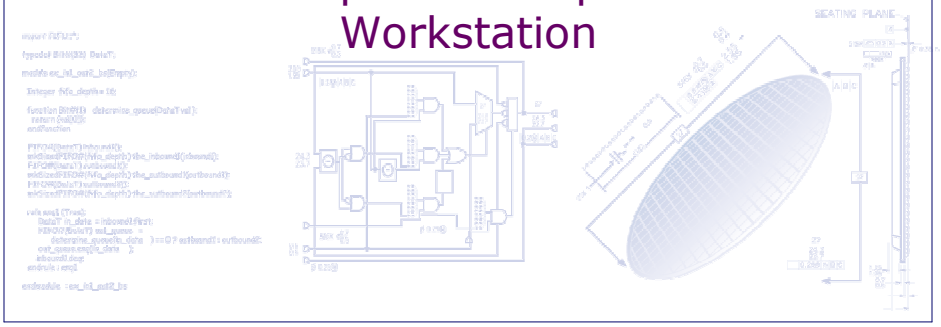
Bluespec SystemVerilog™ Training:

Using the Bluespec Development Workstation



Lecture Outline

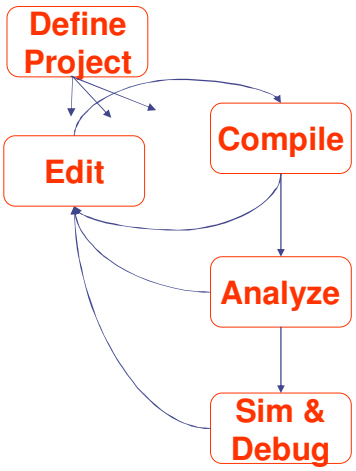
- Overview
- Use Models
- Details of browsers
- Underlying technology
- Demonstration

Overview

- Graphical environment for design and debug of Bluespec models
- Simplifies Bluespec ramp-up and edit-simulate-debug loop
- Integrated with 3rd party tools: editors, simulators, and waveform viewers
- Browsers allow multiple perspectives of the design
- Allows programmable access to compiler information for debug: Packages, Type, Scheduling, Elaborated Hierarchy

3

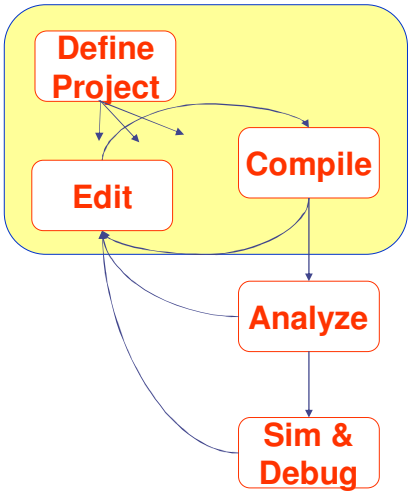
Development Flow



- Current tools
  - Editors
  - Simulator
  - Waveform viewers
  - Custom scripts
- Bluespec adds
  - Project definition
  - Package type browser
  - Schedule analyzer
  - Source-level debug

4

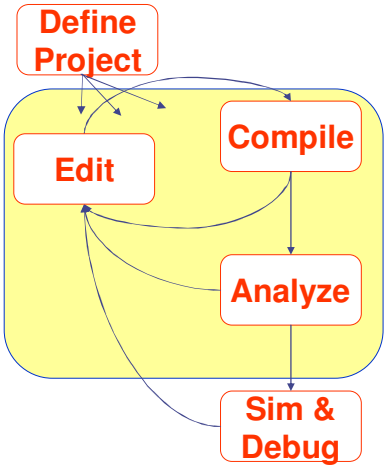
Edit-Compile Flow



- Setup project definition
  - Tools and options
  - Default configuration usually works
- Compile
  - One button to compile
  - hot-link from errors
- Package browser
  - Search for definitions
  - Verify argument types
  - Locate source in Bluespec Azure IP and user code
- Only need to learn & use Main window

