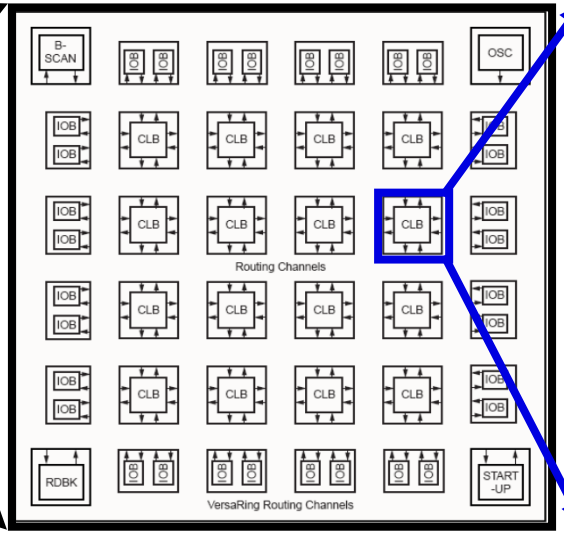
Introduction to FPGA Testing

Examples of FPGAs



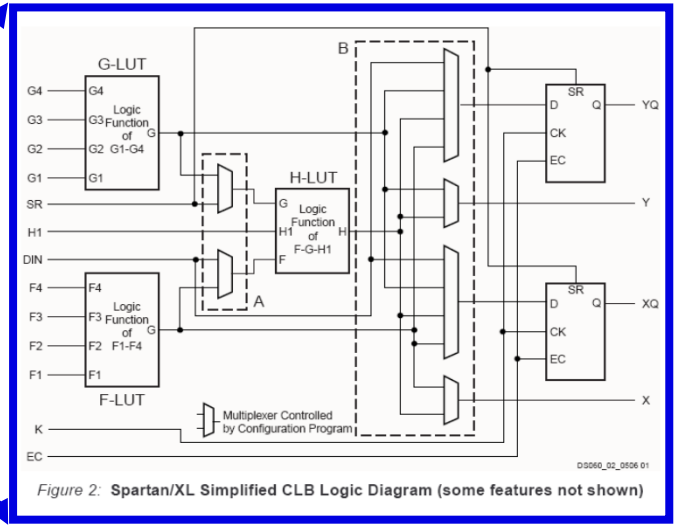
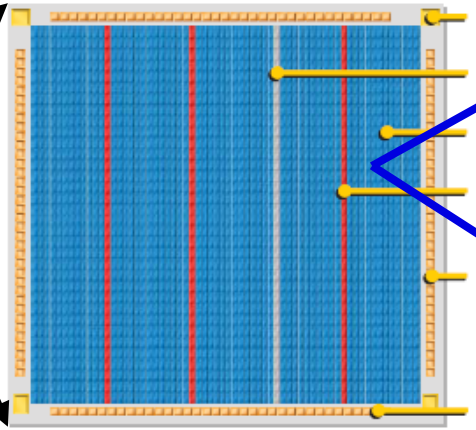
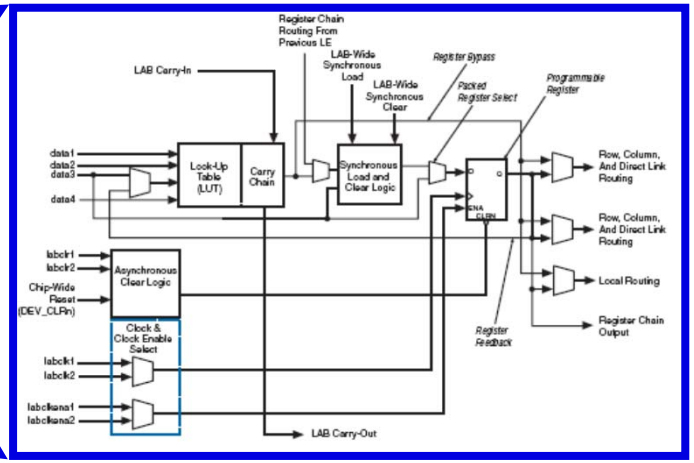


