IEEE P1500

Core Provider's Test Experience

Bruce Mathewson
ARM Ltd
Topics

- Introduction to ARM
- Testing an ARM core
- Questions
ARM Processor Family

- ARM7TDMI
  - 32-bit RISC CPU core
  - Thumb instruction set extension
  - EmbeddedICE Debug and a DSP enhanced multiplier
- ARM740T, ARM710T and ARM720T
  - Cached variants based on the ARM7TDMI core
- ARM9TDMI
  - 5-stage pipeline and Harvard buses
  - Provides more than twice the performance of the ARM7TDMI
- ARM940T and ARM910T
  - Cached variants based on the ARM9TDMI core
- StrongARM
  - Very high performance cached processor
CPUs need support

Apps Software and O/S

Support and training

Development Boards

Consulting

S/W Tools

Embedded CPU

EDA Support
ARM Partnership
How an ARM core is used

- Customer works with ARM semiconductor partner
- ‘C’ model of core used during RTL development
  - ARM has tools to wrap a C model for a variety of VHDL/Verilog simulators
  - ‘C’ model hides the IP of the core
- During layout ARM hard macrocell incorporated into netlist of the ASIC
  - Semiconductor partner has GDSII for appropriate process technology
- ARM provides three test approaches.
  - The system integrator must combine this with the test approach used for other modules
Problems with testing a processor core

- Hard Virtual Components use custom structures
  - Provide better performance/power consumption
  - Not suited to automatic test insertion
- Must be simple to integrate into large ASIC
  - Top level muxing causes integration problems, mainly due to routing and timing.
  - Scan insertion can cause problems at critical point in design flow
- IP Protection is required
  - Scan vectors can simplify reverse engineering
Testing an ARM Core

- **Parallel test**
  - Extracted from a functional simulation
  - Simulation uses a combination of code and test benches
  - Processors generally require very simple testbenches

- **Serial test (boundary scan)**
  - Parallel vectors converted into serial format
  - 105 elements on the scan chain

- **TIC Test**
  - Bus based approach to test
  - Parallel vectors converted to TIC format
AMBA Test Methodology

- Each peripheral should be testable using bus accesses
- This is achieved using a test wrapper for the peripheral to allow
  - Inputs to be stimulated
  - Outputs to be observed
- Bus accesses can then be generated by any element in the system - such as the TIC
Test Interface Controller

3 control signals

TREQA  TREQB  TACK

Memory Control  Memory Address  Memory Data

Re-use these pins to apply the 32-bit test vectors

External Bus Interface becomes bus master and generates transfers on the bus
Testing a Processor using TIC

- 32-bit test vectors
- Simple state machine controls the application of vectors
- 4 vectors required each time the processor is clocked
- Processor test wrapper requires
  - Register to store control
  - Mux to view outputs
Other advantages of TIC

- Unified test vector approach, with two file formats:
  - TIC - Very ‘C’ like
    - Easy to generate and easy to understand
    - Variables, loops, function calls can all be used
  - TIF - ASCII text file
    - Generated directly from TIC
    - Easy to read into simulation environment

- Vectors generated during module development
  - Standard ‘bus-based’ vector format used with standard testbenches, i.e. bus slave testbench

- Subset is used for production test
- Even smaller subset can be used for at-speed test
Summary

- ARM has a ‘standard’ approach to test
  - As do many others!
- Standards for test must be highly portable
  - Must work with different design constraints
  - Must work across different design flows
- Test methodology should not lie on the critical path of design flow
- Need ways to integrate different test methodologies
Thank you

Any Questions?