5

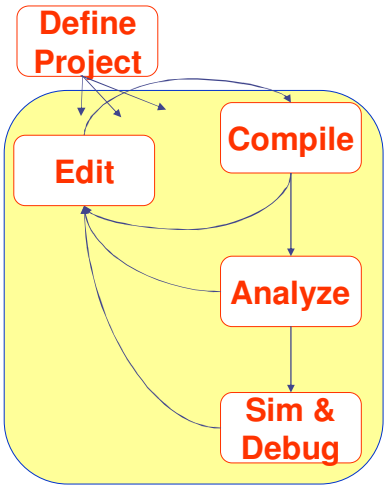
Analyze Edit loop Scheduler



- Various perspectives of a Bluespec model
- Scheduler analysis
  - Relations between rules and method calls
  - Rule relations
  - Graphical views of scheduler
- Improved understanding of scheduler operation
- Analysis tool extendable

6

Simulate-Debug-Edit



- Module browser
  - Elaborated hierarchy
  - Source code links
  - Rules and interfaces linked to waveform viewer
- Waveform viewer
  - Integrated type analysis with waveform viewer
  - Can view interface signals at BSV abstraction level
  - Signals from interface can be sent to wave with type annotation and structure

7

Project

A Bluespec project is a collection of settings and options related to one or more Bluespec modules (file: name.bspec)

- Top File/Top Module
- Search Path
- Build options
  - Compile, Link and Simulate
- Third party tools
  - Editor
  - Simulators
  - Waveform Viewer

Component Windows

- Main window
  - Status, commands, project options
- Project Files window
  - Edit, compile
- Package browser
  - Search, view package contents
- Type analysis
  - Expand, view types

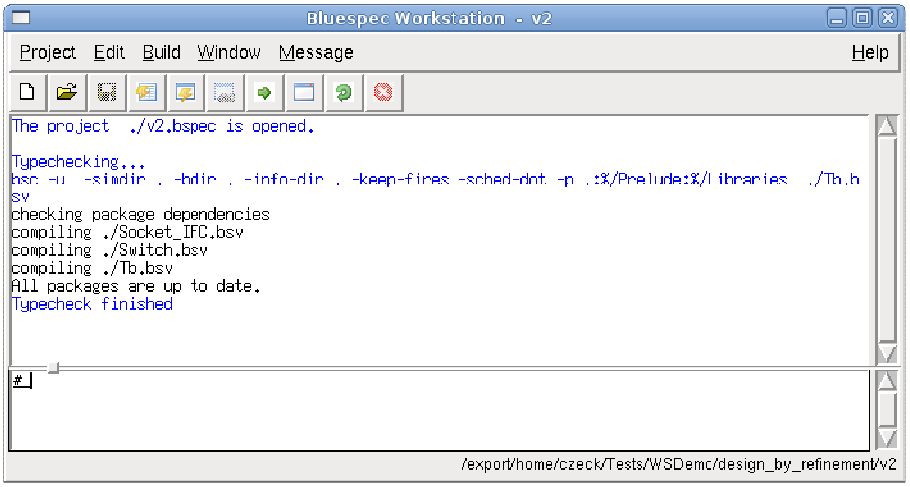
Pre-elaboration
.bo files needed

- Schedule analysis
  - Many views on scheduler information
- Module Hierarchy Browser
  - Elaborate hierarchy, wave integration

