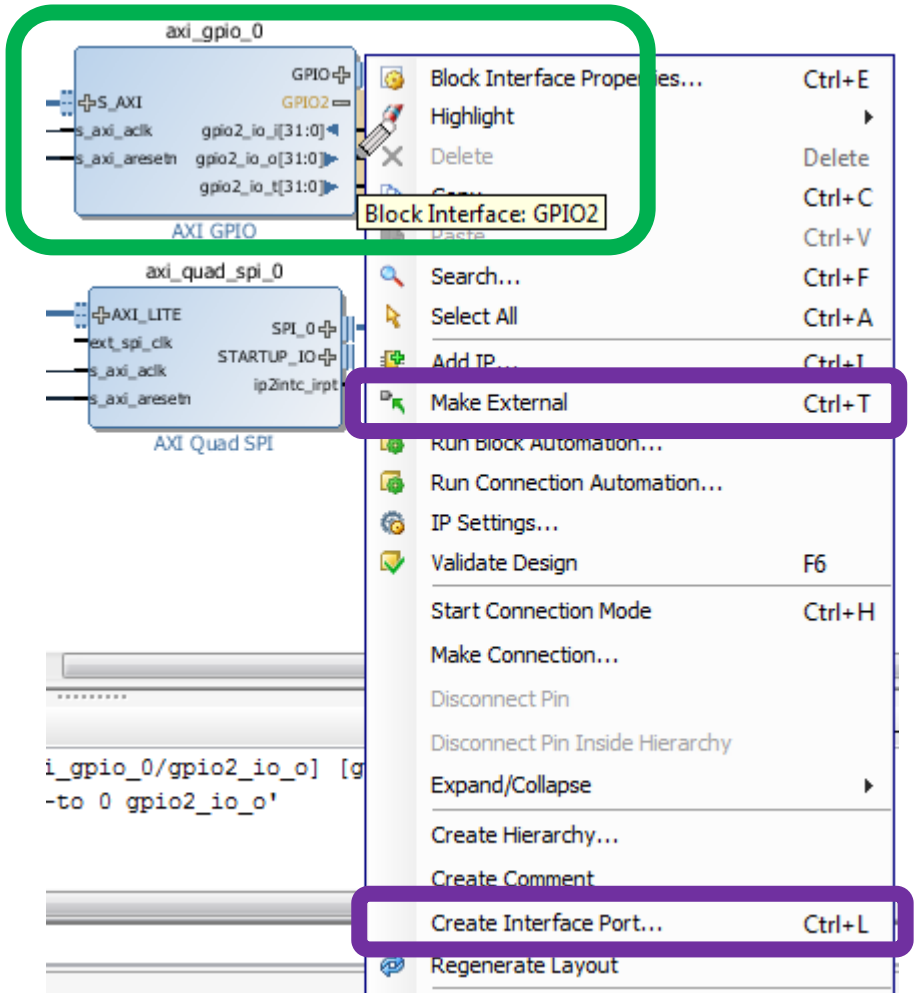
Connecting IPs – Making Connections With the Tool