Figure 2: **Spartan/XL Simplified CLB Logic Diagram (some features not shown)**

Cyclone EP1C20 FPGA
20,060 logic elements





Potential Problems

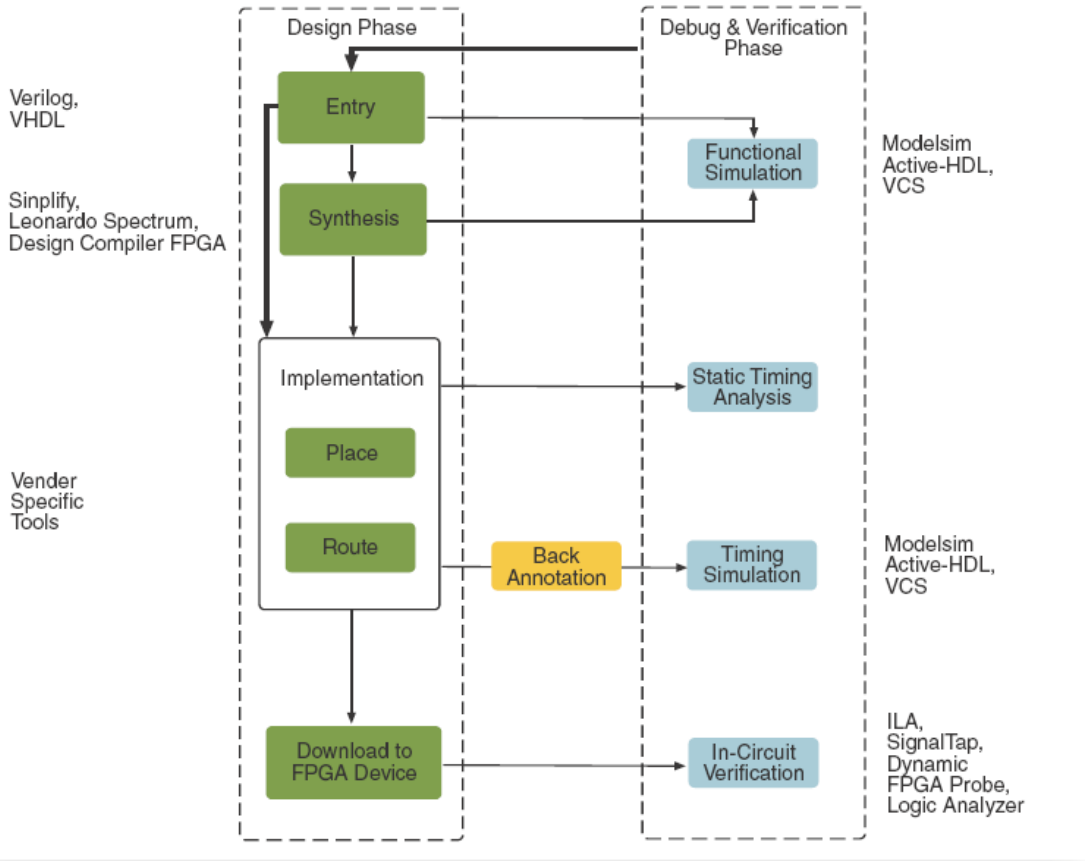
- Functional definition errors
  - At the FPGA or system level
- Functional system interaction problems
- System-level timing issues
  - Asynchronous events
  - Real-world interactions, especially at speed
  - Difficult to simulate timing violations
- Signal fidelity between ICs
  - Noise, cross-talk, reflections, loading, EMI
- Interconnect reliability issues
  - Solder joints, connectors
- Power supply issues
  - Transients and load variations
  - High power dissipation
- Undiscovered FPGA design errors due to incomplete simulation
  - Too complex to provide 100% code coverage
  - Too time-consuming to implement and run

FPGA Debug Challenges

- Design verification has become a critical bottleneck
  - Increased design size and complexity
  - Limited access to internal signals
- Time-to-market constraints reduce debugging time
  - Debugging can take >50% of design cycle time
- Simply looking at the external pins is inadequate
- Adding debug circuitry to FPGA affects the design
  - Consumes valuable chip real estate
  - Requires additional time
  - May affect the design's timing performance
  - Access usually uses up scarce pins on the chip
  - Probing many signals on the board may be difficult