Post-elaboration
.ba files needed

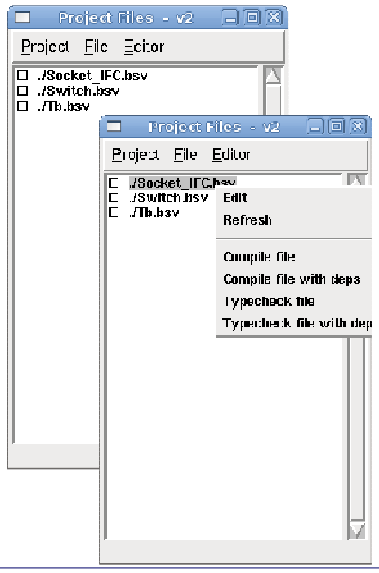
9

Main Window



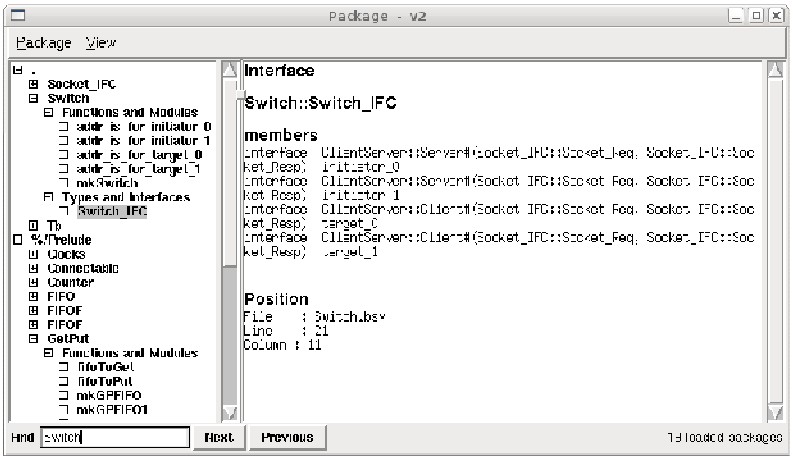
Project Files Window

- Shows all bsv files.
  - Customize with other files as needed.
- Display, edit, typecheck or compile files from this window



Package Window

- Displays contents of .bi/.bo files – including Bluespec library packages

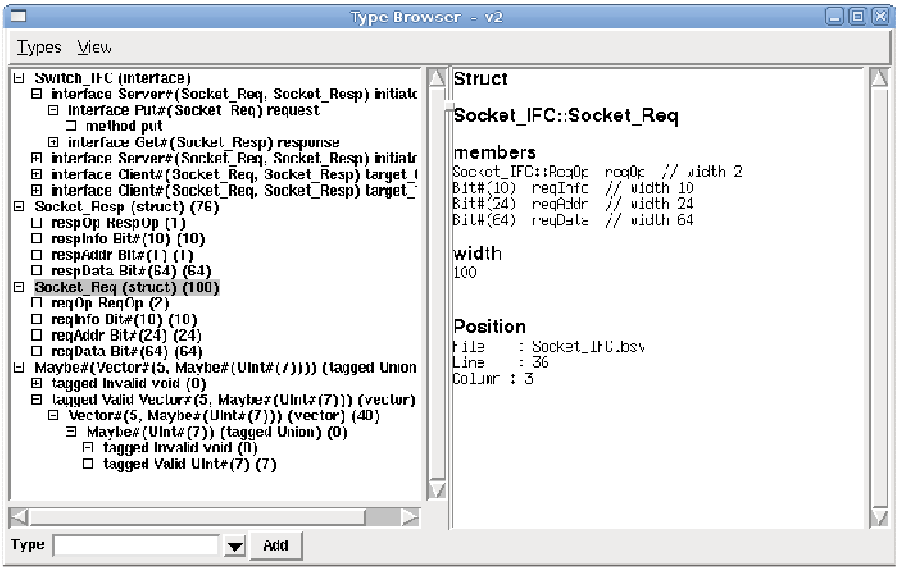


Type Browser

Detailed type information

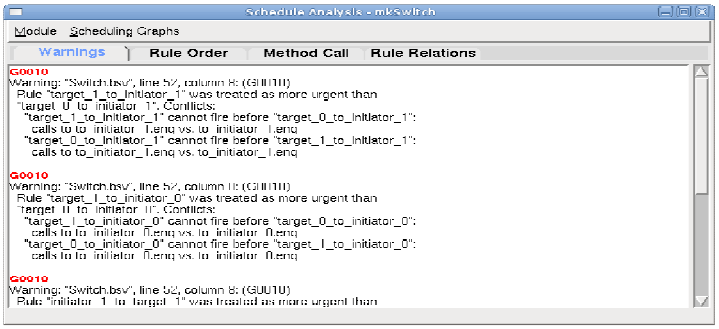
- Full type hierarchy
- Concrete types derived from resolution of polymorphic types
- Shows sizes of bit representation
- Interface methods and attributes defined on interface