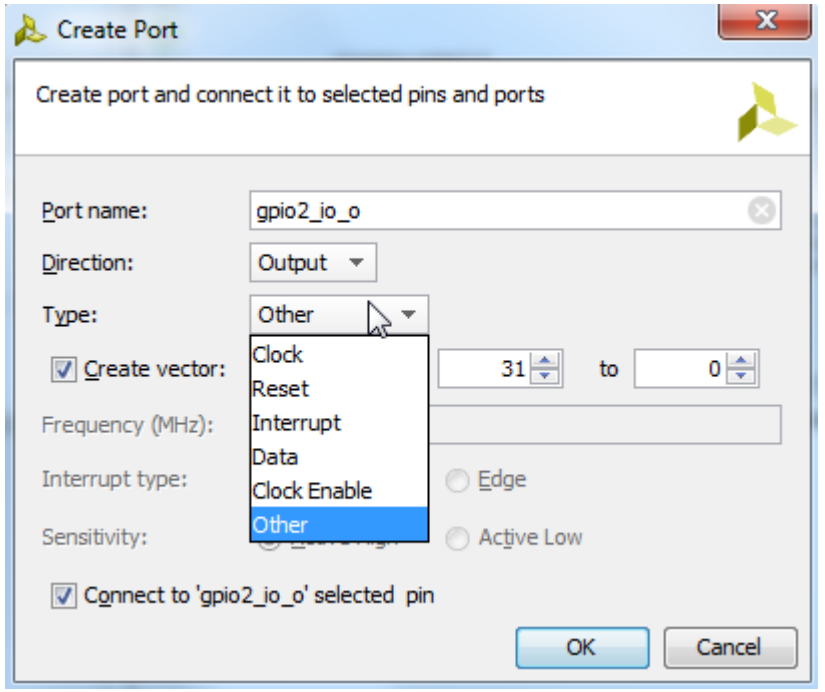


Connecting IPs – Making Connections Manually

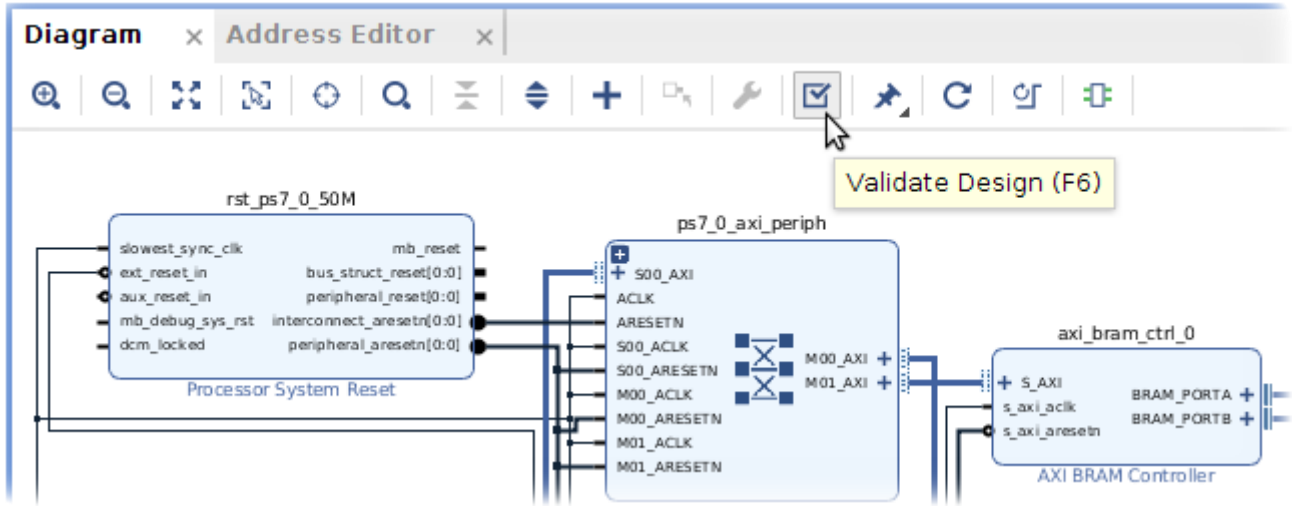


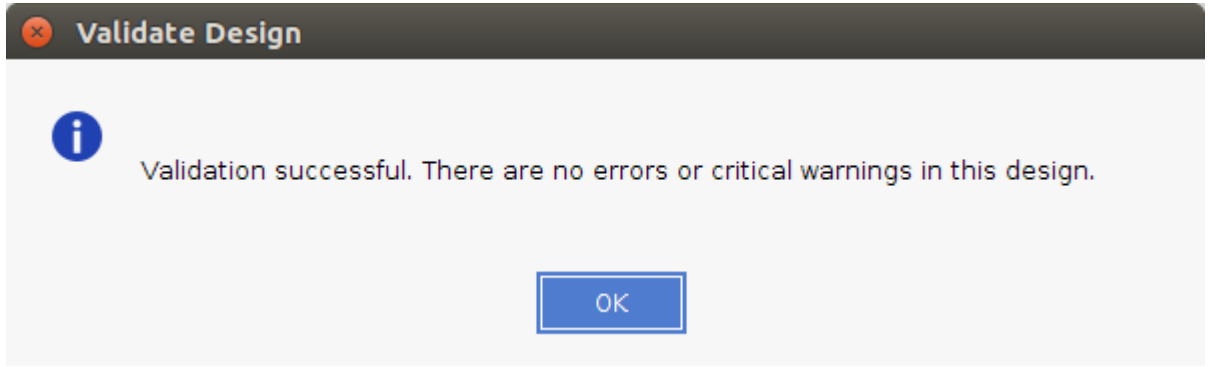
Options for External Connections



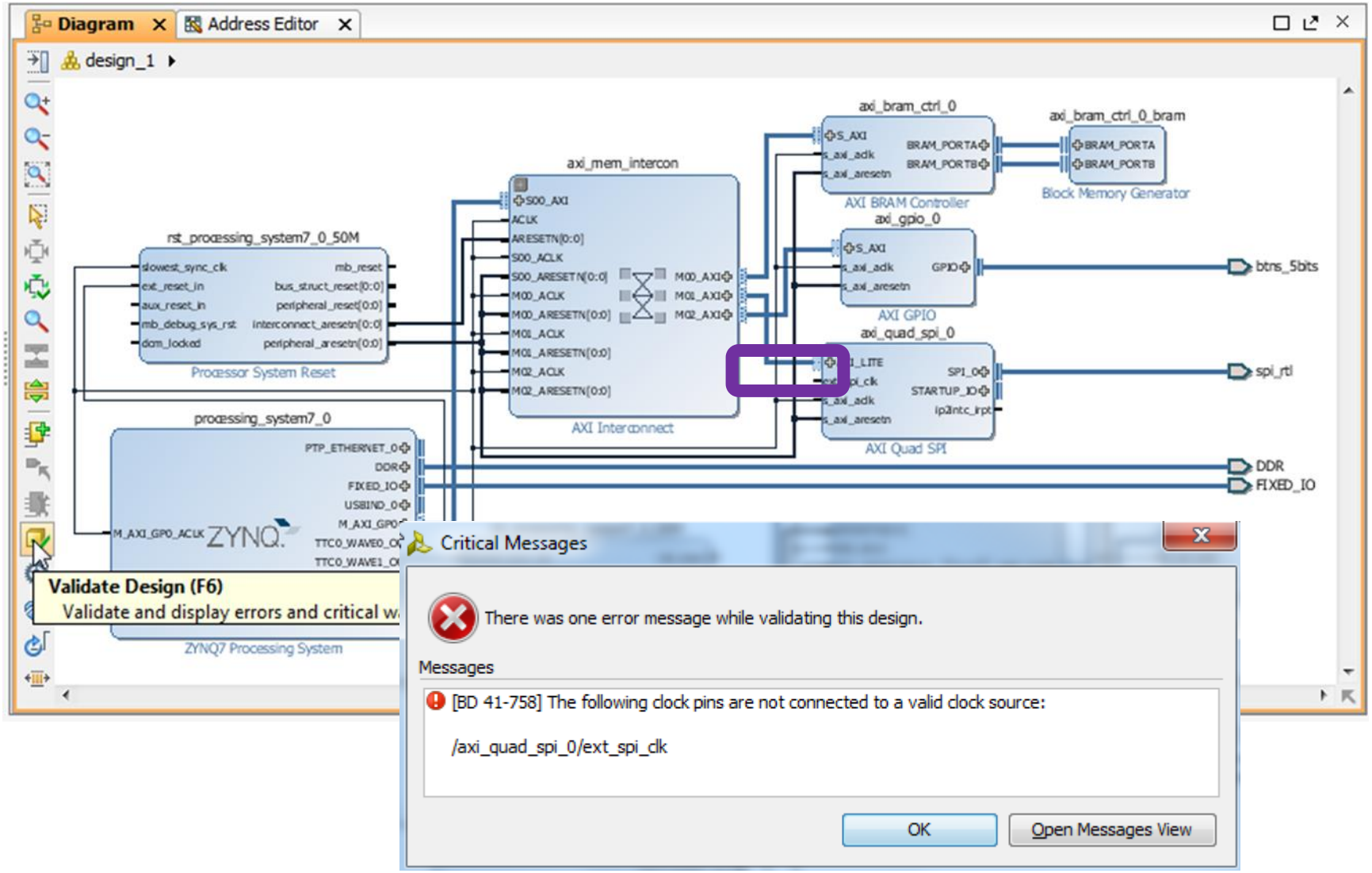


DRC (Desing Rule Check) Design Validation

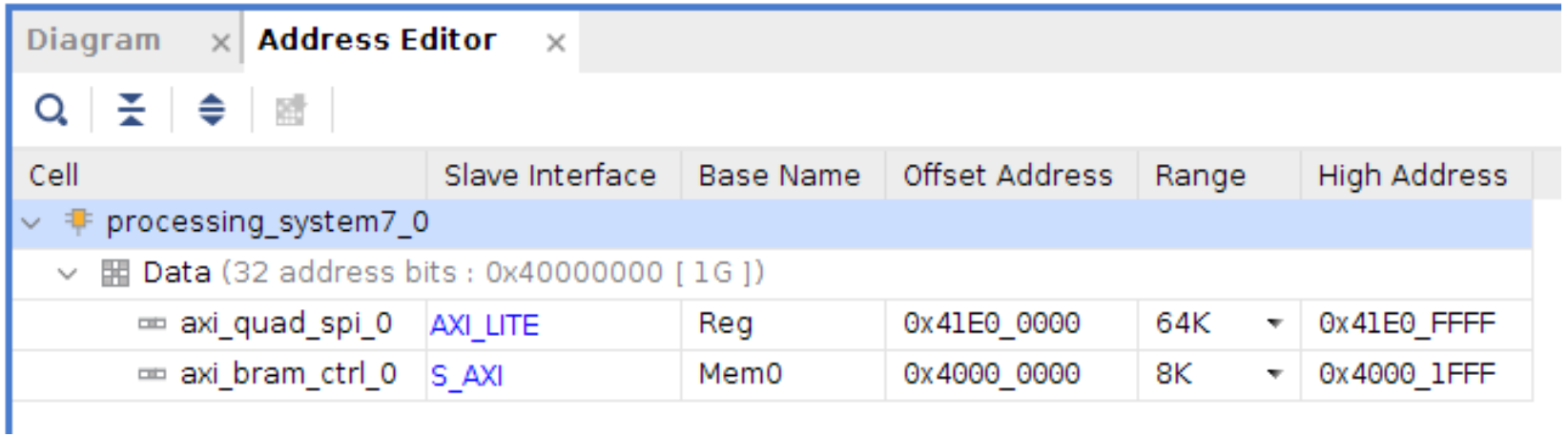




DRC – Design Validation

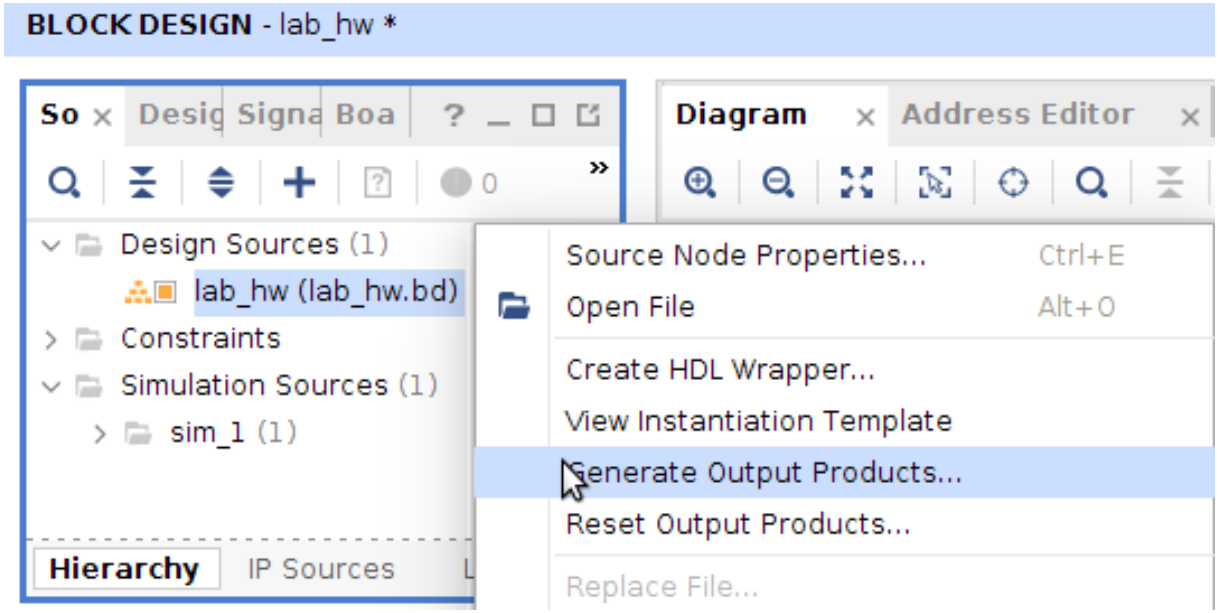


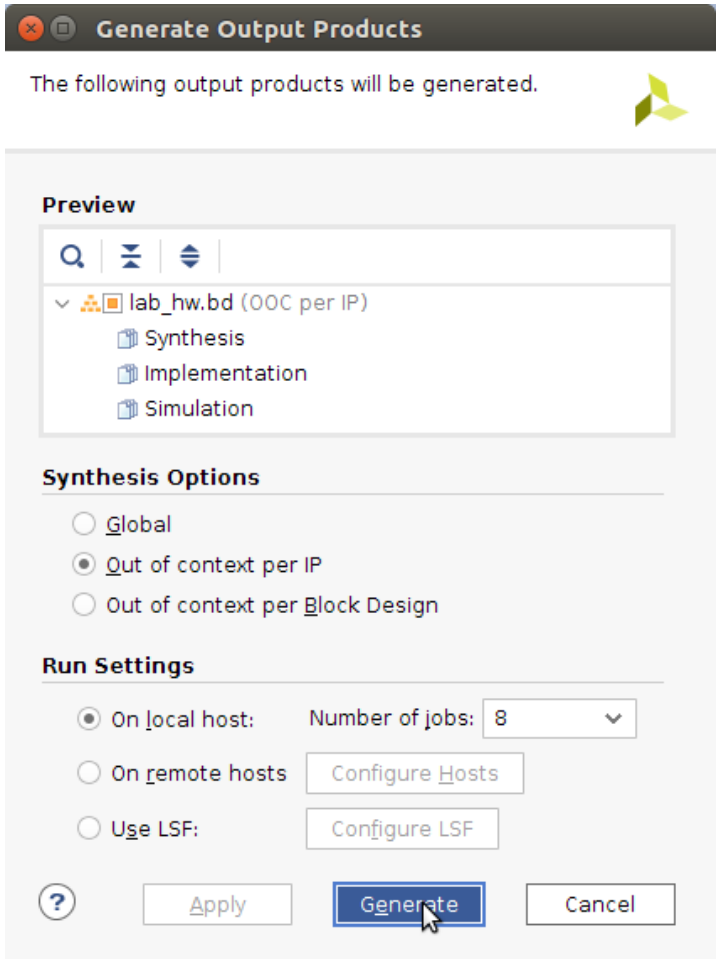
Address Map



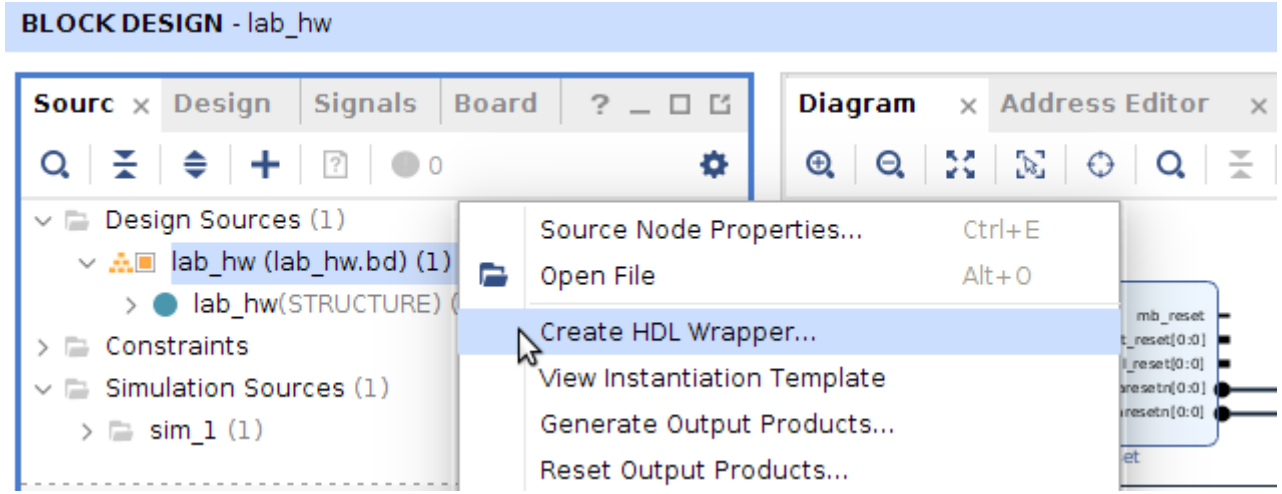
Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
axi_quad_spi_0	AXI_LITE	Reg	0x41E0_0000	64K	0x41E0_FFFF
axi_bram_ctrl_0	S_AXI	Mem0	0x4000_0000	8K	0x4000_1FFF

Generating Output Products





Creating an HDL Wrapper



Project Data and Directories

All project data is stored in a *project_name* directory containing the following directories

- ***project_name.xpr*** file: Object that is selected to open a project (Vivado IDE project file)
- ***project_name.runs*** directory: Contains all run data
- ***project_name.srcs*** directory: Contains all imported local HDL source files, netlists, and XDC files
- ***project_name.data*** directory: Stores floorplan and netlist data

Journal and Log Files

Journal file (*vivado.jou*)

- Contains just the Tcl commands executed by the Vivado IDE

Log file (*vivado.log*)

- Contains all messages produced by the Vivado IDE, including Tcl commands and results, info, warning, error messages, etc.

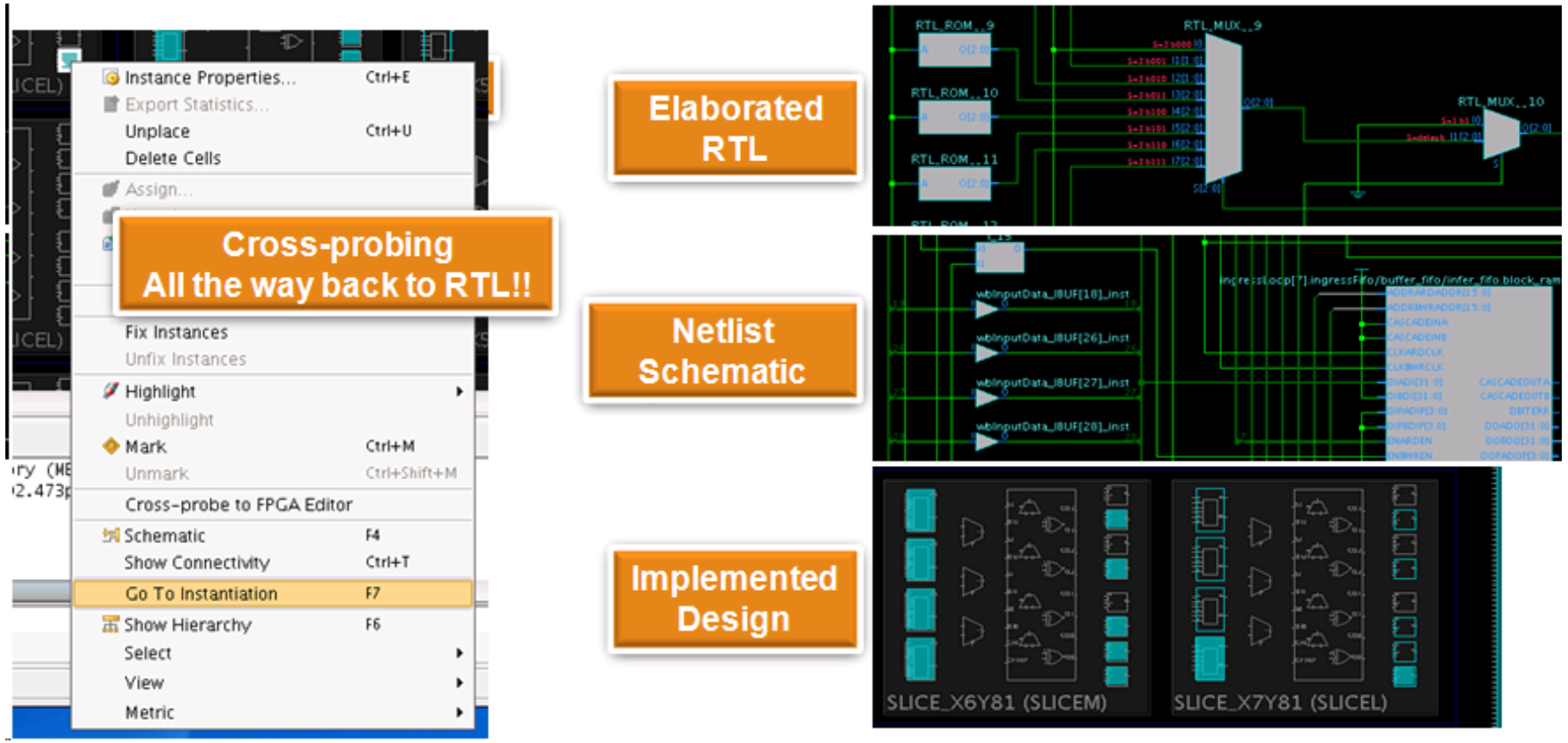
Location

- Linux: directory where the Vivado IDE is invoked
- Windows via icon: *%APPDATA%\Xilinx\Vivado* or *C:\Users\<user_name>\AppData\Roaming\Xilinx\Vivado*
- Windows via command line: directory where the Vivado IDE is invoked
- From the GUI
  - Select File > Open Log File
  - Select File > Open Journal File

Vivado Visualization Features

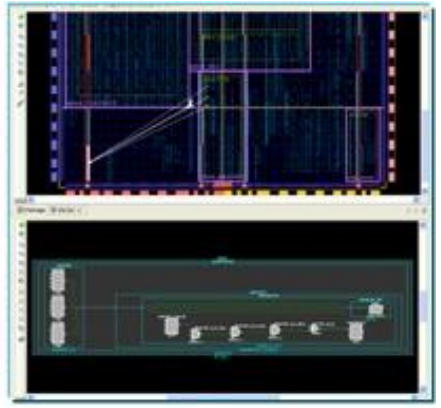
Visualize and debug a design at any flow stage

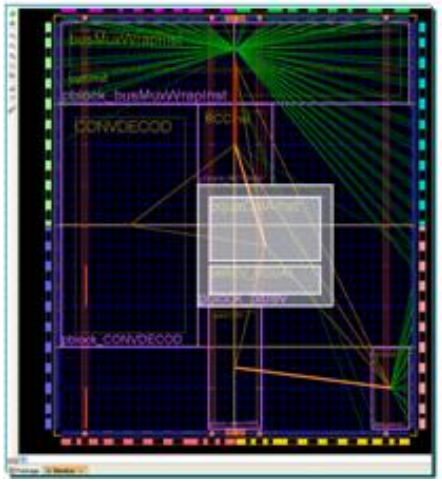
- Cross-probing between netlist/schematic/RTL



Gain Faster Timing Closure

- **Analyze multiple implementation results**
  - Highlight failing timing paths from post-route timing
  - Quickly identify and constrain critical logic path
- **Connectivity display**
  - I/Os, net bundles, clock domains
- **Hierarchical floorplanning**
  - Guide *place & route* toward better results
- **Utilization estimates**
  - All resource types shown for each Pblock
  - Clocks or carry chains



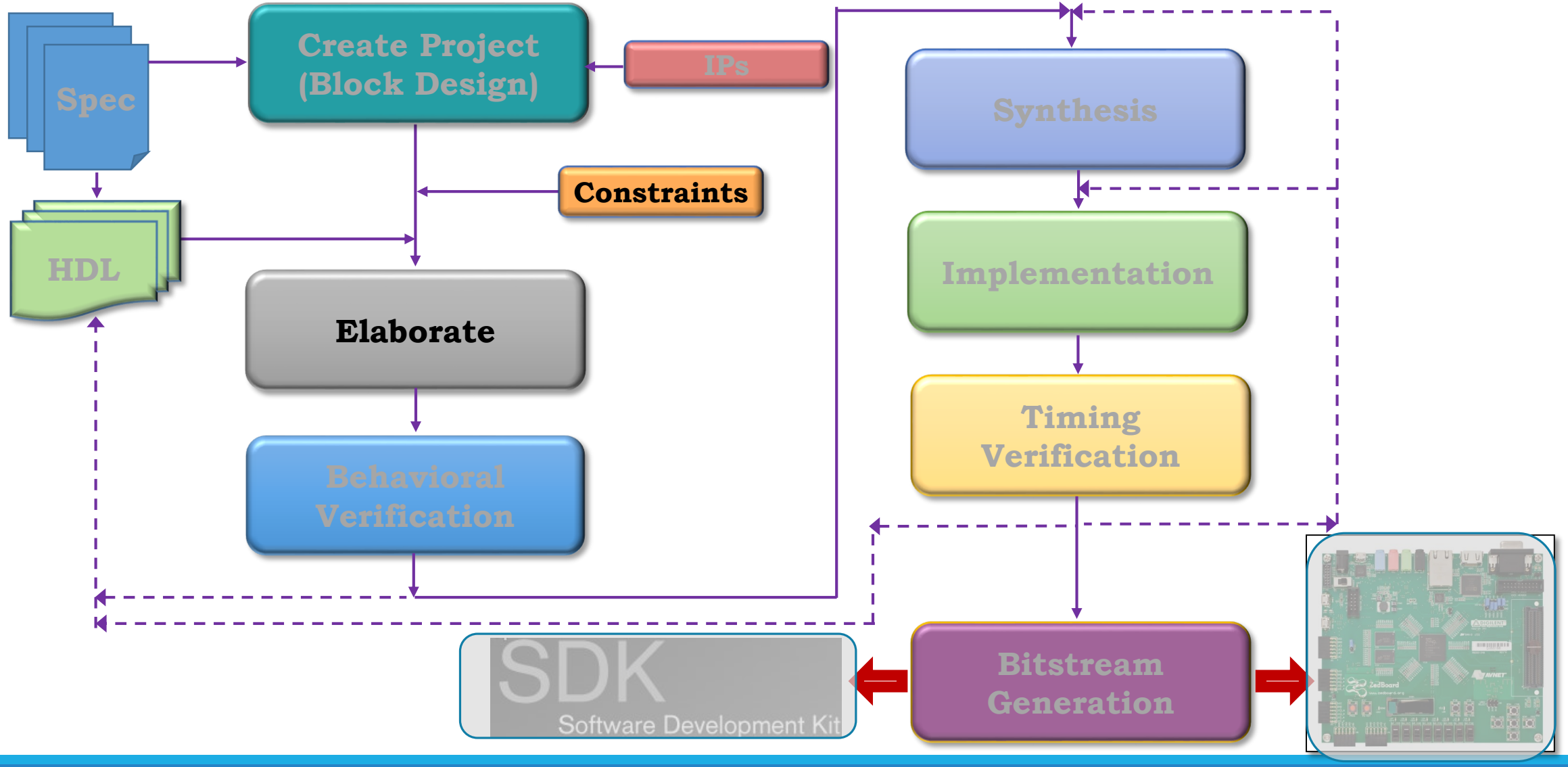


Tool Command Line (.tcl) Features

- Tcl Console enables the designer to actively query the design netlist
- Full Tcl scripting support in two design flows
  - Project-based design flow provides easy project management by the Vivado IDE
  - Non-project batch design flow enables entire flow to be executed in memory
- *Journal* and *log* files can be used for script construction

Vivado Design Suite
Elaboration Process

Embedded System Design – Vivado Flow

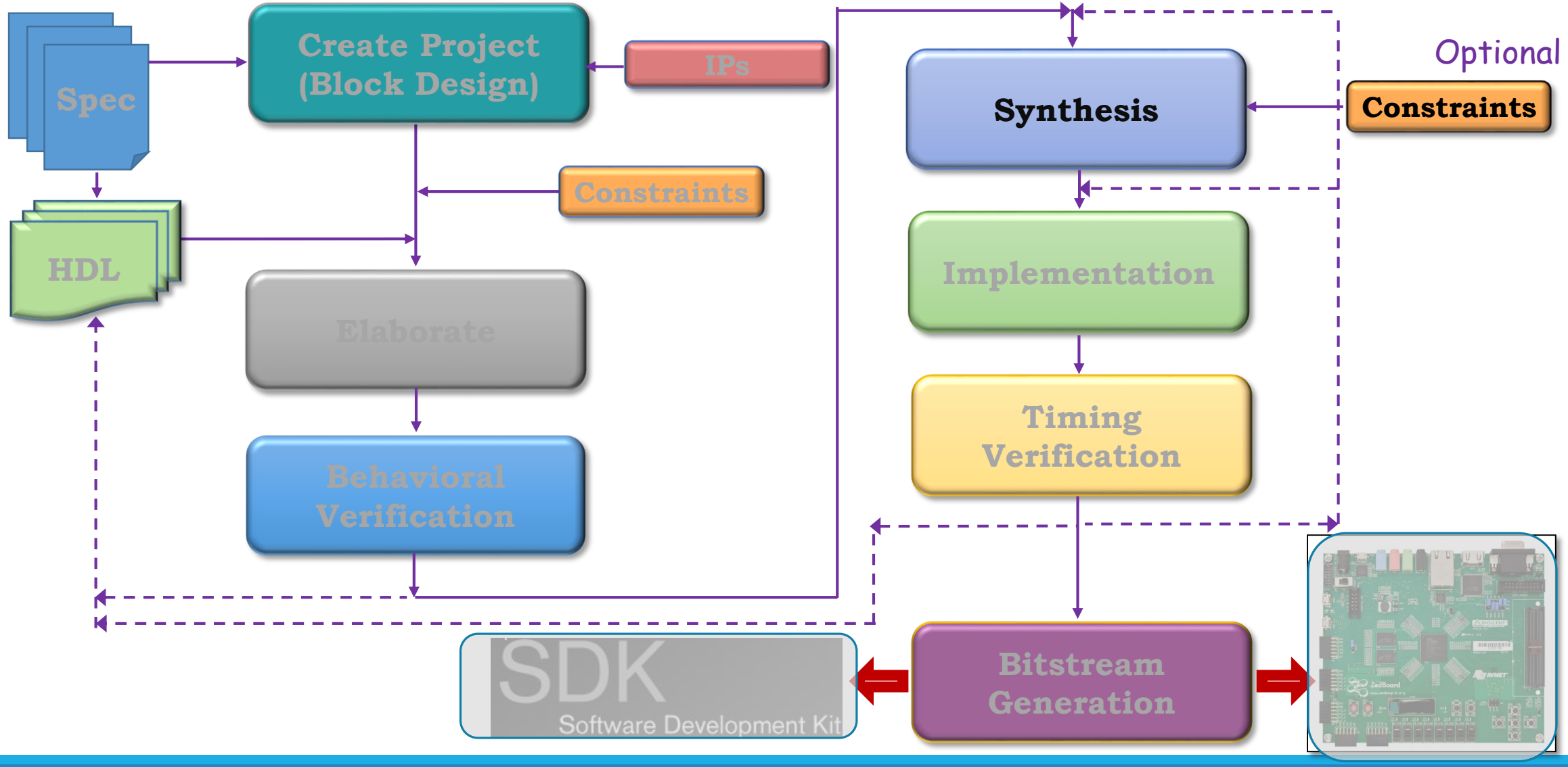


Elaboration

- Elaboration is the RTL optimization to an FPGA technology
- Vivado IDE allows designers to import and manage RTL sources
  - Verilog, System Verilog, VHDL, NGC, or testbenches
- Create and modify sources with the RTL Editor
  - Cross-selection between all the views
- Sources view
  - Hierarchy view: Display the modules in the design by hierarchy
  - Libraries view: Display sources by category