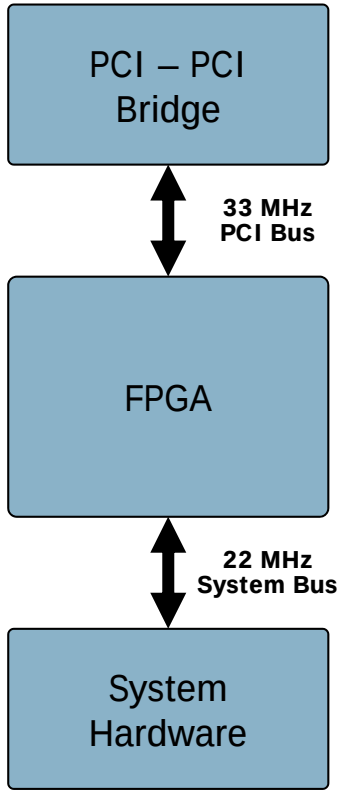
Debugging FPGA Designs

- Challenge
  - How to get visibility into FPGAs that are getting bigger and more complex?
- Review 2 Basic Methodologies
  - advantages and disadvantages of each
- Conclusions
  - Is there a “best” methodology?



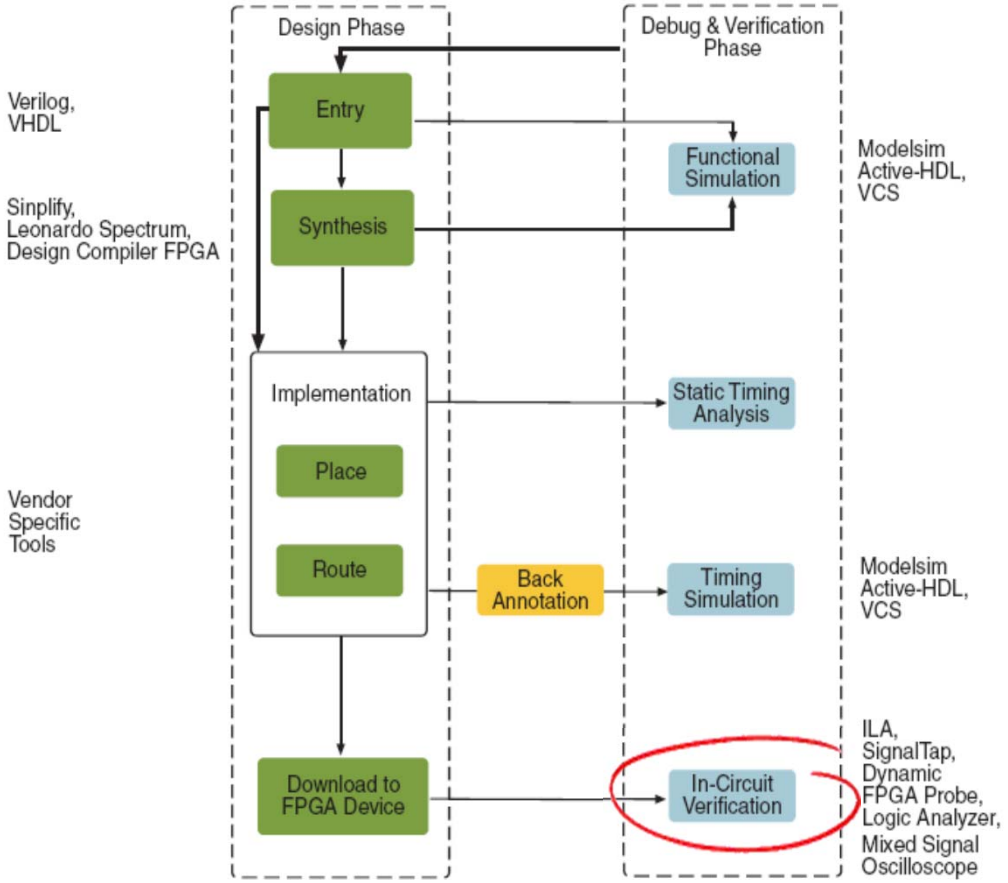
Simulation

- Simulation can help reduce debug time by catching obvious errors, but... simulation can not catch all problems!
  - Can not simulate asynchronous events
  - Hard to simulate real-world interactions, especially at speed
  - Difficult to simulate timing violations
- 100% Code Coverage is Difficult
- Simulation runs are slow



