Verification of Lexra Processor Cores

Elliot Mednick
Charley Lind
Franklin Hooker
Lexra, Inc.

2000 IP World Forum

ABSTRACT

This paper describes the verification strategy used for microprocessors delivered in the form of silicon intellectual property (IP). The well-known challenges of verifying a complex microprocessor are compounded in the case of IP processor cores, which have large numbers of user configuration options and must be enhanced continually. To successfully verify processor cores, IP developers must have a high capacity, multi-tiered, and configurable verification capability. This includes in-system testing using real application code and on-chip debug, capabilities that are ideally made available with the processor IP, since users typically embed these cores deep inside a complex SOC.

Fortunately, technology is advancing on multiple fronts to meet the verification challenge. Lexra is incorporating many of these advancements into its verification strategy. Module verification at the RTL level, the first tier of verification, is benefiting from code analysis tools, accurate reference models, pseudo-random tests, and improved price-performance in simulation. Hardware-software co-verification models enable the next tier by providing the simulation bandwidth necessary to run useful amounts of application code in a simulation environment. The latest PLD generation provides the capability for hardware prototyping and software development in a generic development board before committing an SOC to silicon. Finally, test chip silicon using Lexra's optimized SmoothCore enables prototypes requiring real-time performance.

AUTHORS/SPEAKERS

Elliot Mednick
Processor architect and designer, Lexra, Inc.

Charley Lind
Design verification engineer, Lexra, Inc.

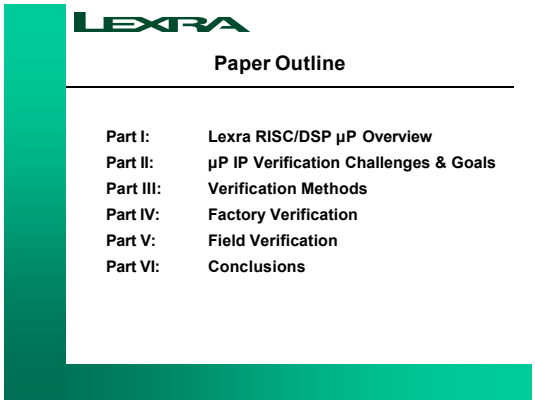
Franklin Hooker
Director of software development, Lexra, Inc.

Bob Gelinas
Processor architect and designer, Lexra, Inc.

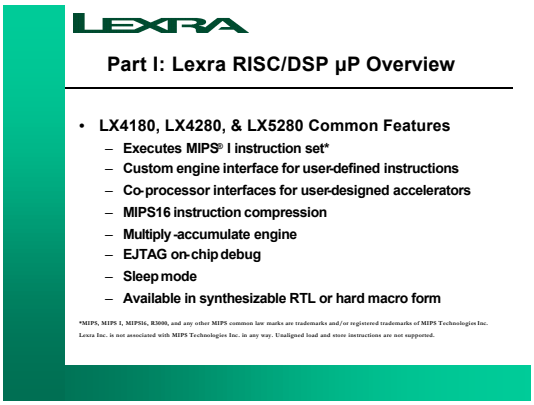
Jonah Probell
Logic and PLD designer, Lexra, Inc.

Todd Snyder
Design and verification engineer, Lexra, Inc.

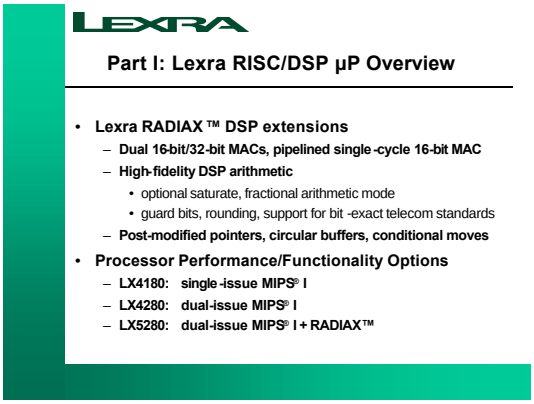
Slide 1: This paper explains the multi-faceted verification strategy Lexra applies to it's Lexra µP cores. These cores are available to customers for IP licensing. Consequently, their verification must be done in two contexts: the "factory", where all customer independent functionality is verified; and the "field", where customer-specific add-ons and system integration are verified. The topic of this paper is the overall framework Lexra uses to satisfy both the factory and field verification requirements. Point verification tools, such as pseudo-random testing, are only described to the first order of detail, and are the subject of separate papers.



Slide 2: The paper is divided into six parts. Part I gives an overview of the Lexra µP cores to substantiate the verification challenges outlined in Part II. Part III describes the collection of point verification tools and methodologies used. Parts IV and V describe how the point tools are used to satisfy the factory and field verification requirements, and Part VI summarizes the conclusion.