Vivado Design Suite
Synthesis Process

Embedded System Design – Vivado Flow

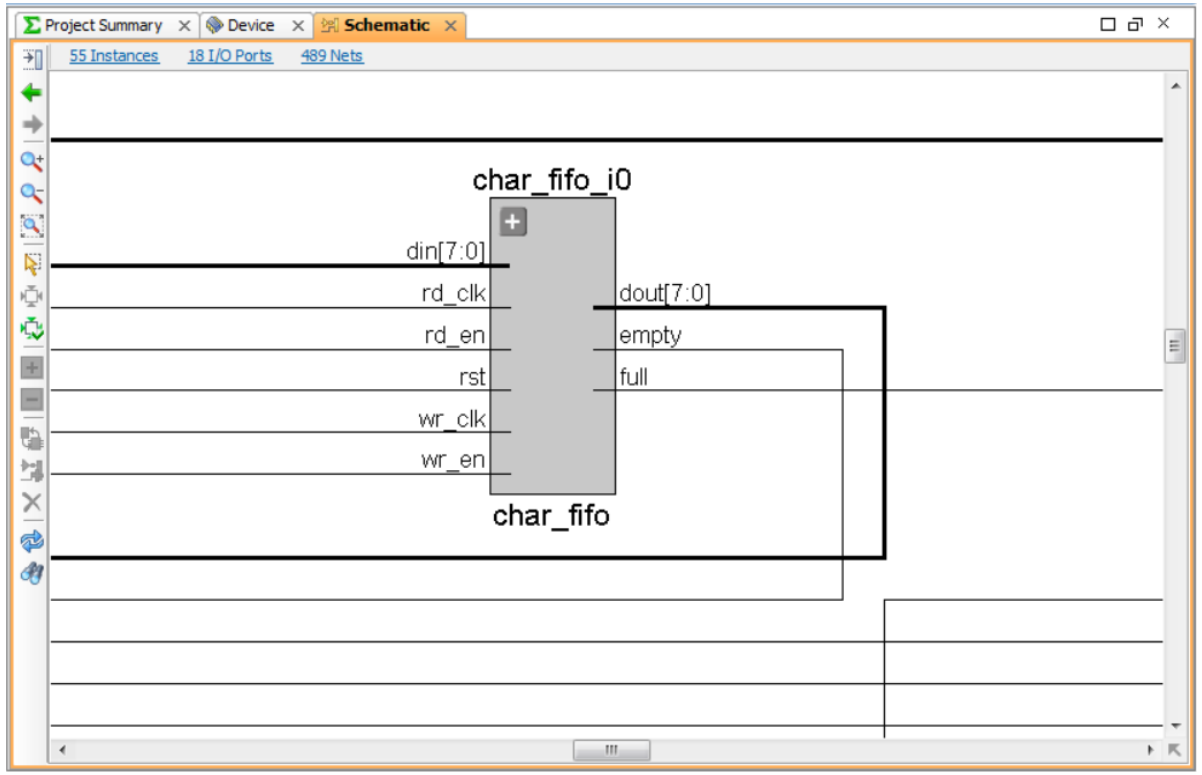


Vivado IDE Synthesis

- **Applicable only for RTL (HDL) design flows**
  - EDIF is black boxed and linked after synthesis
- **Synthesis tool uses XDC constraints to drive synthesis optimization**
  - Design must first be synthesized without timing constraints for constraints editor usage
  - XDC file must exist
- **Synthesis settings provide access to additional options**

Logic Optimization and Mapping to Device Primitives

Synthesis of an RTL design not only optimizes the gate-level design but also maps the netlist to Xilinx primitives (sometimes called technology mapping)



Synthesized Design

Accessed through the Flow Navigator by selecting Open Synthesized Design

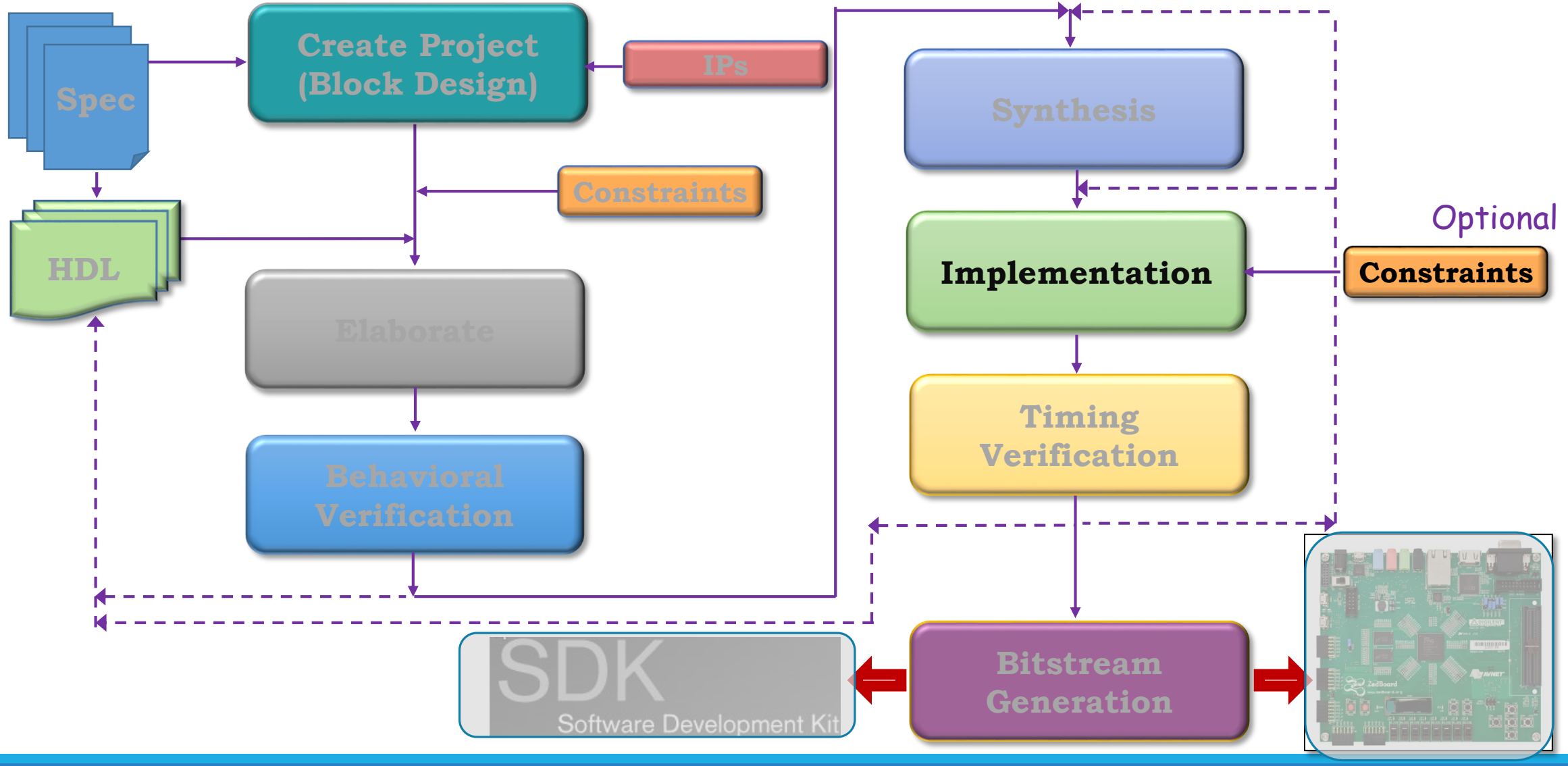
Representation of the design after synthesis

- Interconnected netlist of hierarchical and basic elements (BELs)
  - Instances of modules/entities
  - Basic elements
    - LUTs, flip-flops, carry chain elements, wide MUXes
    - Block RAMs, DSP cells
    - Clocking elements (BUFG, BUFR, MMCM, …)
    - I/O elements (IBUF, OBUF, I/O flip-flops)

Object names are the same as names in the elaborated netlist when possible

Vivado Design Suite
Implementacion Process

Embedded System Design – Vivado Flow



Vivado Implementation Sub-Processes

Vivado Design Suite Implementation process transform a logical netlist (generated by the synthesis tool) into a placed and routed design ready for bitstream generation

- **Opt design**
  - Optimizes the logical design to make it easier to fit onto the target FPGA
- **Place design**
  - Places the design onto the FPGA's logic cells
- **Route design**
  - Routing of connections between the FPGA's cells

Using Design Constraints for Guiding Implementation

There are two types of design constraints, *physical constraints* and *timing constraints*.

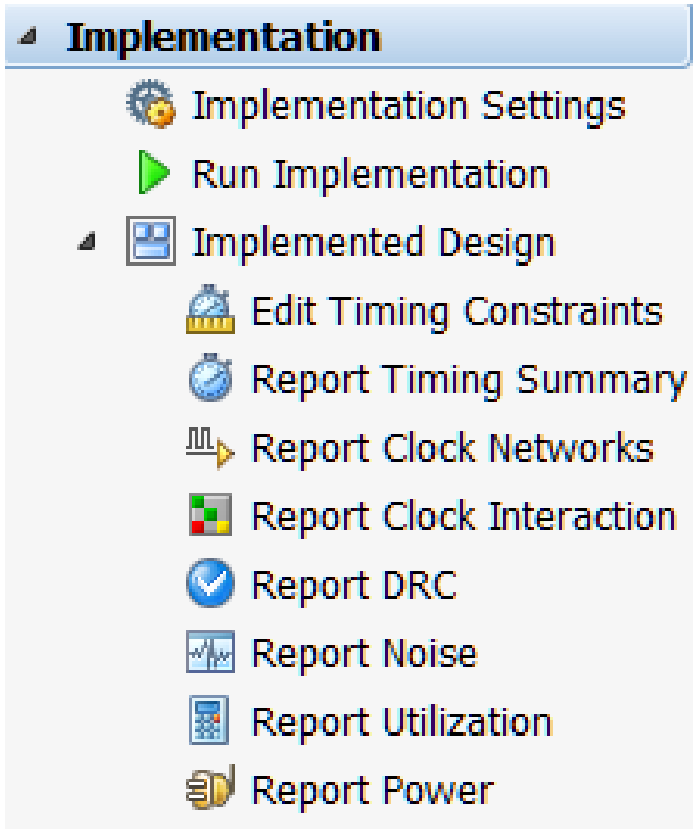
Physical Constraints: define a relationship between logic design objects and device resources

- Package pin placement
- Absolute or relative placement of cells:
  - Block RAM
  - DSP
  - LUTs
  - Filp-Flops
- Floorplanning constraints that assign cells to general regions of an FPGA

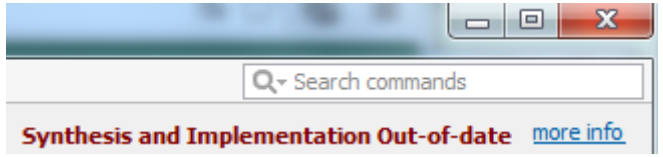
Timing Constraints: define the frequency requirements for the design. Without timing constraints, Vivado Design Suite optimizes the design solely for wire length and routing congestion and makes no effort to asses or improve design performance

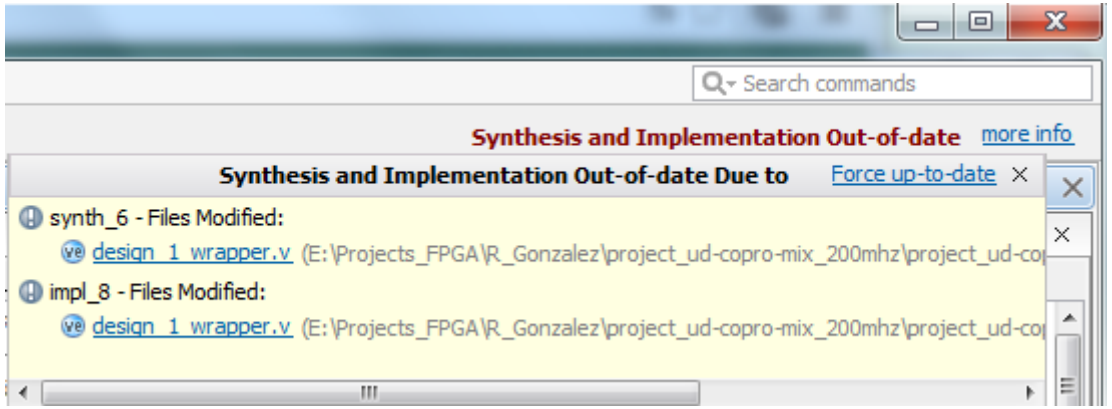
After Implementation

- Sources and Netlist tabs do not change
- Now as each resources is selected, it will show the exact placement of the resource on the die
- Timing results have to be generated with the Report Timing Summary
- As each path is selected, the placement of the logic and its connections is shown in the Device view
- This is the cross-probing feature that helps with static timing analysis



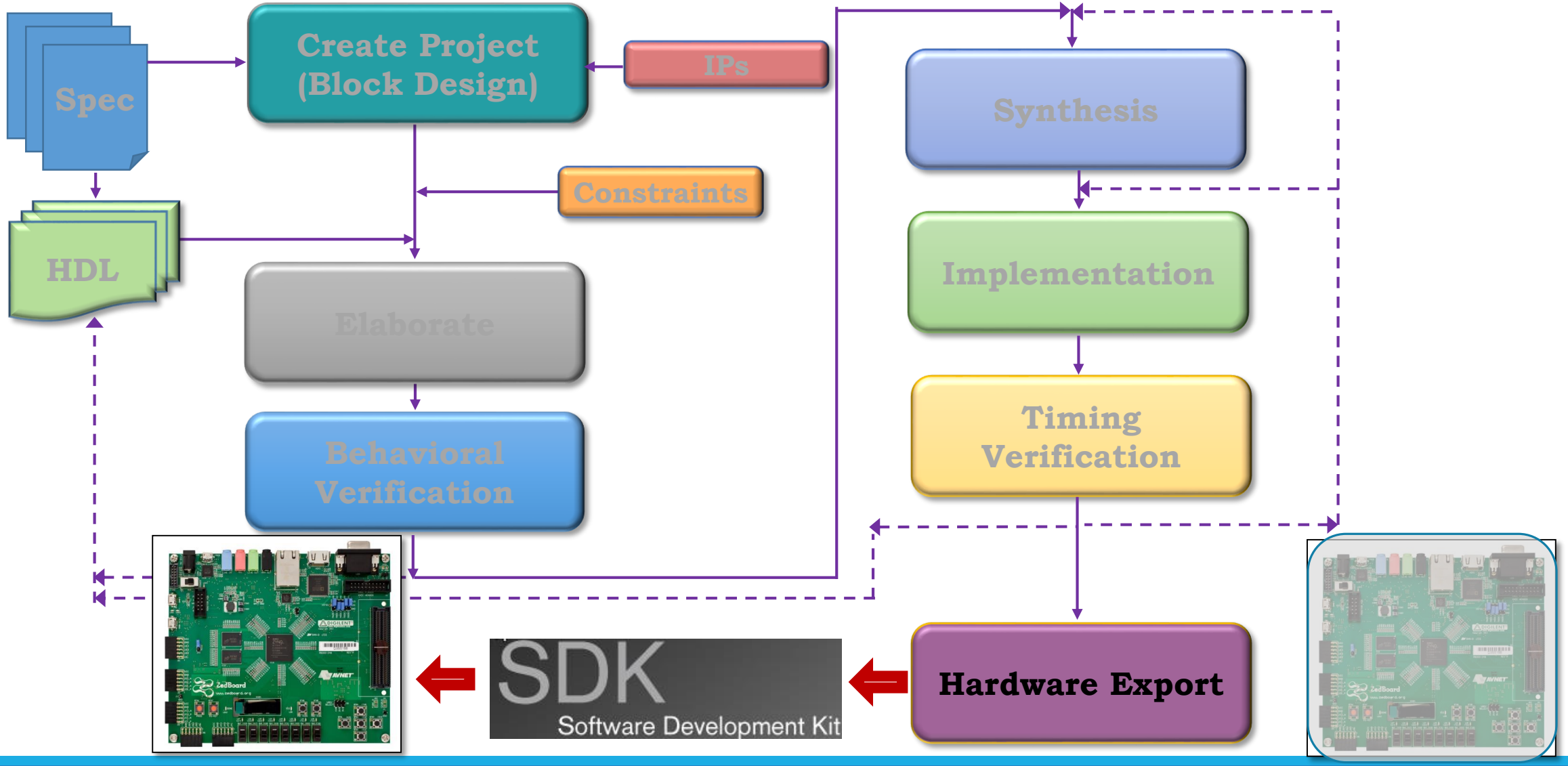
Implementation Out-of-Date Message





Software Development Kit (SDK)