Type Browser

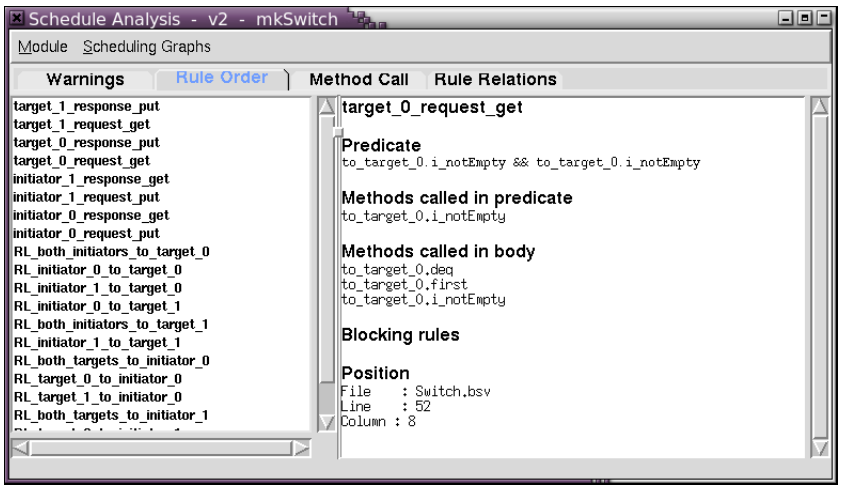


Schedule Analysis - Warnings

- Warnings generated by compiler about scheduling decisions

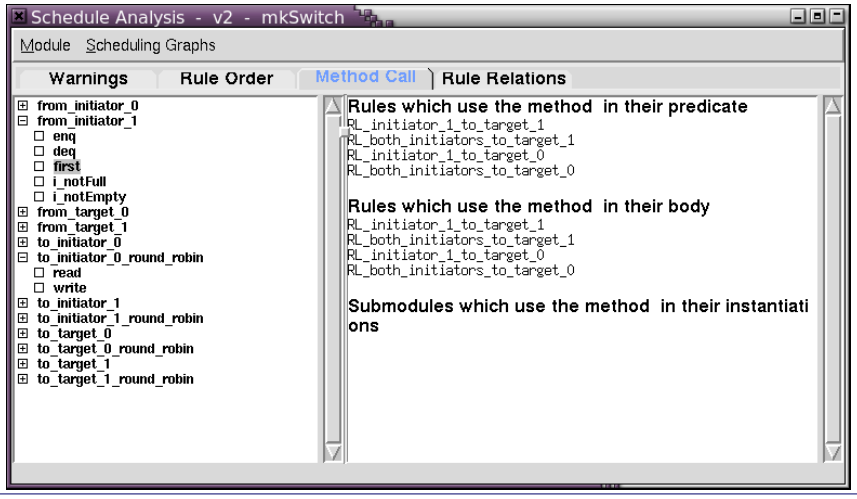


Schedule Analysis – Rule Order

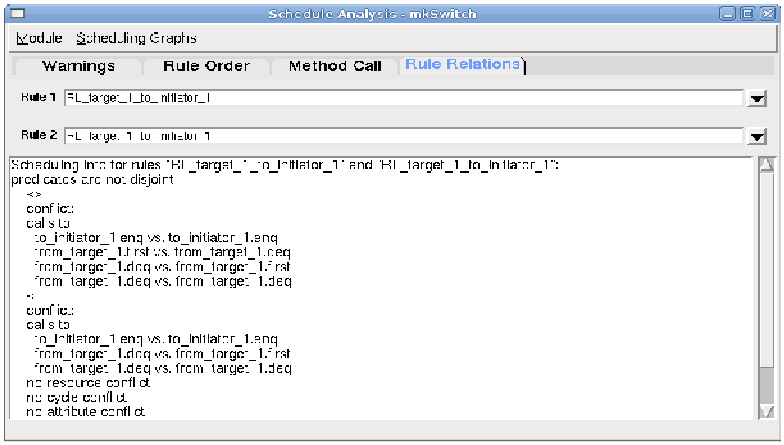


Schedule Analysis – Method Call

- Mirror of Rule order



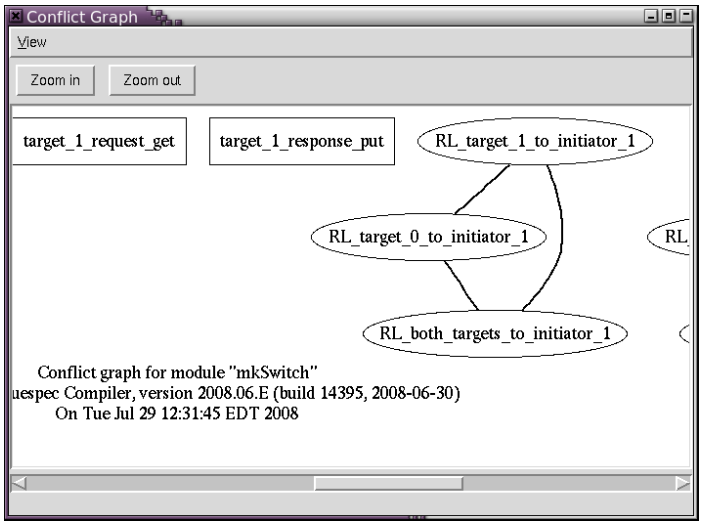
Schedule Analysis – Rule Relations