FPGA Design Process

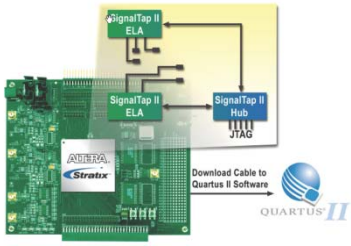
- Design Phase Tasks
  - Design Entry
  - Design Implementation
  - Simulation
- Debug and Verification Phase
  - Validate Design
  - Correct any Bugs found
- Debug and Verification Methods
  - Simulation
  - In-Circuit Verification



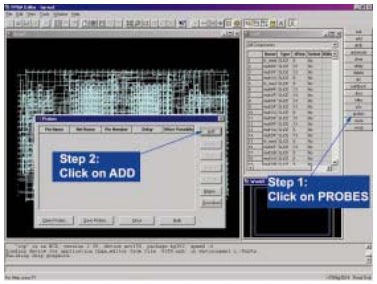
Overview of In-Circuit FPGA Debug Methods

Basic Approaches

- Embedded Instruments
  - Embed a logic analyzer/bus analyzer into your design
    - Logic analyzer functionality is inserted in design
      - SignalTap® II (Altera)
      - ChipScope™ ILA (Xilinx)



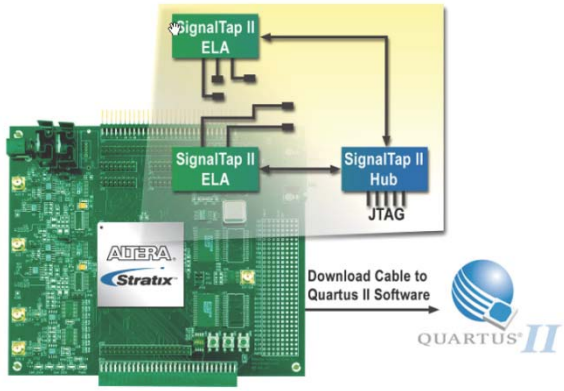
- External Instrument
  - 1:1 Signal/Pin
    - Modify RTL as needed
      - Take advantage of the programmability of the FPGA to route internal signals to a small number of pins
    - Use FGPA tools to add “Probes” to your design
      - Adds a “green wire” to your FPGA
        - » SignalProbe (Altera)
        - » PROBE (Xilinx)
  - N:1 Signal/Pin
    - Insert a test mux
      - Allows more visibility than Modify RTL method
    - Instrument-controlled test mux
      - Creates instrument awareness and control of inserted test mux



Embedded Logic Analyzer

SignalTap II / ChipScope ILA / CLAM

- FPGA vendors offer logic analyzer cores
- Inserted in design and contain triggering and trace storage resources
  - Uses FPGA resources
- Core is accessed via JTAG
- Data displayed in FPGA vendors viewer application



Advantages

- Fewer pins required. Uses JTAG pins
- Simple probing
  - Just hook up JTAG cable
- Embedded logic analyzer cores are relatively inexpensive

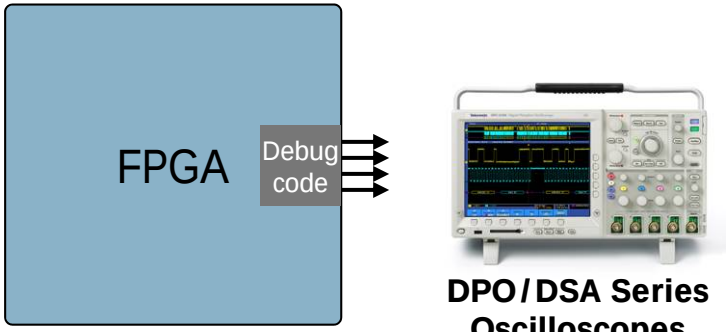
Disadvantages

- Size of core limits use to large FPGAs
- Designers have to give up internal memory for trace storage
  - Limited memory depth
- Data can be captured in state mode only and at a limited speed
  - Limited to fastest clock in FPGA
- Can't correlate (see) FPGA trace data with other system traces