Embedded System Design – Vivado-SDK Flow



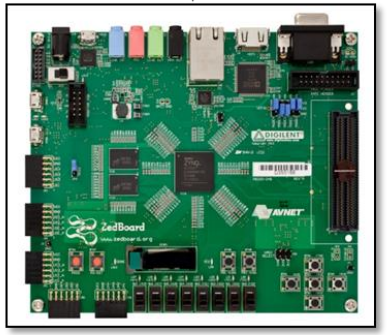
Embedded System Design – Vivado-SDK Flow

VIVADO

- Open VIVADO
- Create New Project
- Create New Block Design
- Add PS7
  Configure PS7
  Run Block Automation
  Add and Configure other IPs
  Run Connection Automation
  Validate Design
- Create HDL Wrapper
- Generate Output Files
- Generate Bitstream
- Export Hardware to SDK

SDK
Software Development Kit

- Create Board Support Package
- Create New 'C' Application
- Write 'C' Code
- Build the Application
- Configure FPGA
- Run on Hardware (Debug)



C7 Technology ©

Embedded System Tools: Software

Eclipse IDE-based Software Development Kit (SDK)

- Board support package creation : LibGen
- GNU software development tools
- C/C++ compiler for the ARM Cortex-A9 processor (gcc)
- Debugger for the ARM Cortex-A9 processor (gdb)

Board support packages (BSPs)

- Stand-alone BSP
  - Free basic device drivers and utilities from Xilinx
  - NOT an RTOS

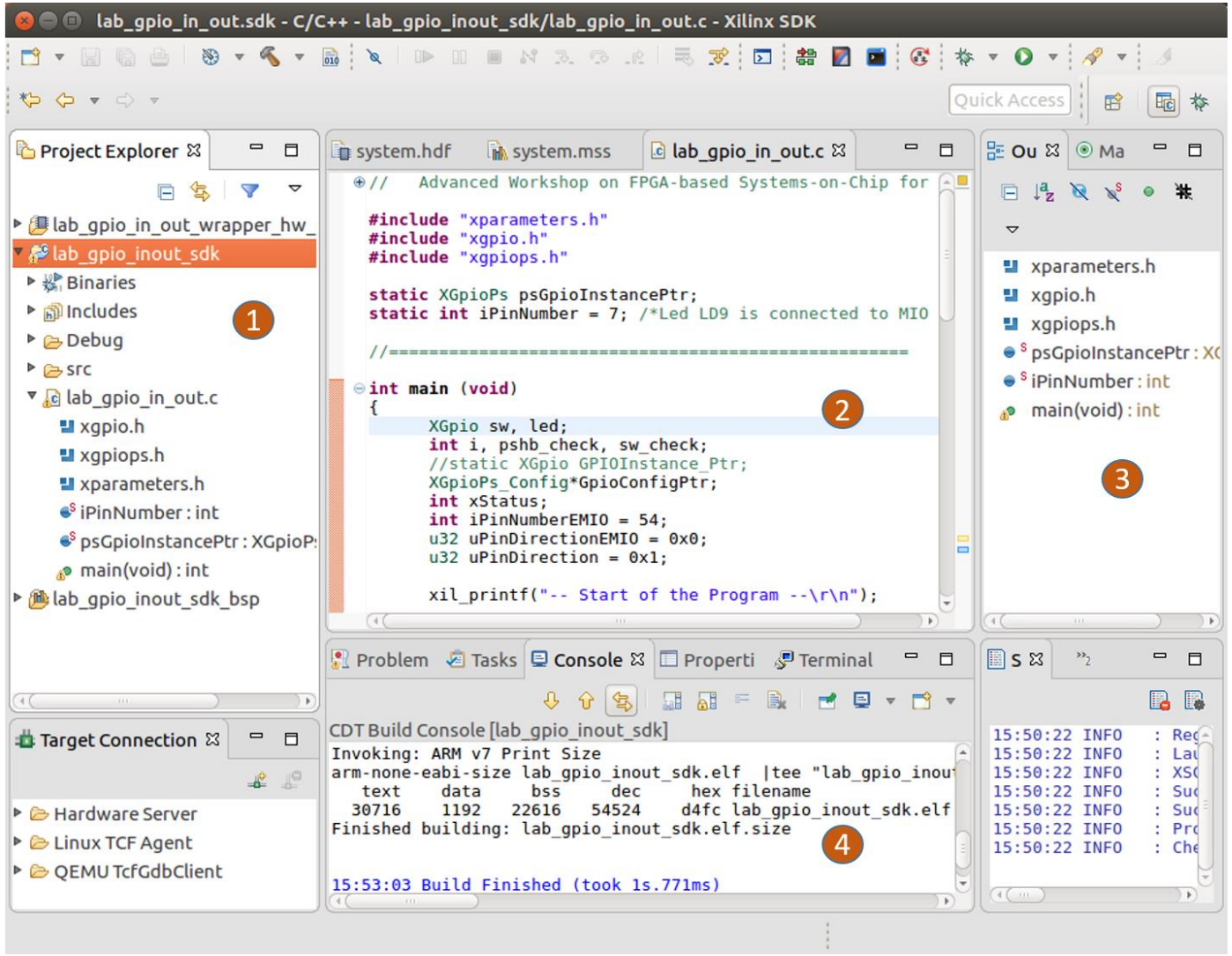
SDK Workbench Views

① C/C++ project outline displays the elements of a project with file decorators (icons) for easy identification

② C/C++ editor for integrated software creation

③ Code outline displays elements of the software file under development with file decorators (icons) for easy identification

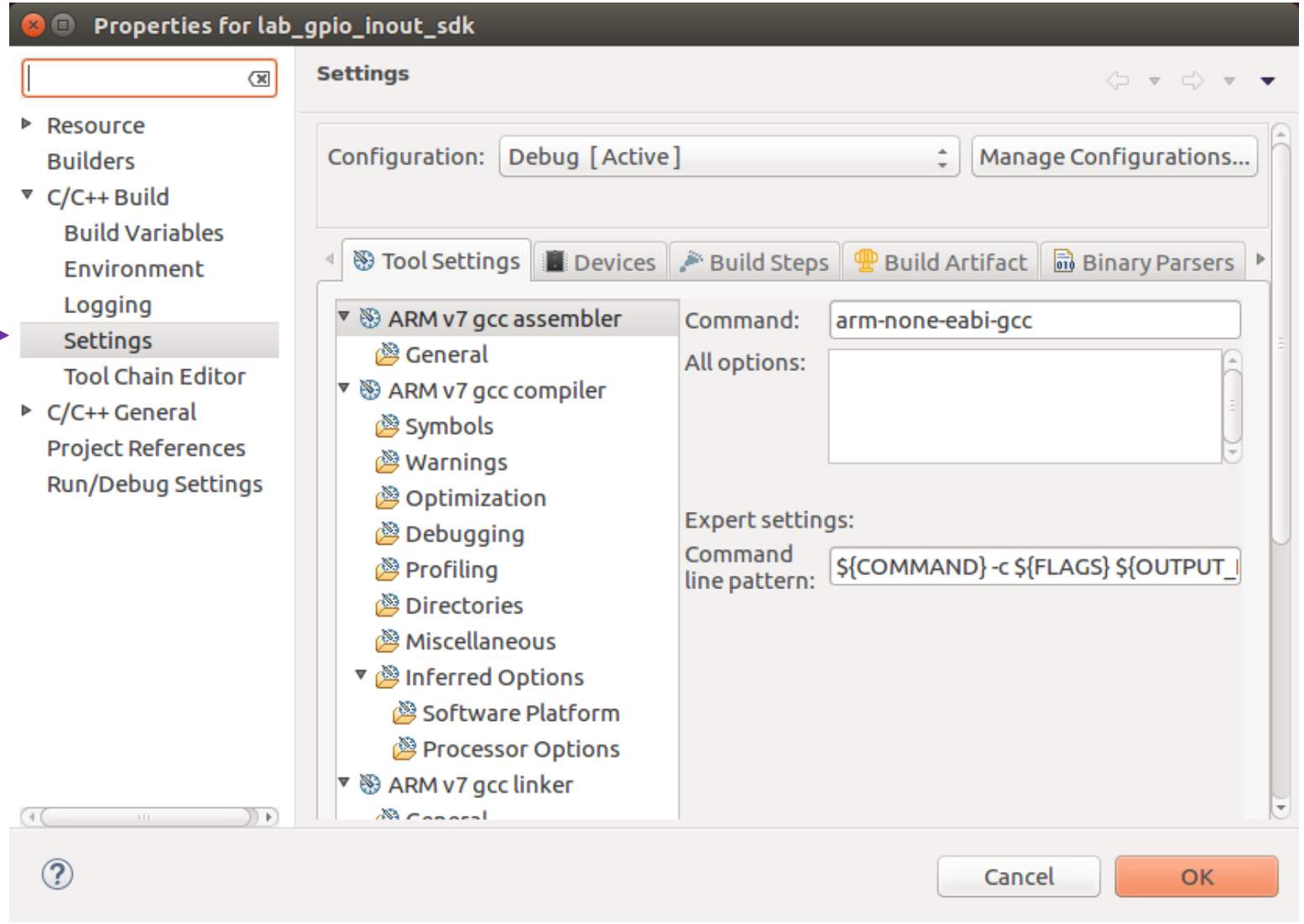
④ Problems, Console, Properties views list output information associated with the software development flow



Software Management Settings

Software is managed in three major areas

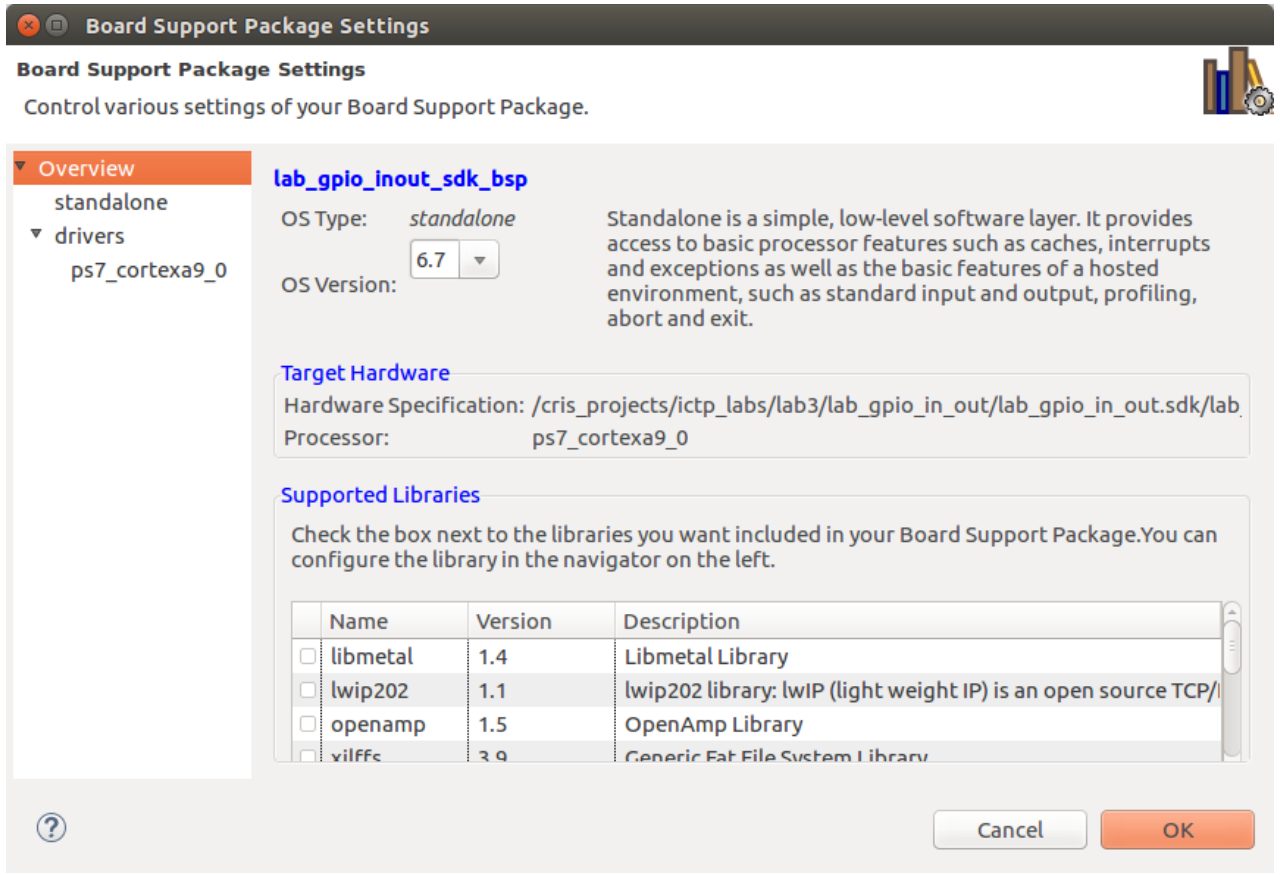
- Compiler/Linker Options
  - Application program



Software Management Settings

Software is managed in three major areas

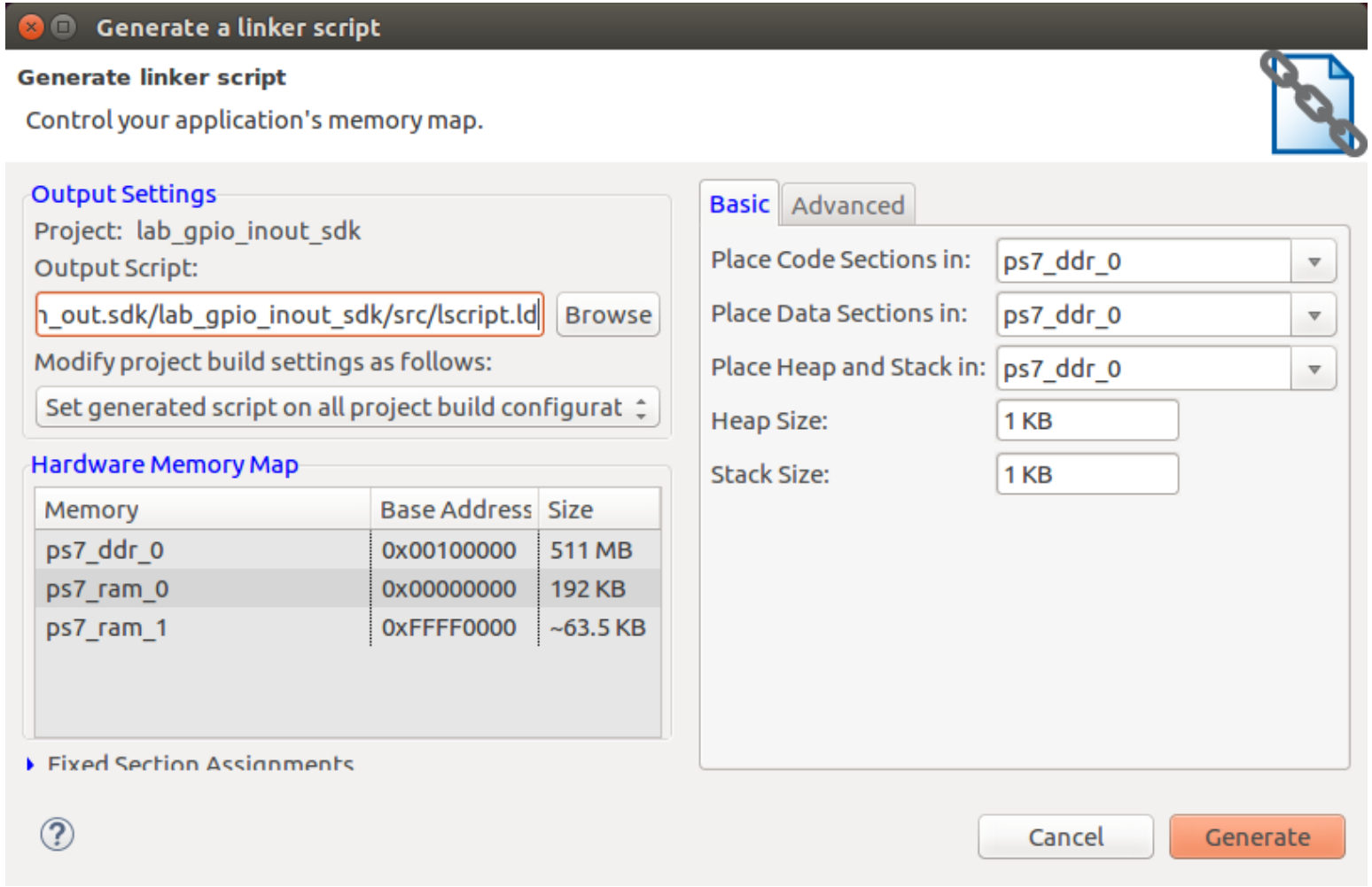
- Software Platform Settings
  - Board support package