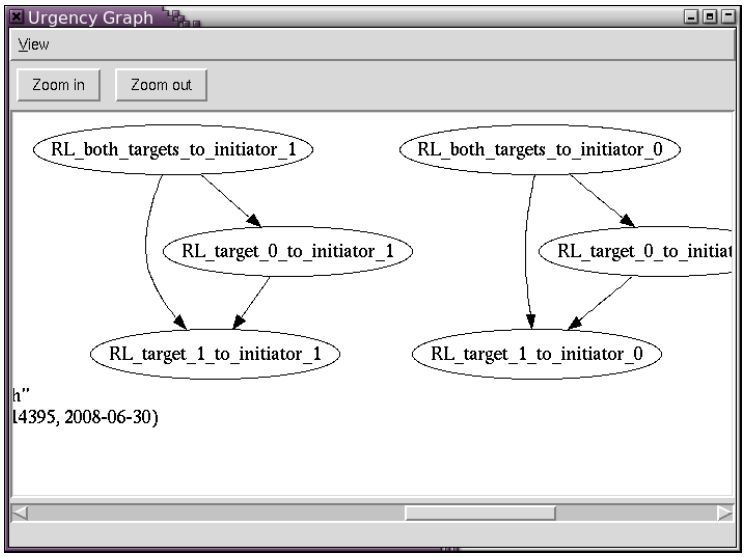
Scheduling Graphs

- Useful for visualizing relationships between rules/methods
- Requires:
  - Synthesized module
  - `-sched-dot` flag during compilation. (This flag generates the `.dot` files to populate the graphs)
  - Graphviz Tcl extensions (TclDot) compatible with Tcl 8.4
- See user guide for details of graph infomation