External Test Equipment

Oscilloscopes

- Inserts custom debug code in design
  - Leverages the programmability of FPGA
  - Can route copies of internal signals of interest to output pins for viewing
- Can implement complex triggering internally
- User interface is familiar and frequently used



DPO/DSA Series Oscilloscopes

Advantages

- Lowest (capital) cost technique
- May use few FPGA logic resources
- Uses no FPGA memory
- Can correlate FPGA signals to other analog or digital signals system

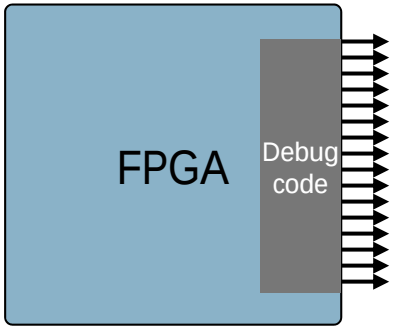
Disadvantages

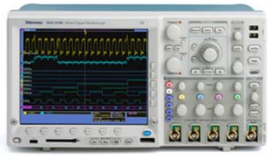
- Debug code must be redesigned and recompiled for each experiment.
- Consumes valuable FPGA gates and pins
- Very limited visibility of complex designs
  - Limited by pins and scope channels
- Inefficient technique for complex designs

External Test Equipment

Mixed-Signal Oscilloscopes

- Inserts custom debug code in design
  - Leverages the programmability of FPGA
  - Can route copies of internal signals of interest to output pins for viewing
- Can implement complex triggering internally
  - MSO may reduce this need





MSO Series Mixed-Signal Oscilloscopes

Advantages

- MSOs offer more channels and wider logic triggering than oscilloscopes
- MSOs offer parallel bus and event table displays of the digital signals
- May use few FPGA logic resources
- Uses no FPGA memory
- Can correlate FPGA signals to other analog or digital signals system

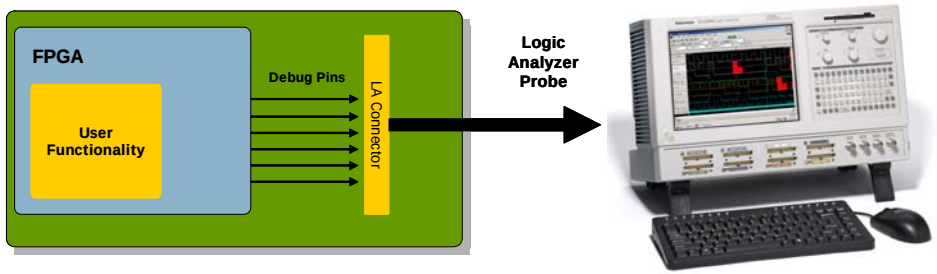
Disadvantages

- Debug code must be redesigned and recompiled for each experiment.
- Consumes valuable FPGA gates and pins
- Visibility of complex designs somewhat limited by pins
- Have to manually update signal names and channel assignments on MSO

External Logic Analyzer

Modify RTL source as needed

- Take advantage of the programmability of the FPGA to route internal signals to a small number of pins
- 1:1 Relationship of internal signals to FPGA pins
- Use external logic analyzer to capture data



Advantages

- Use few, if any, FPGA logic resources
- Uses no FPGA memory
- Data can be captured in state mode and in timing mode
- Can correlate (see) FPGA signals to other system signals

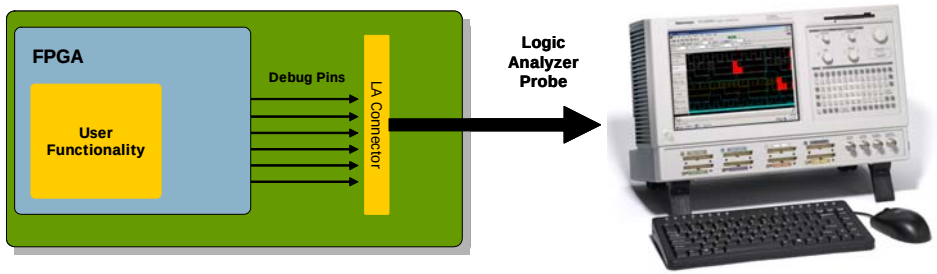
Disadvantages

- Requires more pins on FPGA
- Moving probe points can take a recompile of the design
  - Uses time and changes timing of design
- Have to manually update signal names on logic analyzer