Software Management Settings

Software is managed in three major areas

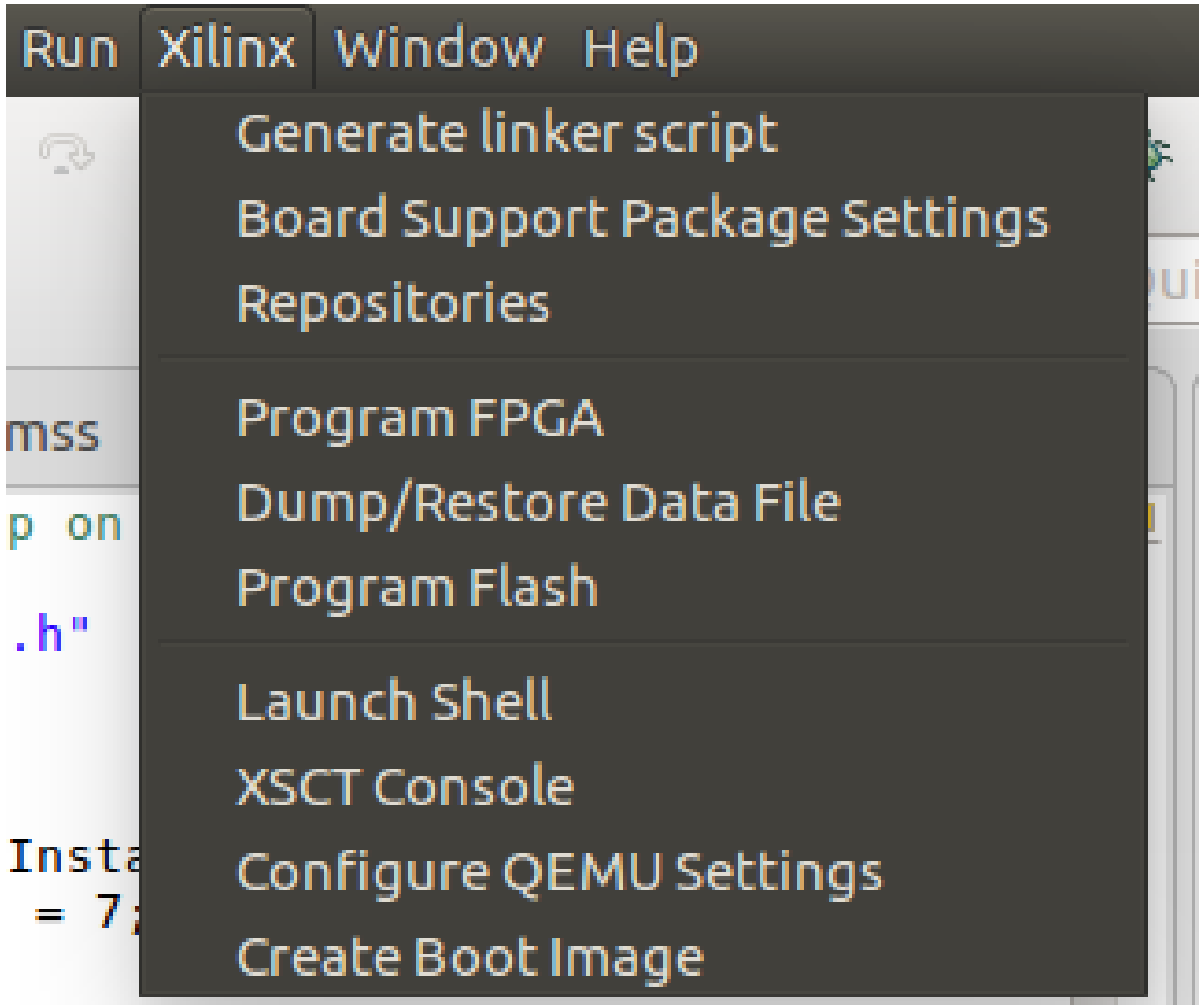
- Software Platform Settings
  - Board support package



Integrated Xilinx Tools in the SDK

Xilinx additions to the Eclipse IDE

- BSP Settings
- Software Repositories
- Generate Linker Script
- Program the programmable logic
  - Bitstream must be available
- Create Zynq Boot Image
- Program Flash Memory
- Launch XMD Console
- Launch Shell
- Configure JTAG Settings
- SysGen Co-Debug Settings



Apendix

Vivado Design Suite

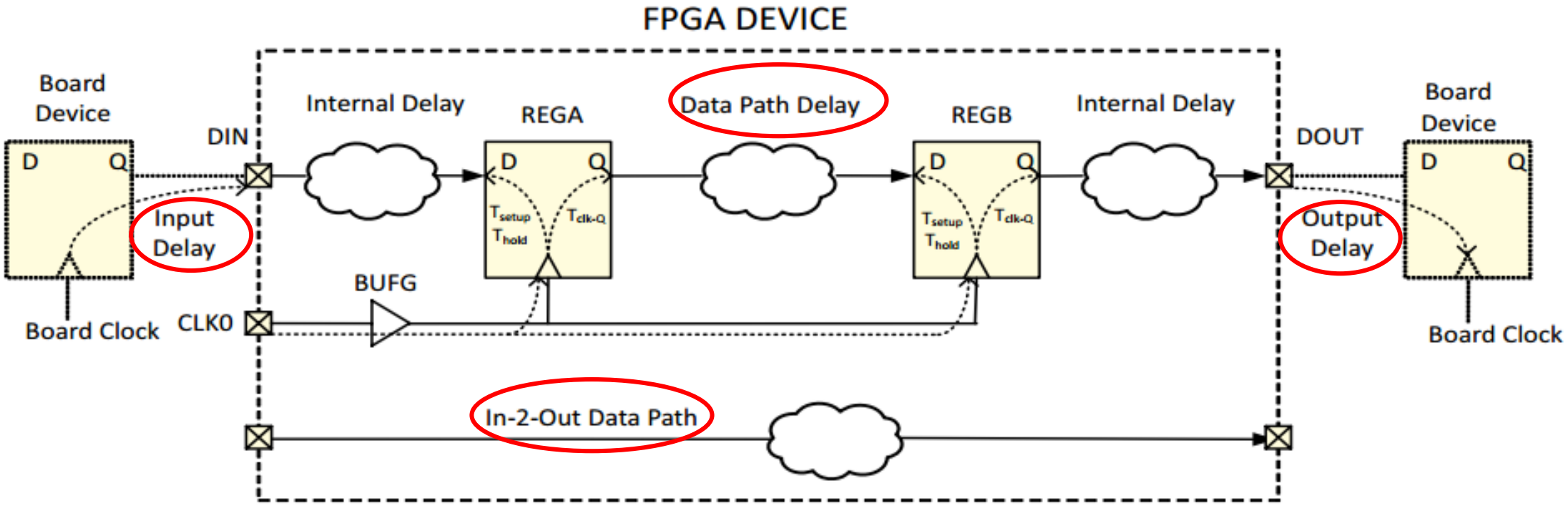
Basic Static Timing Constraints

Basic Timing Constraints

There are three basic timing constraints applicable to a sequential machine

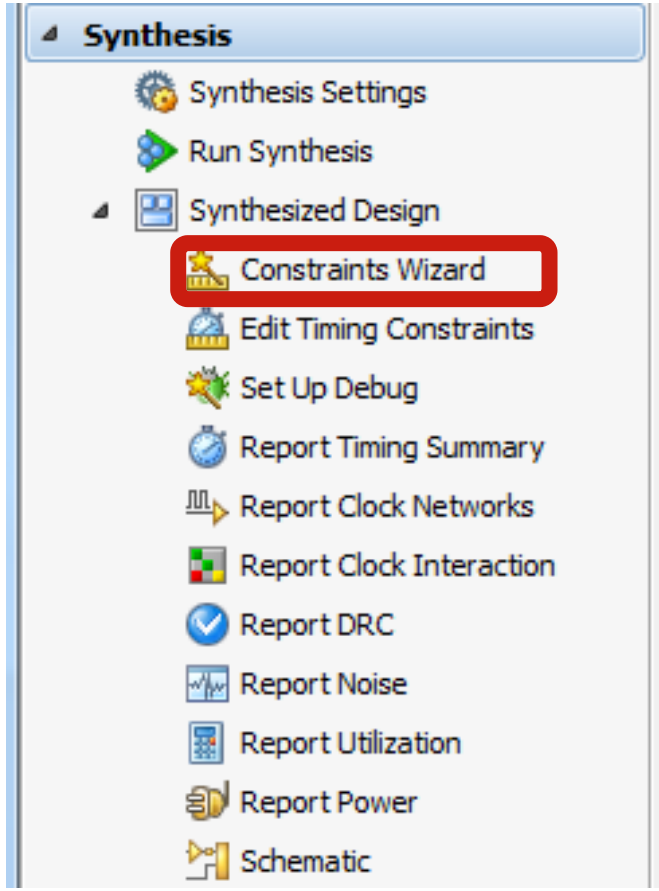
- Period
  - Paths between synchronous elements clocked by the reference clock net
    - Synchronous elements include flip-flops, latches, synchronous RAM, and DSP slices
  - Use `create_clock` to create the constraint
- Input Delay
  - Paths between input pin and synchronous elements
  - Use `set_input_delay` to create the constraint
- Output delay
  - Paths between synchronous elements and output pin
  - Use `set_output_delay` to create the constraint

Timing Paths Example

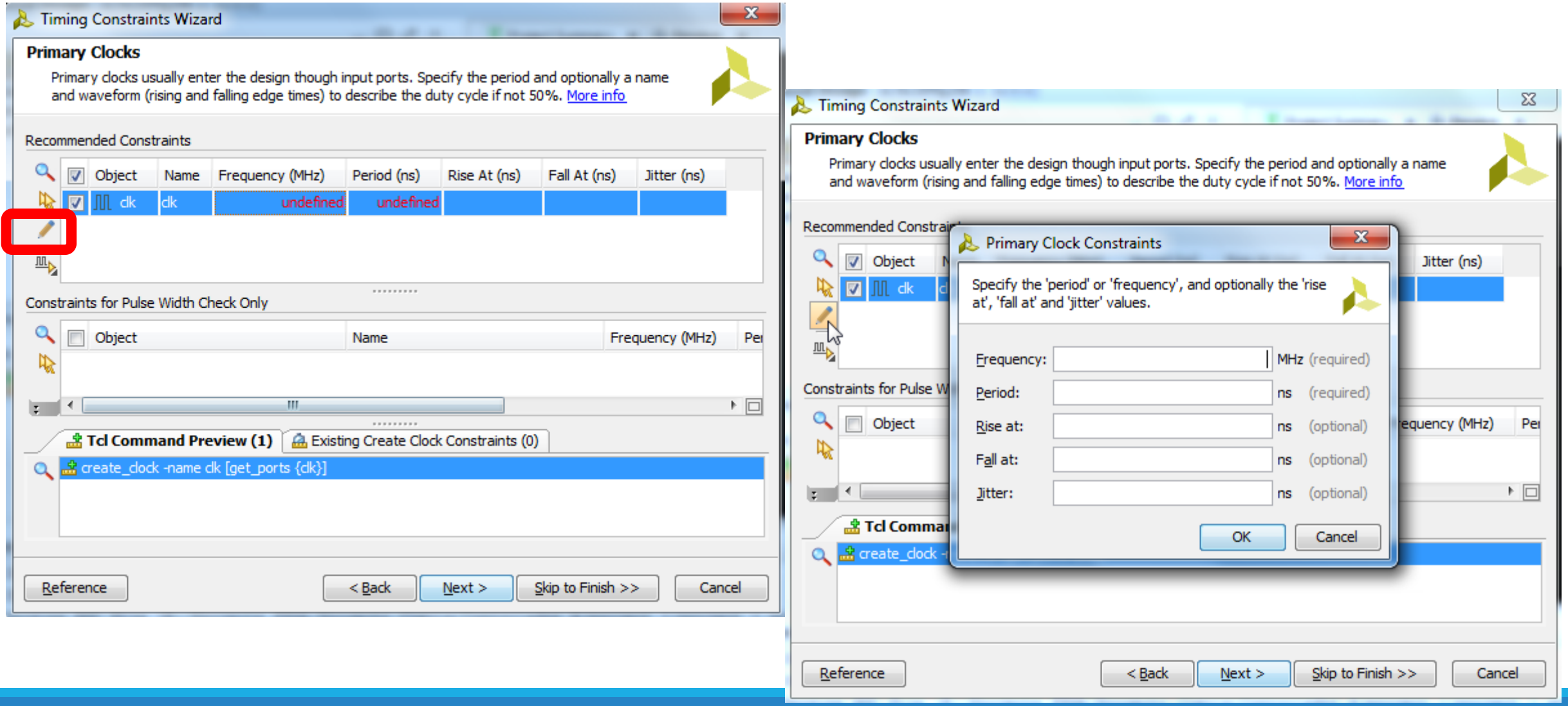


Creating Basic Timing Constraints in Vivado IDE

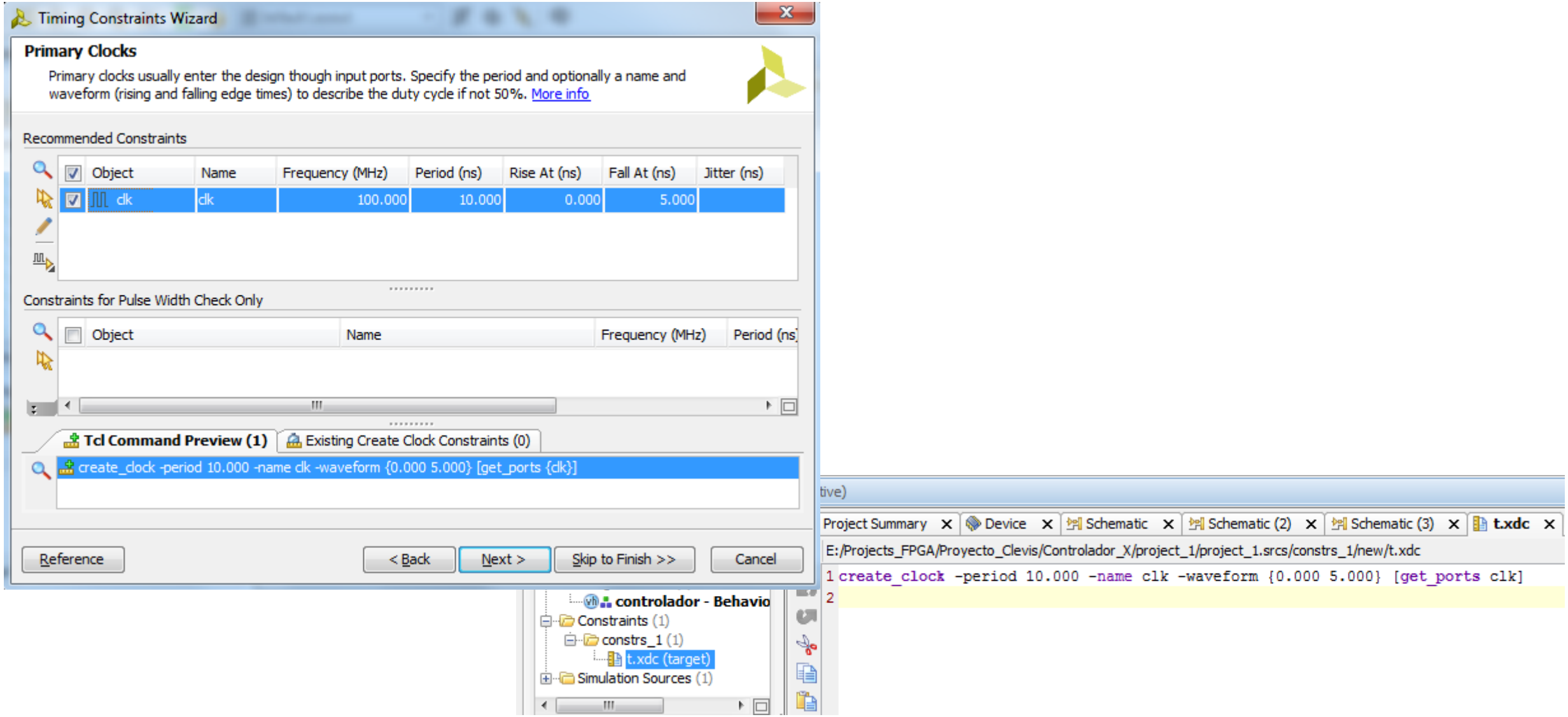
1. Run Synthesis
2. Open the synthesized design
3. Invoke constraints editor



