Verification Successes

• 3rd Generation 32bit floating point SHARC DSP family

• Multi-Pronged Verification

• Formal verification on critical blocks like External Mem Intf/External Bridge

• Many Deep state bugs caught using FV

• Closure metric well defined, tracked and executed

• First-Pass Silicon
Verification Failures

Top 5 reasons for functional failures:

- Incomplete Verification Plans for blocks due to schedule pressures
- Lack of formal verification on critical blocks resulting in missed deep state bugs. Eg: CORE-DMA conflict resolution error under some very corner case scenario
- Lack of clarity in block specs a deterrence for using formal verification
- Not enough functional coverage cases defined resulting in uncovered cases
- Inaccurate Verilog modeling for some cells resulting in erroneous operations
How to get positive ROI from FV tools

• Choose a control-dominated block. FV on Jump control Unit caught lots of bugs while on Data Mapping Unit, no bugs were found

• End-to-End properties gives best ROI

• Making designers write in-line assertions during the code development

• Cut down on time wasted during stabilizing input constraints
  • Partitioning at well defined interfaces
  • Active designer involvement in describing input constraints

• Cut down on COI by reducing property. Rather than checking priority logic for 64 interrupts, check for 16.

• Use of assistant coverage goals to improve convergence for semi-formal tools
Other Applications of FV

• Generating test-cases for interesting scenarios in design
  • In Instruction Fetch Unit, generate test-case where back to back instr are predicted jumps and stall is asserted.

• Bug Reconstruction and Bug Fix validation during post-silicon debug

• Performance validation for worst-case throughput/latency related issues
  • \texttt{intrpt\_generated} |-> \texttt{#}[1:20] \texttt{intrpt\_serviced};

• Equivalence checking between two blocks of with same function but different implementation

• Validating that re-timing didn't introduce bugs