FPGA-Centric Functional Verification

Mark Litterick
Senior Consultant, Verilab Ltd.

22-May-2002
Presentation Outline

• Complex FPGA verification problem
• Propose a practical solution based on modified ASIC methodology
• Key verification aspects of FPGAs
• Overview of methodology
• Conclusion
FPGA Verification Problem

• Traditional ad-hoc FPGA verification is inadequate for modern complex FPGAs
  – Longer design cycle
  – Increased time-to-market

• Major issues include:
  – Debug
  – Fixes introduce new bugs
  – Back-end build-and-test loop too slow
  – System & S/W correspondingly complex
Adapt ASIC Methodology

- Similar complexity problems
- Capitalise on unique FPGA features
- Invest in proper simulation testbenches
- Appropriate back-end processes
FPGAs are Special

- Biggest advantage: Re-programmable
  - Fix bugs
  - Phased product releases
  - Prototype ASICs
  - Evolve with specifications
  - Field upgrades

- Biggest disadvantage: Re-programmable
  - Relied on to fix bugs
  - Promotes trial-and-error engineering
FPGA - Similarities to ASIC

• Require to **verify** every feature
• Regression is required
• Independent verification => better quality
FPGA - Differences to ASIC

• Do not have to **simulate** every feature
• Ability to system test features that are:
  – Slow to simulate
  – Difficult to emulate
• Regression simulations do not have to cover all features
• System regression tests augment regression simulations
Overview of FPGA Methodology

• What is it?
• Who needs it?
• What are the benefits?
What is FPGA Verification?

• Methodology
• Ensure FPGAs in a system have required functionality
• Functionality includes both correctness and performance
• Verification strategy includes:
  – Simulation
  – System Test
  – Regression
Who Needs FPGA Verification

- High-complexity applications
- Large gate-count applications
- High-quality applications
- Expensive field upgrade applications
Benefits of a Good Methodology

- Minimize cycle time
- Maximize quality
- Certify conformance
- Debug environment
- Repeatability through regression
- Maximize ROI through reuse
Overview of Methodology
Overview of Methodology
Design For Verification

• What is DFV?
  – Addition of operational mode to allow other features to be simulated faster or easier

• How does it compare to DFT?
  – DFT is for manufacturing test vectors
  – DFT is about improving test coverage or shortening test time

• More important in FPGA than ASIC
  – Faster simulations using DFV modes
  – Only system tests in full-functional mode
Design For Verification

• Implementation:
  – minimal point of application
  – real features must operate normally
  – in actual source, not a separate design

• Examples:
  – short-frame modes
  – special line-standards
  – reduced FIFO depth
  – large counter control/load
DFV : SDH Short-Frame Mode

STM-1 Frame

STM-1 Short-Frame

DFV

<table>
<thead>
<tr>
<th>Frame Length</th>
<th>Short-Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM-1</td>
<td>270</td>
</tr>
<tr>
<td>STM-4</td>
<td>1080</td>
</tr>
<tr>
<td>STM-16</td>
<td>4320</td>
</tr>
<tr>
<td>STM-64</td>
<td>17280</td>
</tr>
</tbody>
</table>
Verification Plan

- Identifies set of essential functional features
- Identifies where verified
- Used to specify verification effort
- Used to plan verification effort
  - Prioritise
- Specifies criteria for verification completion
- Enables functional coverage analysis
  - Manage split between sim and test
- Just as crucial to FPGA as ASIC
Overview of Testbench Design

• More important for FPGA than ASIC
  – ROI through reuse
  – Not justified in one ASIC/FPGA

• Language
  – HLVL or HDL

• Structure
  – Design for reuse
  – Package for maintenance
  – Layer for flexibility
  – Abstract interfaces
Design for Reuse

- Generic Verification Components
- Fix interfaces early - evolve functionality
- Encapsulate
- Package related functionality separately
- Package user modifiable stuff separately
Package for Maintenance

- If a testbench is difficult to maintain it will not be maintained
- Documentation is crucial
- Modularise by packaging
- Improving maintainability also improves reuse
Layer for Flexibility
Abstract Interfaces

• User Interface
  – Hide complexity
  – Only testbench designers need to be expert

• Command Interface
  – Correct abstraction is crucial
  – Operation level, not low-level BFPs

• Procedural Interface
  – Encapsulate data structures
  – Hide implementation detail
  – Isolates modifications, e.g. change data format
System Test

• System test used instead of simulation:
  – Features that are slow to simulate
  – Features that are difficult to emulate
  – When the project pressure is on…

• Revert to simulation for debug
Simulation Debug Environment

• Ultimate logic analyser
• Testbench capable of simulation all features
  – Target features for system test
  – Use testbench to debug when they don’t work!
• Single biggest improvement to minimising FPGA and product design cycle
Back-End Verification

- Limited or no gate-level simulation
  - Massive saving on simulation effort
- Static timing analysis
  - Put effort into timing constraints
  - Do not over-constrain
  - Do not apply multiple guard-bands
- Formal Verification
  - Equivalence checking
  - Appropriate for transformation checking
  - It’s the synth/build tool’s problem
Regression Environment

• Regression Simulations
  – Maximise feature set covered by simulations
  – Limited to FPGA scope
  – Fast debug

• System Regression Tests
  – Reserve for specific features
  – Long debug cycle when a test fails
  – Failure could be S/W, H/W or FPGA

• Verification Plan is key to managing regression responsibilities
Conclusion

• A recipe for successful functional verification of complex FPGAs:
  – Take the best from ASIC
  – Modify for the benefits of FPGA
  – Invest in quality testbench design
  – Focus on design-for-verification
  – Use regression in simulation and system test
  – Pull it all together with a verification plan
  – Reuse it all in your next project!
Contact

Mark Litterick
Senior Consultant
Verilab Ltd.
Willow House
Strathclyde Business Park
Bellshill
Scotland
ML4 3PB

Phone: 01698 464500
Mobile: 07810 822206
E-mail: mark@verilab.com

www.verilab.com

22-May-2002