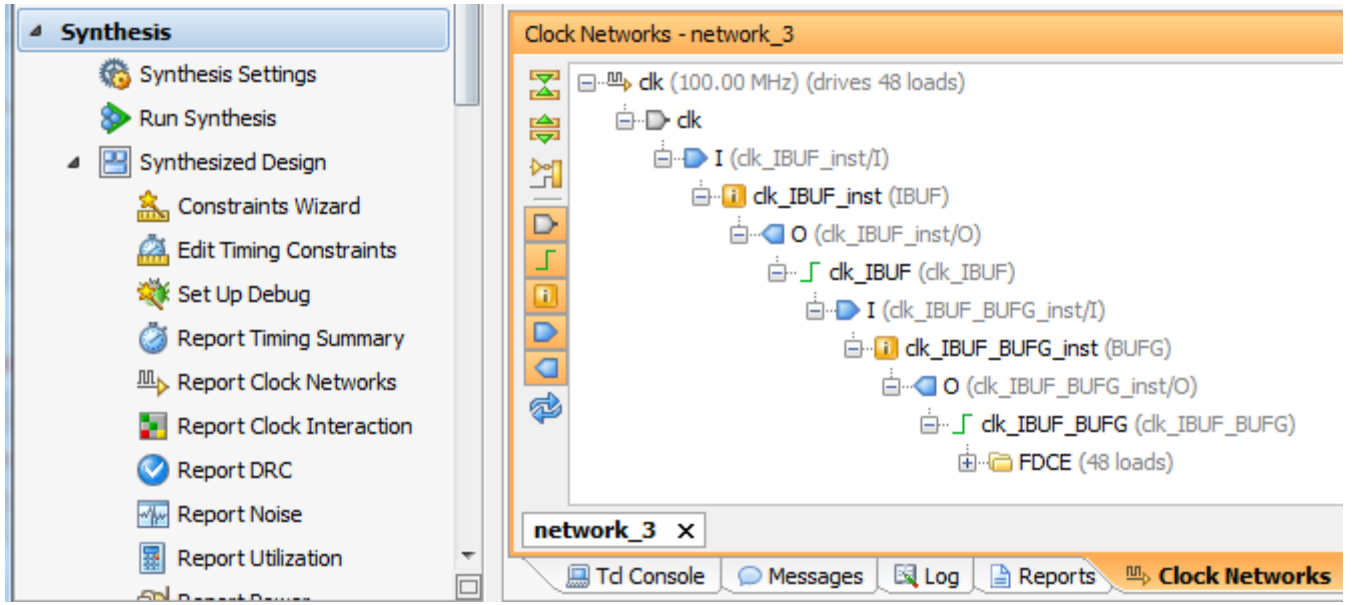
Clock Constraint Setting



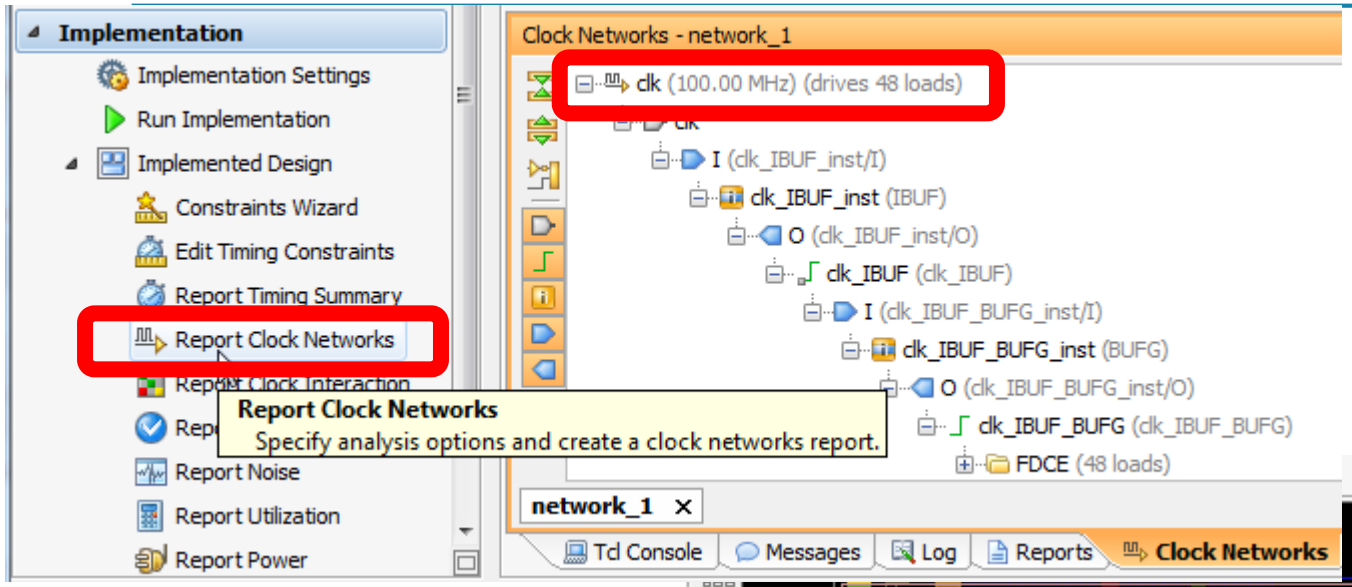
Clock Constraint Setting

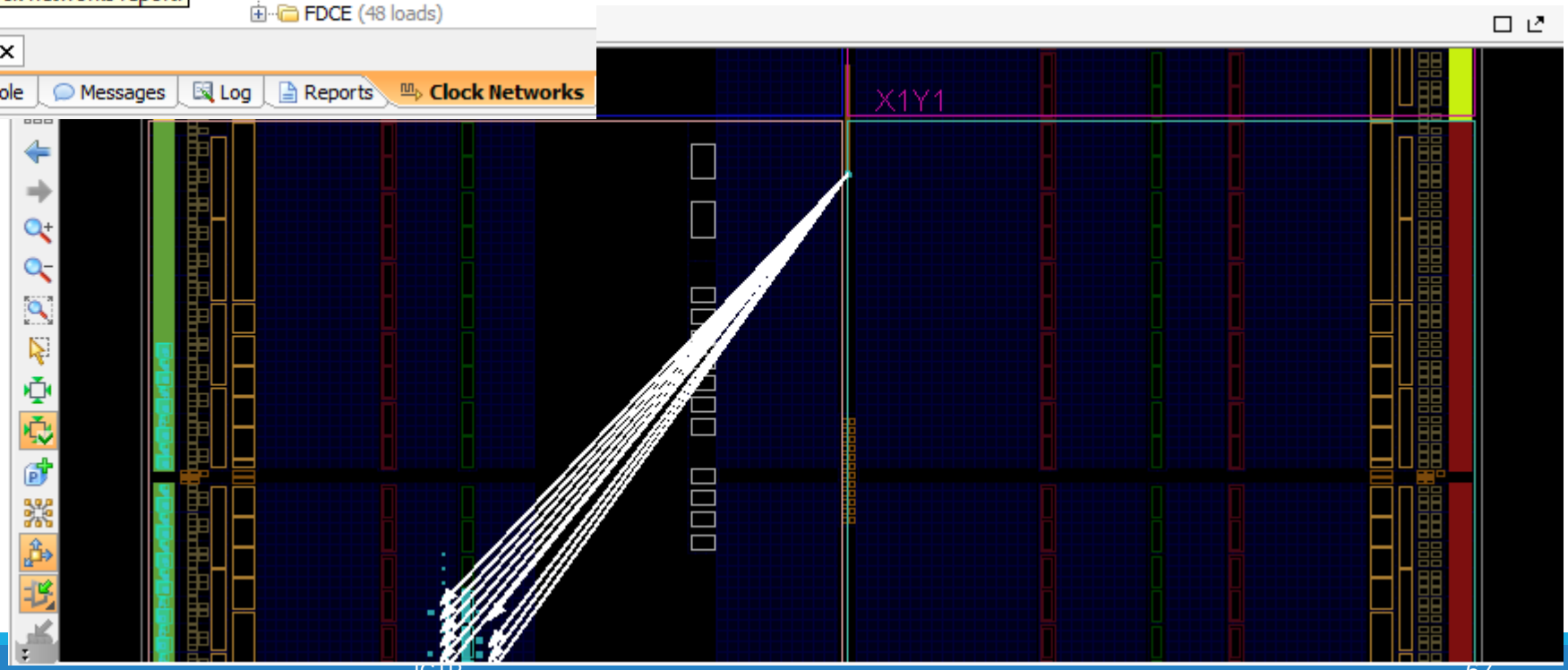


Clock Network Report



Clock Network Report and Visualization





Clocking Resources: MMCM and PLL

Up to 24 CMTs per device

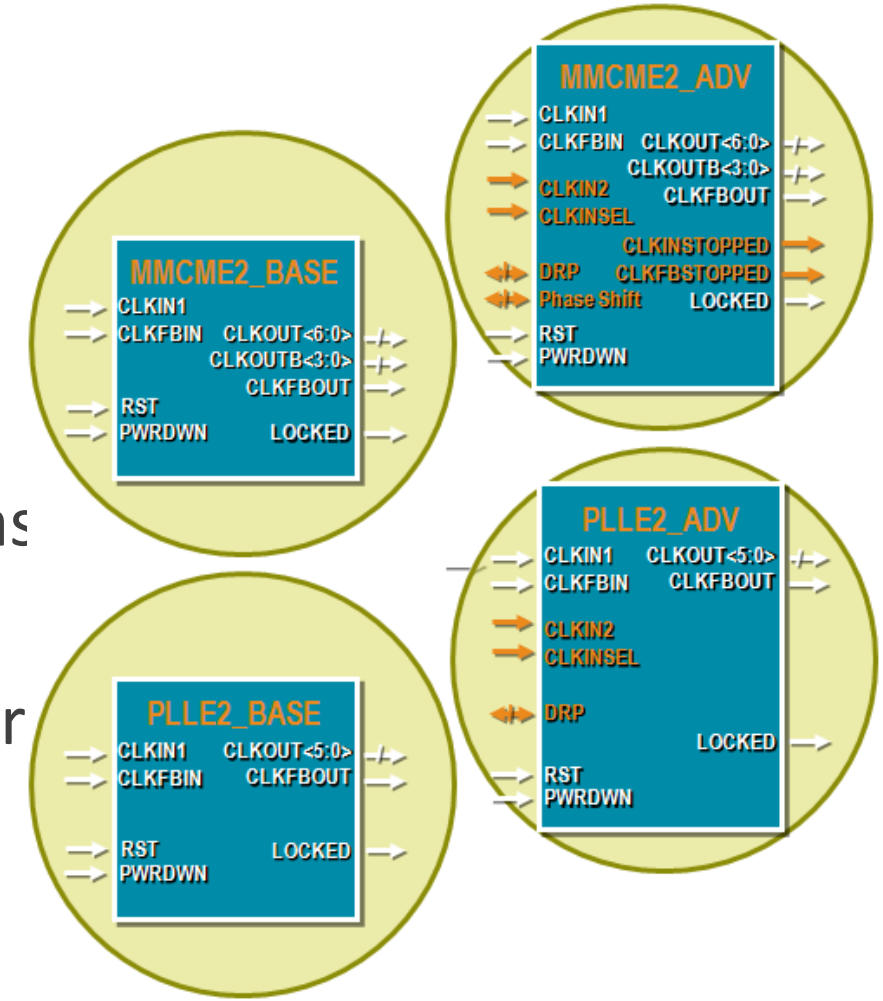
One MMCM and one PLL per CMT

Two software primitives (instantiation)

- *_BASE has only the basic ports
- *_ADV provides access to all ports

PLL is primarily intended for use with the I/O phas
for high-speed memory controllers

The MMCM is the primary clock resource for user



Inference

Clock networks are represented by nets in your RTL design
- The mapping of an RTL net to a clock network is managed by using the appropriate clock buffer to generate that net

Certain resources can be inferred
- A primary input net (with or without an IBUF instantiated) will be mapped to a global clock if it drives the clock inputs of clocked resources
  - The BUFG will be inferred
- BUFH drivers will be inferred whenever a global clock (driven by a BUFG) is required in a clock region
  - BUFHs for each region required will be inferred

BUFIO, BUFR, and BUFMR cannot be inferred
- Instantiating these buffers tells the tools that you want to use the corresponding clock networks

PLLs and MMCMs cannot be inferred

Instantiation

All clocking resources can be directly instantiated in your RTL code

- Simulation models exist for all resources
- Refer to the Library Guide for HDL Designs
- Use the Language Templates () tab

PLLs and MMCMs have many inputs and outputs, as well as many attributes

- Optimal dividers for obtaining the desired characteristics may be hard to derive
- The Clocking Wizard via the IP Catalog
  - Only *_ADV available

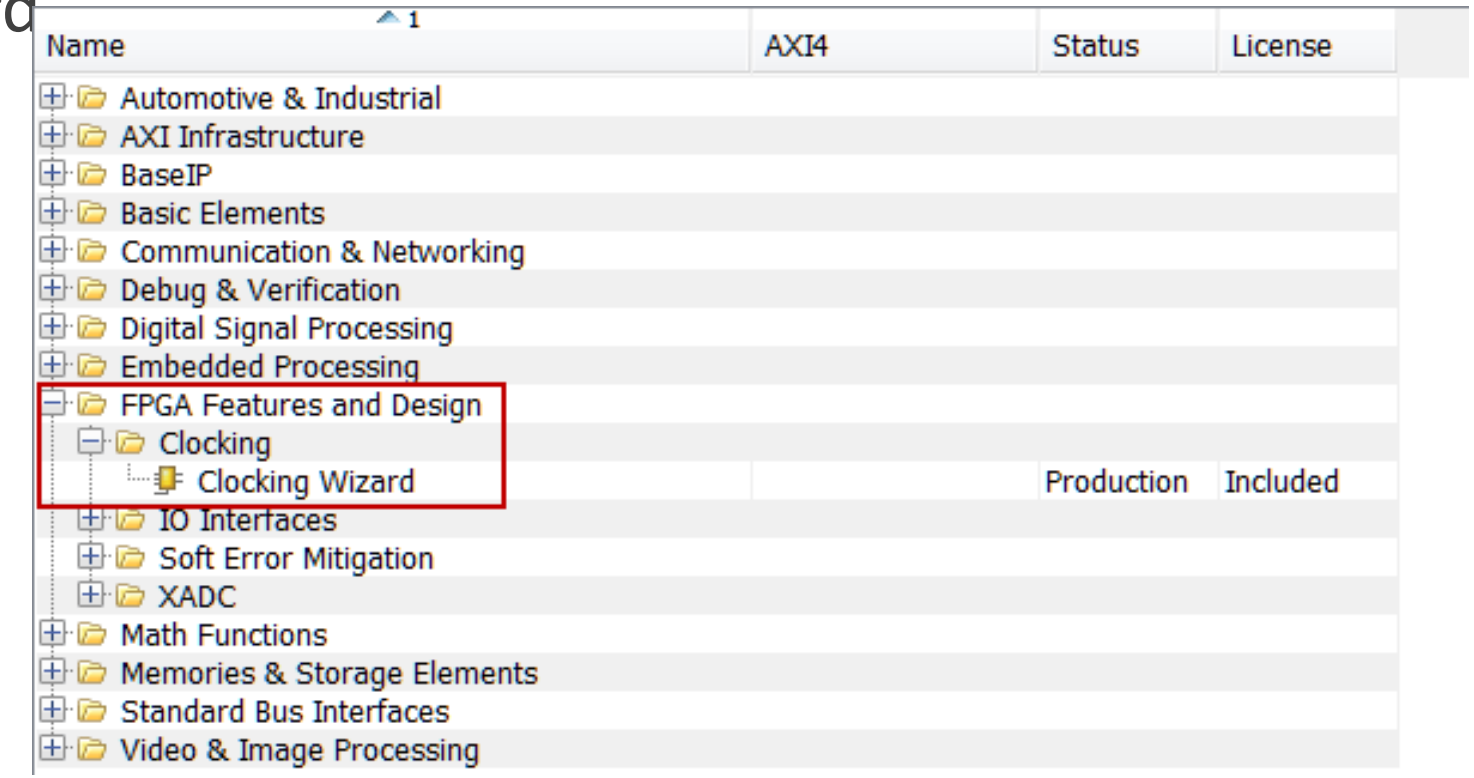
Invoking Clocking Wizard

Click on the IP Catalog

Expand FPGA Features and Design > Clocking

Double-click on Clocking Wizard

The Clocking Wizard walks you through the generation of complete clocking subsystems