Conflict graph

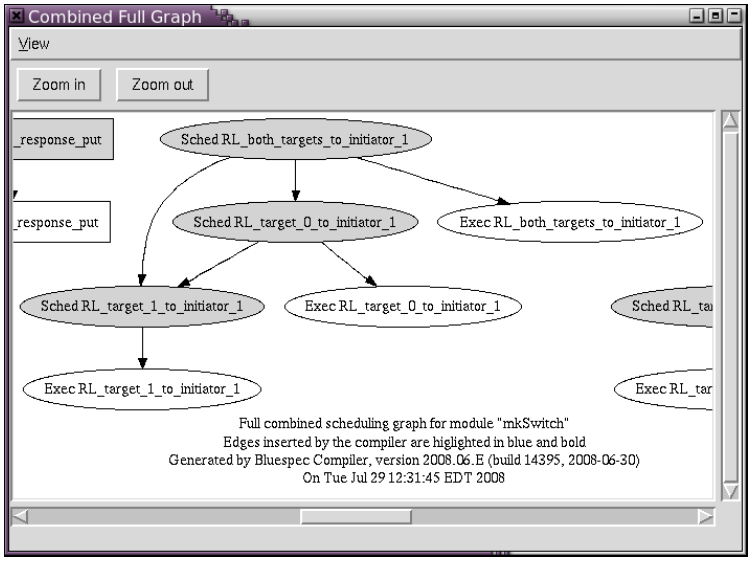


Urgency graph



21

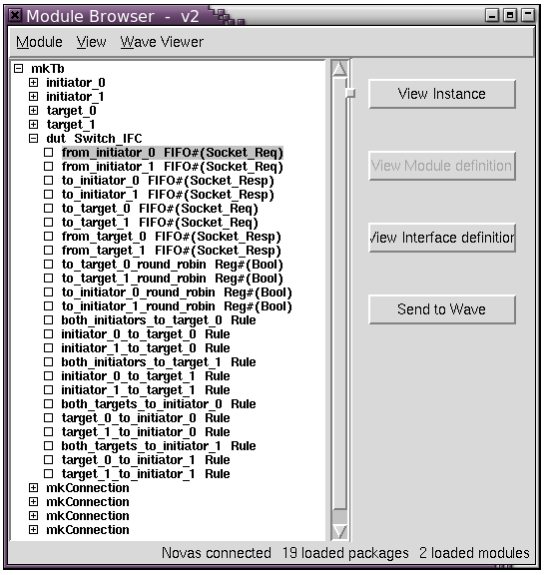
Execution Graph



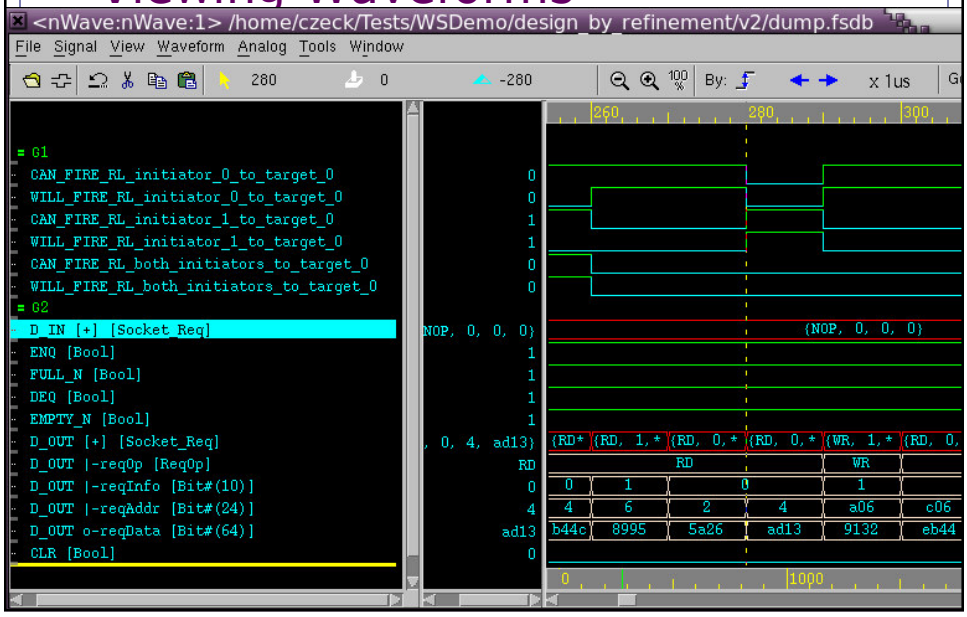
22

Module Browser

- Post-elaboration view of module hierarchy
- Links BSV interface & types to waveform viewer



Viewing Waveforms



bluespec

Keyboard and Mouse Shortcuts

- Double-click, Enter – Goto source, goto error
- Arrow up/down – obvious
- Arrow left/right – collapse/expand hierarchy tree
- All menus have key equivalents (Alt-<letter>)
- Button-2 pop-ups on lists

bluespec

Behind the scenes

- Workstation built with Tcl/Tk on top of a tcl interface to Bluespec code -- Bluetcl
- Bluetcl extends tcl with commands allowing access to information from .bi/.bo and .ba files
- Users can define their own programmable extensions, E.g.,
  - C language wrapper generated from interface type
  - Synthesizable instances from module definition and concrete type
  - Mapping of BSV type to its Verilog representation

bluespec

Examples of Bluetcl packages

See BSV user guide

- Bluesim package: tcl-api to control Bluesim simulator
- expandPort script: generates Verilog wrapper expanding BSV structures
- Types package: queries functions on type hierarchy
- InstSynth package: Scripts to automate synthesis boundary generation for polymorphic modules

27

bluespec

Resources

- Workstation user guide
- Bluetcl reference in bsv user guide
- Examples and tutorials include .bspec files (invoke with: `bluespec x.bspec`)
- Tcl references abound