Use FGPA tools to add “Probes” to your design

Green Wire

- FPGA vendors offer ability to add “green” wire to design
  - Does not require modification to RTL
  - 1:1 Relationship of internal signals to FPGA pins
- Use external logic analyzer to capture data



Advantages

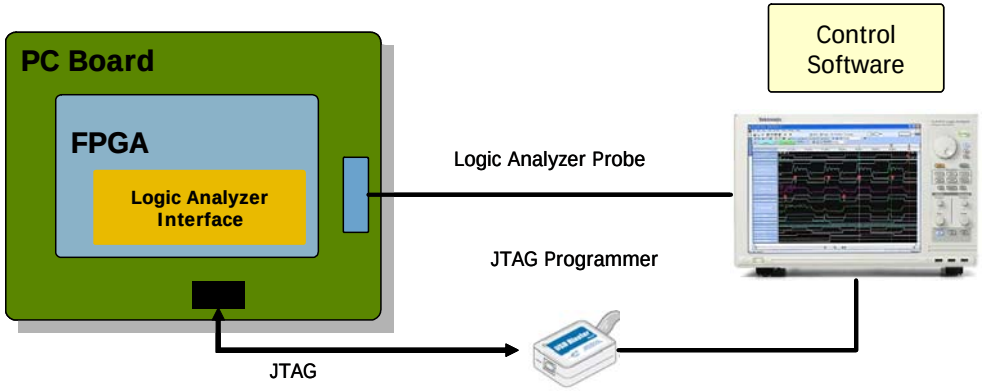
- No modification of RTL required
  - No need to recompile design
- Uses no FPGA memory
- Data can be captured in state mode and in timing mode
- Can correlate FPGA signals to other system signals
- Reduces need to recompile design

Disadvantages

- Requires more pins on FPGA
- Requires a test mux
- Requires a way to control test mux
- Still have to manually update signal names on logic analyzer

Instrument-controlled test mux

- Closed-loop system between FPGA and test equipment
  - Test equipment is aware of the design
- Increased visibility of internal signals
  - N:1 relationship of internal signals to pins
- Use external logic analyzer to capture data



Advantages

- Uses no FPGA memory
- Test Mux designed and tested by 3rd party
- Data can be captured in state mode and in timing mode
- Can correlate FPGA signals to other system signals
- Eliminates recompiles

Disadvantages

- Requires more pins on FPGA
- External instrument vendor specific

Picking the Right FPGA Debug Methodology

Feature	Embedded Logic Analyzer	"Probe"	Modify RTL	Insert Test Mux	Instrument-controlled Test Mux
Sample Depth	**Fair**	**Best**	**Best**	**Best**	**Best**
Instrument Performance	**Fair**	**Best**	**Best**	**Best**	**Best**
Correlation of FPGA Signals to board signals	**None**	**Excellent**	**Excellent**	**Excellent**	**Excellent**
Uses FPGA Memory	**Design-dependent**	**None**	**None**	**None**	**No**
Use of FPGA Resources	**Logic & Routing**	**Routing**	**Routing**	**Logic + Routing**	**Logic + Routing**
Output Pin Usage	**JTAG**	**1-256**	**1-256**	**1-256**	**1-256**
Cost	**ELA License**	**External Instrument**	**External Instrument + Time**	**External Instrument**	**External Instrument+ Control SW**

FPGA Debug

Summary

- Is there a best methodology for all designs?

  Almost Impossible !

- Choosing the right FPGA debug methodology for your design can reduce debug and validation time.
  - Make the choice early – in the design phase.
    - Are you pin-constrained or FPGA-resource constrained?
  - Choice of methodology impacts the design of the board
    - How to connect to external instrument?
    - FPGA Pin useage

- Embedded logic analyzers and external logic analyzers each have trade-offs.
  - Pins vs. Resources
  - Ability to correlate FPGA signals to board-level signals
  - Power of instrument
    - Memory, triggering, resolution

Summary

- Reduce Debug and Validation Time
  - Choosing the right FPGA debug methodology can reduce debug and validation time
- Understand the Trade-offs
  - Embedded logic analyzers and external test equipment each have their own advantages and disadvantages