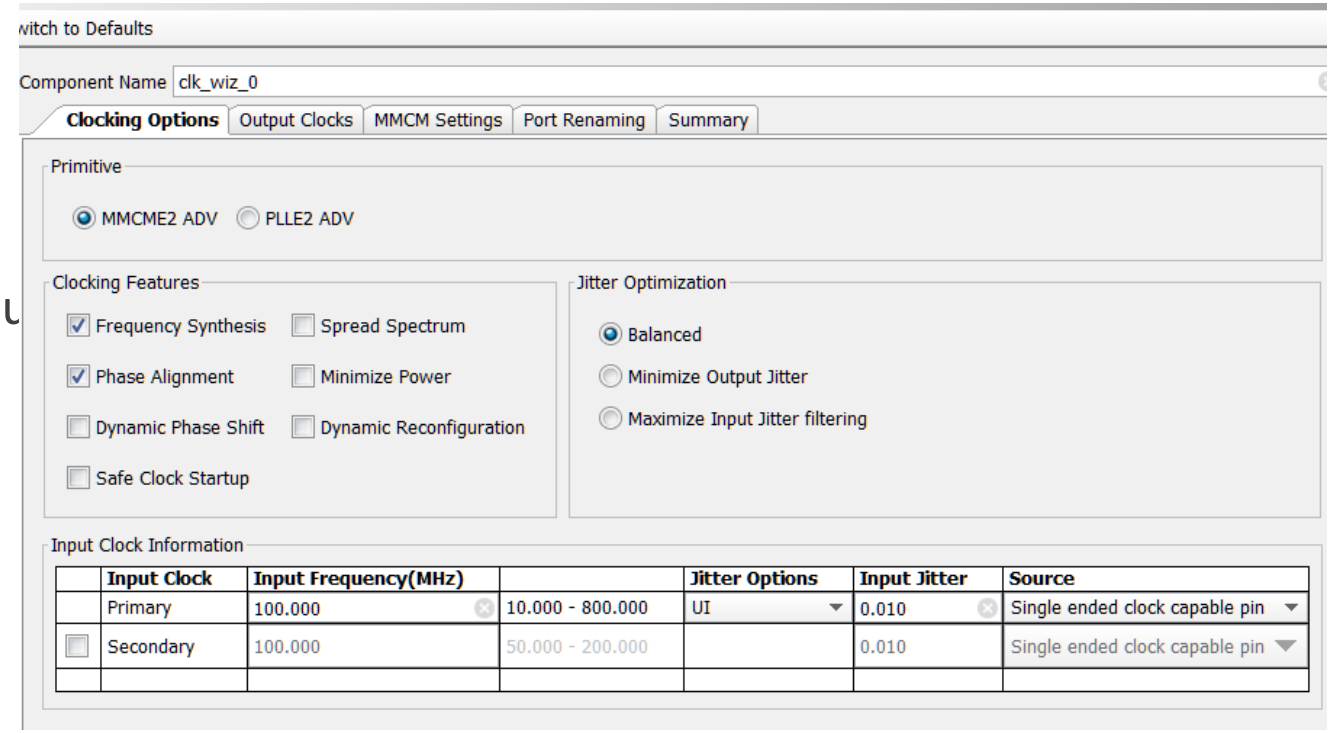


The Clocking Wizard: Clocking Options

Select Primitives to be used
- MMCME2_ADV
- PLLE2_ADV

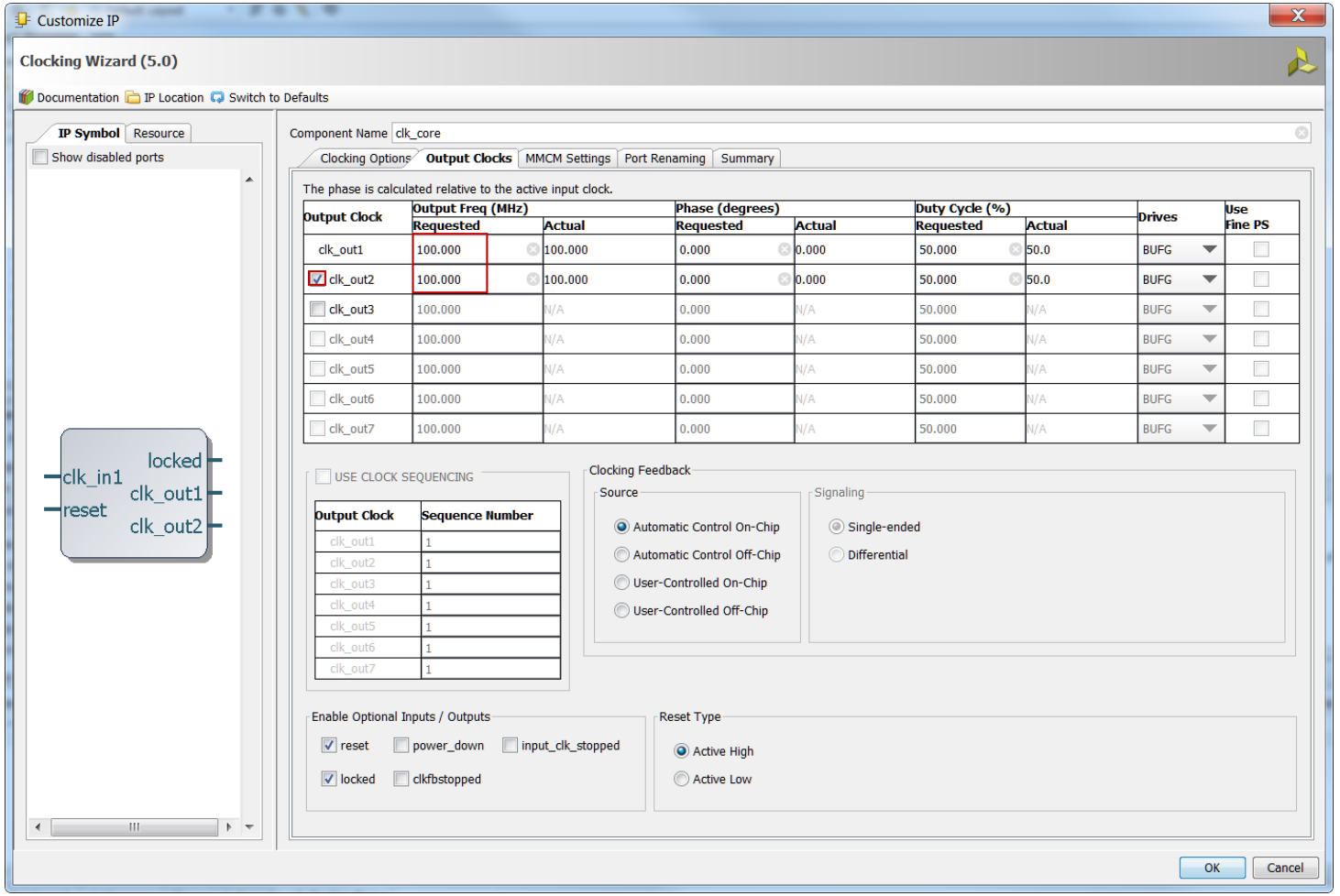
Specify the primary input frequency and source type
- Optionally, select and specify secondary inpu

Select clocking features
- Frequency synthesis
- Phase alignment
- Dynamic phase shift
- …



The Clocking Wizard: Output Clocks

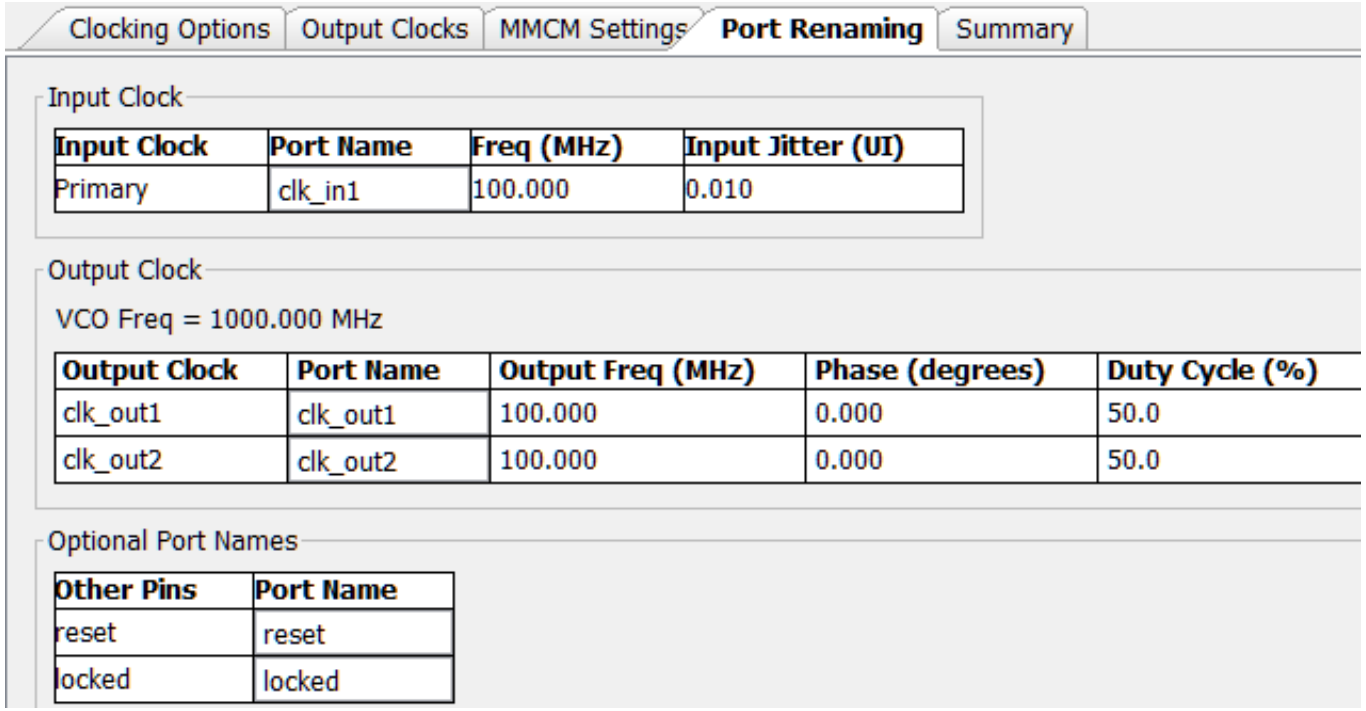
- Select the desired number of output clocks
- Set the desired output frequencies
- Select optional ports



The Clocking Wizard: Port Renaming

Change input/output port names

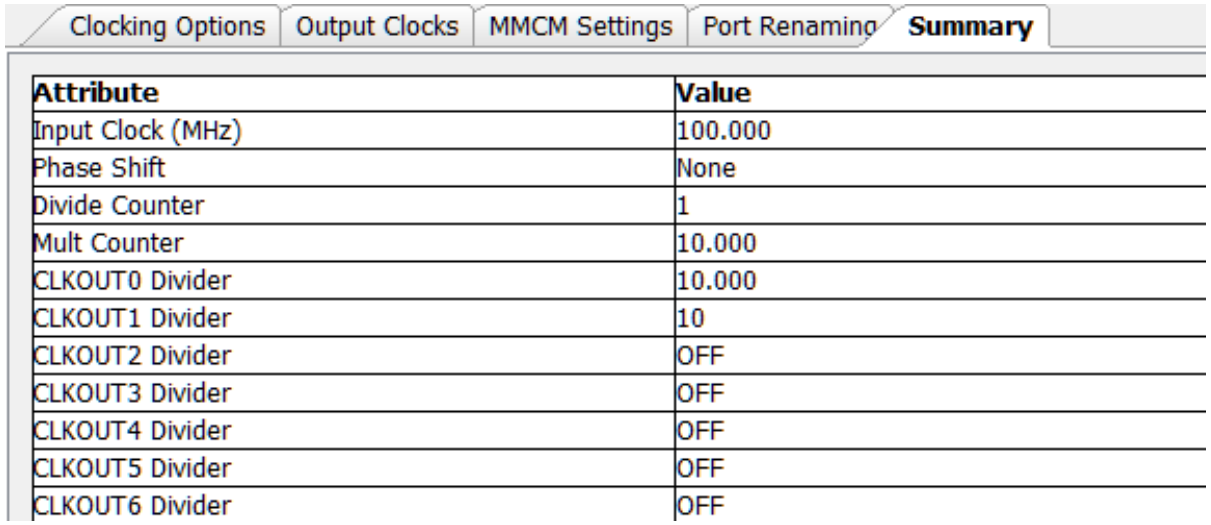
Change optional port names



The Clocking Wizard: Summary

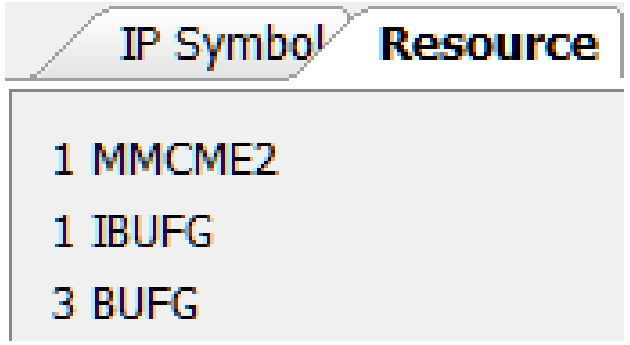
Shows the input, output frequencies

Other attributes depending on the selections made



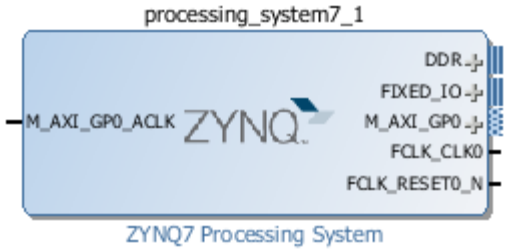
Attribute	Value
Input Clock (MHz)	100.000
Phase Shift	None
Divide Counter	1
Mult Counter	10.000
CLKOUT0 Divider	10.000
CLKOUT1 Divider	10
CLKOUT2 Divider	OFF
CLKOUT3 Divider	OFF
CLKOUT4 Divider	OFF
CLKOUT5 Divider	OFF
CLKOUT6 Divider	OFF

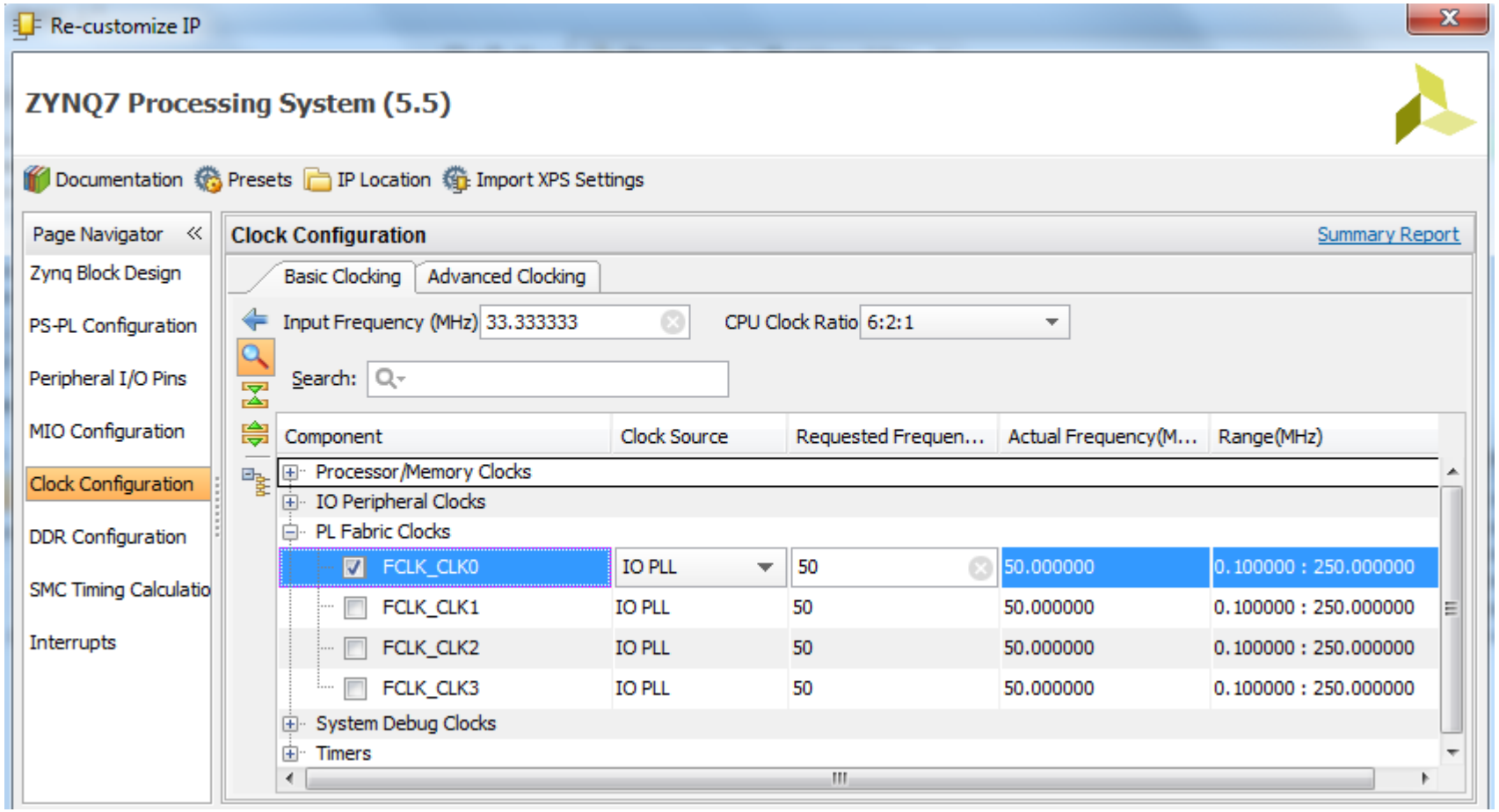
The Resource tab on the left provides summary of type and number of resources used