28

bluespec

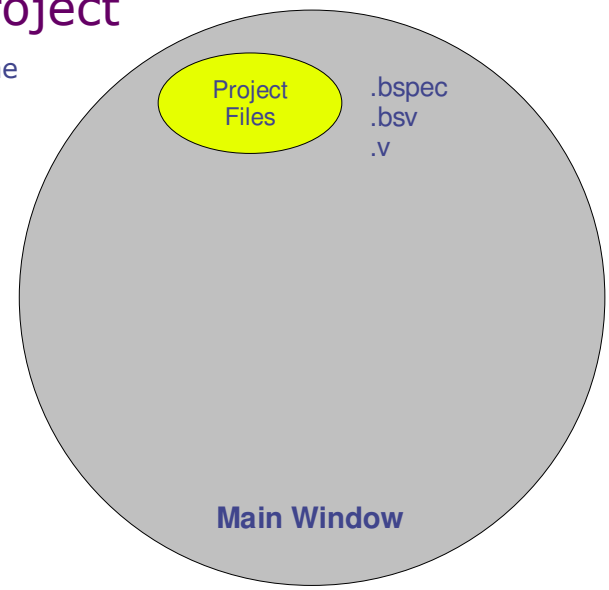
ESL SYNTHESIS

End of Lecture

bluespec

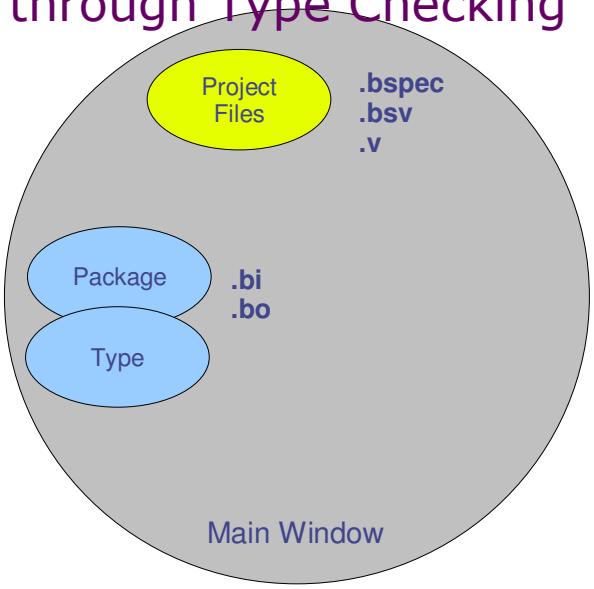
Open Project

- Main window is the control center of workstation
- Project files window used to view, edit and compile files



Compile through Type Checking

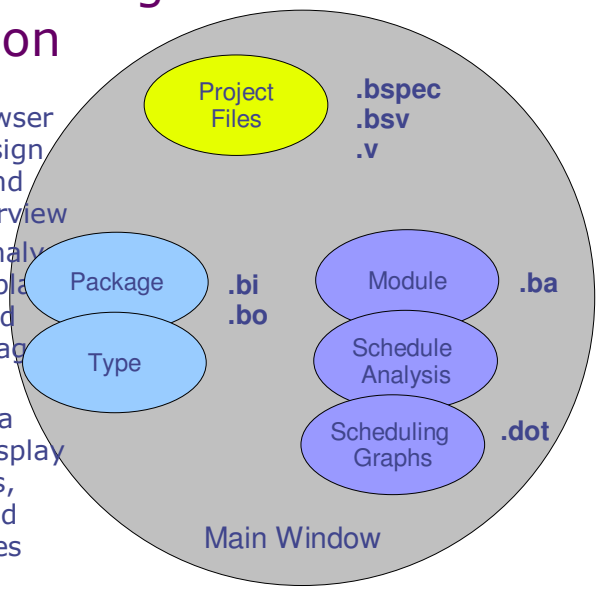
- Package Window is the high level view of package contents
- Type Browser is primary means for viewing details about types and interfaces



31

Compile through Code Generation

- Module Browser displays design hierarchy and module overview
- Schedule Analy Window displa schedule and resource usag
- Scheduling Graphs are a graphical display of schedules, conflicts, and dependencies



32

Schedule Analysis – Rule Relations

- Same as generated from -show-rul-rel flag
- Types of conflicts:
  - <> : conflict
  - < : first rule cannot be executed after the second rule
  - Resource: more rules vying for a method than available ports
  - Cycle: conflict introduced to break execution order cycle
  - Attribute: conflict introduced by scheduling attribute, such as the preempts attribute.