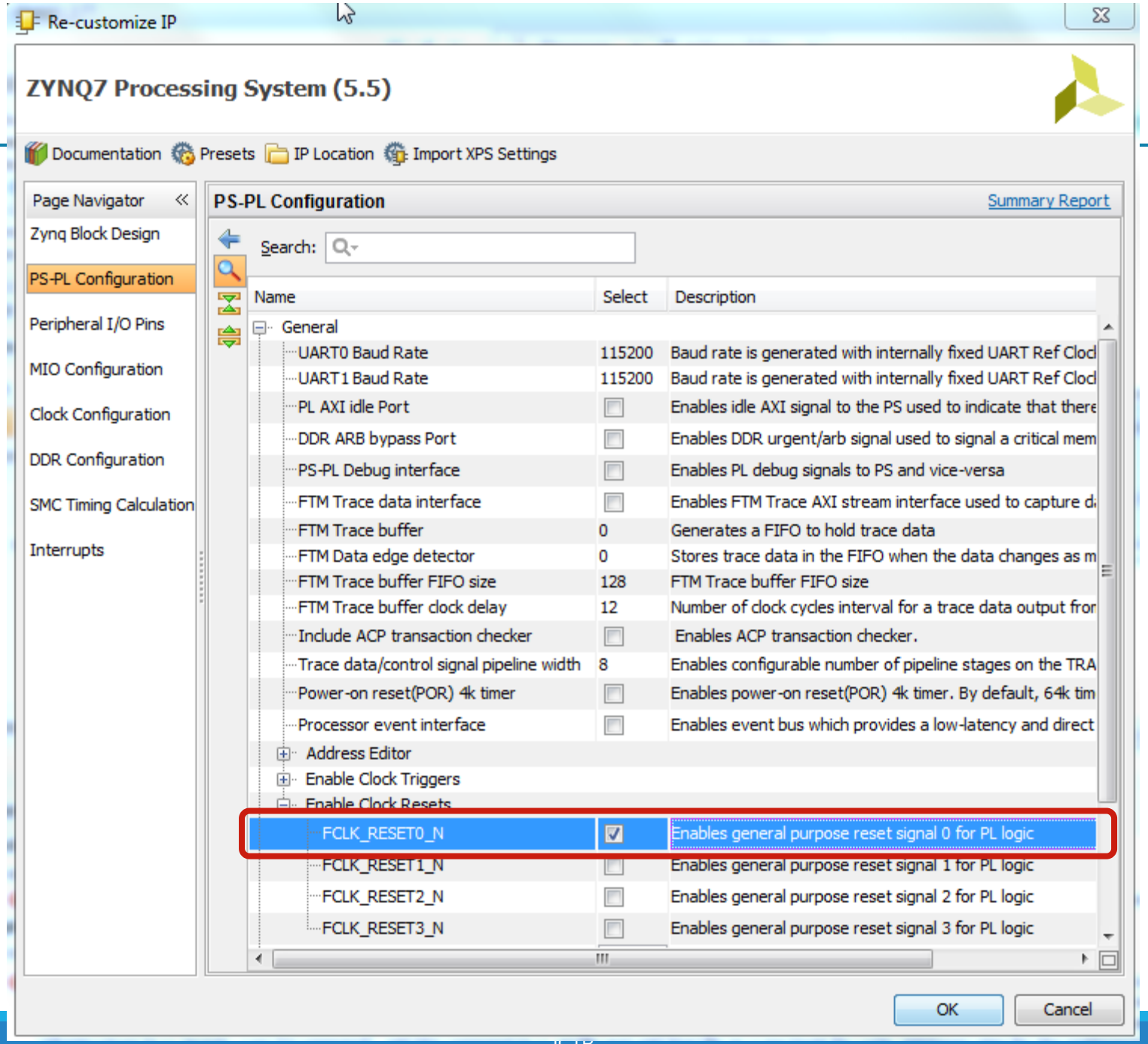


Reset and Clock Topology

Enabling Clock for PL






Re-customize IP
ZYNQ7 Processing System (5.5)
Documentation
Presets
IP Location
Import XPS Settings
Page Navigator
Zynq Block Design
PS-PL Configuration
Peripheral I/O Pins
MIO Configuration
Clock Configuration
DDR Configuration
SMC Timing Calculation
Interrupts
PS-PL Configuration
Summary Report
Search:
Name
Select
Description
General
UART0 Baud Rate
115200
Baud rate is generated with internally fixed UART Ref Clock
UART1 Baud Rate
115200
Baud rate is generated with internally fixed UART Ref Clock
PL AXI idle Port
Enables idle AXI signal to the PS used to indicate that there
DDR ARB bypass Port
Enables DDR urgent/arb signal used to signal a critical mem
PS-PL Debug interface
Enables PL debug signals to PS and vice-versa
FTM Trace data interface
Enables FTM Trace AXI stream interface used to capture da
FTM Trace buffer
0
Generates a FIFO to hold trace data
FTM Data edge detector
0
Stores trace data in the FIFO when the data changes as m
FTM Trace buffer FIFO size
128
FTM Trace buffer FIFO size
FTM Trace buffer clock delay
12
Number of clock cycles interval for a trace data output fror
Include ACP transaction checker
Enables ACP transaction checker.
Trace data/control signal pipeline width
8
Enables configurable number of pipeline stages on the TRA
Power-on reset(POR) 4k timer
Enables power-on reset(POR) 4k timer. By default, 64k tim
Processor event interface
Enables event bus which provides a low-latency and direct
Address Editor
Enable Clock Triggers
Enable Clock Resets
FCLK_RESET0_N
Enables general purpose reset signal 0 for PL logic
FCLK_RESET1_N
Enables general purpose reset signal 1 for PL logic
FCLK_RESET2_N
Enables general purpose reset signal 2 for PL logic
FCLK_RESET3_N
Enables general purpose reset signal 3 for PL logic
OK
Cancel

SDK Compilers

GNU Tools: GCC

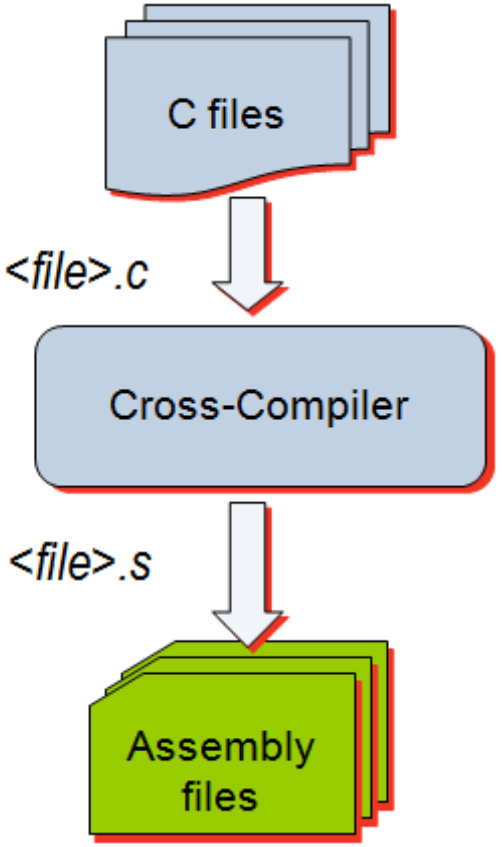
GCC translates C source code into assembly language

GCC also functions as the user interface, passing options to the GNU assembler and to the GNU linker, calling the assembler and the linker with the appropriate parameters

Supported cross-compilers

ARM processor compiler

- GNU GCC (arm-xilinx-eabi-gcc)
- GNU Linux GCC (arm-xilinx-linux-eabi-gcc)



GNU Tools: AS

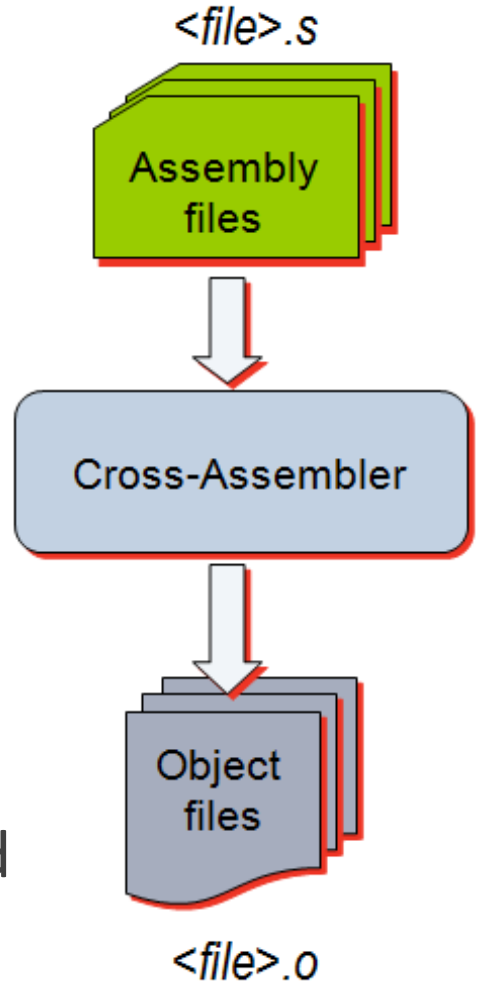
Input: assembly language files
- File extension: .s

Output: object code
- File extension: .o

Contains
- Assembled piece of code
- Constant data
- External references
- Debugging information

Typically, the compiler automatically calls the assembler

Use the -Wa switch if the source files are assembly only and use gcc



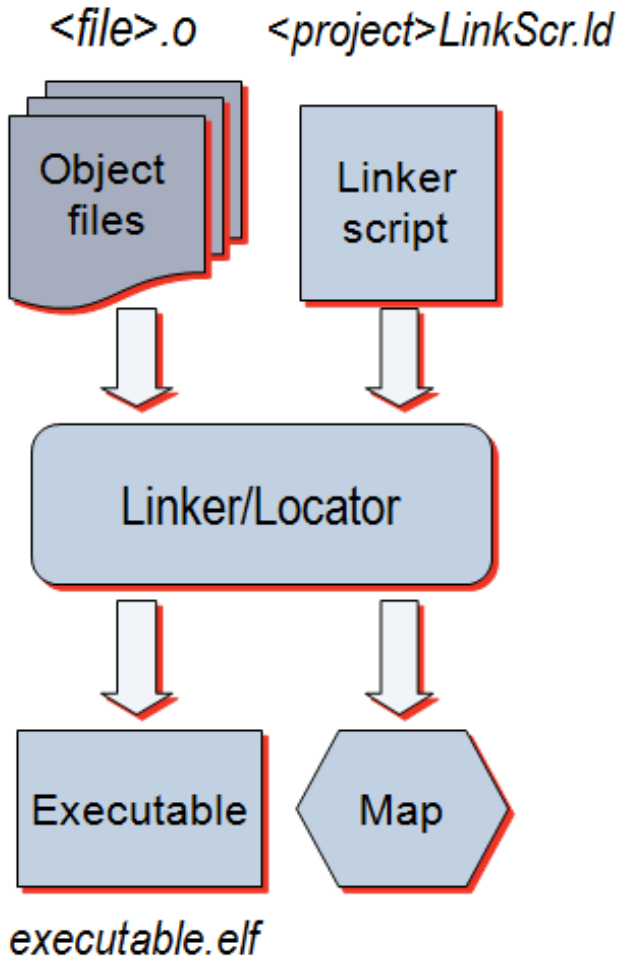
GNU Tools: Linker (LD)

Inputs

- Several object files
- Archived object files (library)
- Linker script (**.ld*)

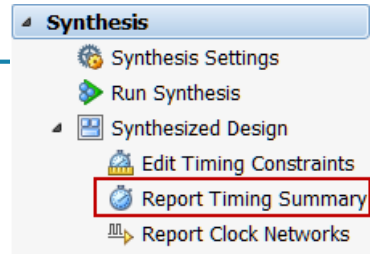
Outputs

- Executable image (ELF)
- Map file



Timing Reports

Report Timing Summary



Tcl command: report_timing_summary

report_timing_summary -delay_type max -report_unconstrained -check_timing_verbose -max_paths 10 -input_pins -name timing_1

Vivado IDE

Options tab

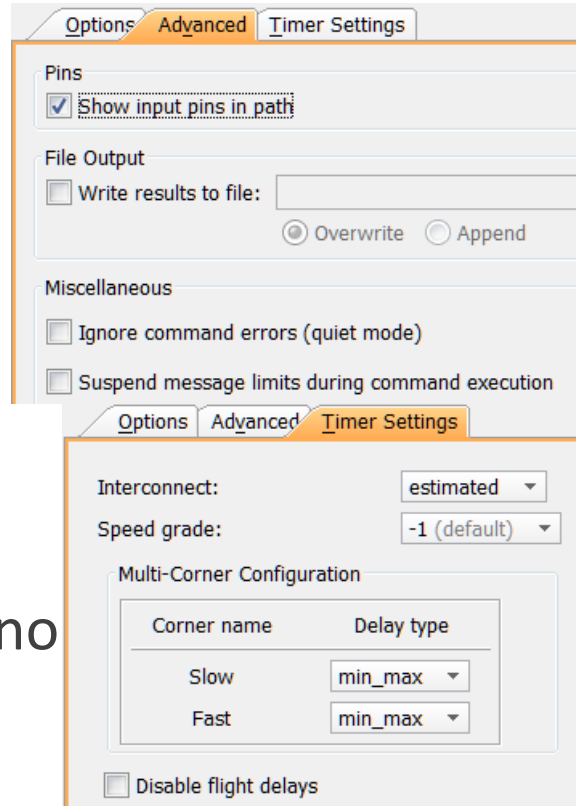
- Maximum number of paths

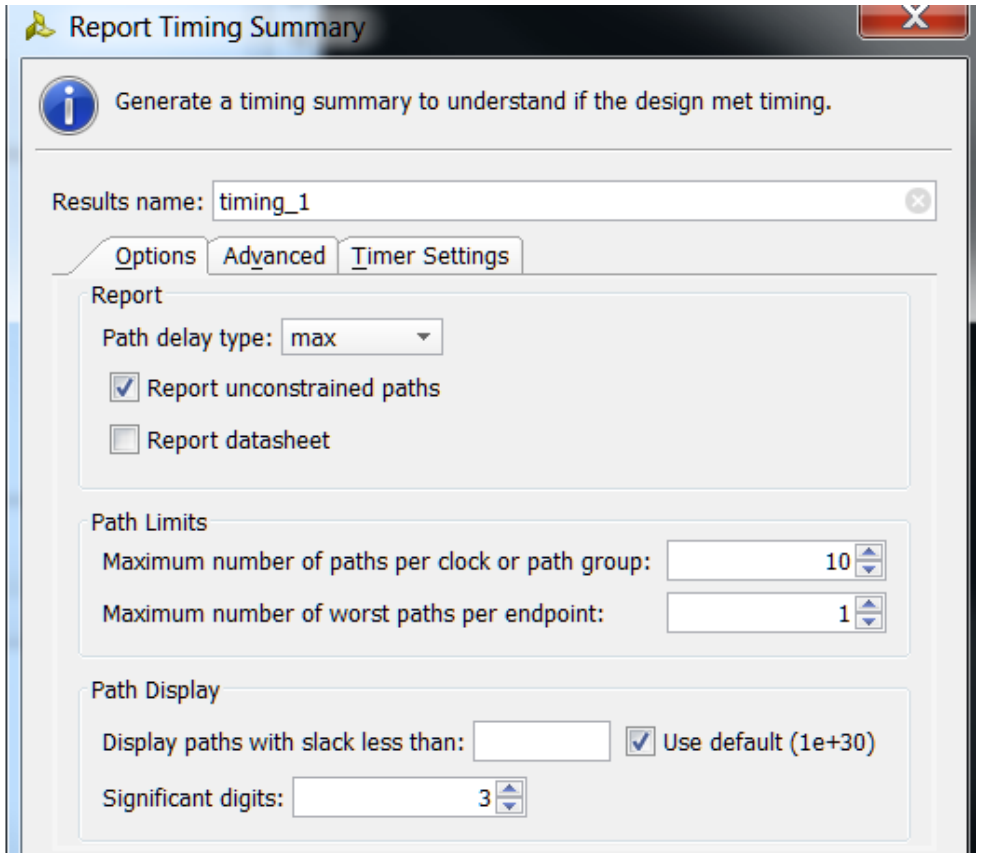
Advanced tab

- Write to a file

Timer Settings

- Interconnect delay can be igno
- Flight delays can be disabled





Report Timing Summary

Design Timing Summary

- WNS, TNS, total number of endpoints are of interest

Clock Summary

- Primary and derived clocks

Check Timing

- Number of unconstrained internal endpoints