Slides 3: Lexra develops RISC (LX4180, LX4280) and RISC/DSP (LX5280) µP cores. All cores execute the MIPS I[1] instruction set architecture (ISA) and provide ISA extensibility through co-processor and custom engine interfaces. All cores also provide MIPS16 code compression, enhanced JTAG on-chip debug (EJTAG), a Multiply-Accumulate (MAC) engine, and a sleep mode option.

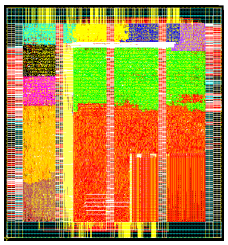


Slide 4: The LX5280 also executes the Radiax™ developed by Lexra as a set of DSP extensions to the MIPS 1 ISA. The LX4180 is single-issue processor, while the LX4280 and LX5280 are dual-issue processors.

1 MIPS, MIPS I, MIPS16, R3000, and any other MIPS common law marks are trademarks and/or registered trademarks of MIPS Technologies Inc. Lexra Inc. is not associated with MIPS Technologies Inc. in any way. Unaligned load and store instructions are not supported.

LEXRA

Part I: Lexra RISC/DSP µP Overview



- LX4180 0.25 µm specs
  - 155 MHz
  - 3.8 mm²
  - 175 mW
- LX5280 0.18 µm specs
  - 200 MHz
  - 6.0 mm²
  - 225 mW

Slide 5: The Lexra µP cores are engineered for ease-of-use. Many user options are provided to facilitate integration into complex system-on-chip (SOC) designs. Examples are configuration settings for cache, co-processor, custom engine, and the system bus, as well as clock and reset options. The are available in synthesizable or hard macro form, and run at high frequencies (200 MHz in 0.18µm).

LEXRA

Part II: Verification Challenges of µP IP

- Inherent complexity of ISA verification
  - Op-codes: MIPS I, MIPS16, RADIAX, user-defined
  - Pipeline: interlocks, exceptions, interrupts
- User configurations – thousands!!
  - ICACHE, DCACHE, IRAM, DRAM, 1KB-64KB
  - Co-processors (0-3) & custom engines (01)
  - Bus controllers – LBC/CBUS, synchronous/asynchronous
  - EJTAG & MAC options
- Embedded system constraints
  - Processor access can be limited by the surrounding SOC
  - High on-chip clock frequencies can exceed test capabilities

Slide 6: The inherent complexity of verifying an ISA in a fixed standard part is well known to be high from industry experience. The complexity is compounded in two ways for µP cores. The first is by the extensibility and number of processor configurations, which creates an explosion in the number of cases that must be tested. The second is a reduction in observability and controllability incurred by embedding a µP core into an SOC. Verifying a configurable µP core intended for an embedded system is therefore a bigger and more difficult problem than verifying an fixed configuration standalone µP chip.

LEXRA

Part II: Verification Goals for µP IP

- Confirm the correct execution of compiled code
  - Processor works with standard tool chains (Green Hills, GNU, …)
  - Processor runs application code developed with standard tool chains
  - Processor runs standard operating systems (VxWorks, Nucleus, …)
- Validate the instruction set architecture specification
  - Rigorous and detailed verification of instruction set execution
  - Cover every instruction, every mode, and all interactions
- Verify the SOC integration features
  - Custom engine and co-processor capabilities
  - System bus and memory controllers

Slide 7: The ultimate goal of a verification strategy is to guarantee that the processor will correctly execute any code written or compiled to the ISA specification. This includes all tool chains, applications, and real-time operating systems (RTOS) developed with the ISA. Since achieving high confidence levels with real application code is inefficient, this empirical goal is usually translated into a rigorous and detailed verification of the ISA specification. But the rigorous approach is still subject to human error, so in practice both goals are pursued: a significant sample of application code is verified to run correctly and a methodical ISA test plan is executed. In addition to these ISA verification goals, the SOC features of the µP core must be verified. These include all of the ISA extensibility features and processor configuration options provided to the SOC designer.

LEXRA

Part III: Verification Methods

- Directed Testing
  - Manually written regression tests
- Pseudo-random testing
  - Machine generated statistical tests
- Software Matrix Validation
  - Real-world applications running in evaluation board
- PLD Prototyping
  - Verifies user's custom engines, co-processors, and bus devices
- On-chip Debug
  - Addresses testing constraints of embedded systems
- HW-SW Co-verification
  - Verify application software on fast simulation model of hardware

Slide 8

Part III: Directed Testing

- **Test specific features and functionality**
  - E.g. multiply-accumulate instructions, exceptions and interrupts
- **Stress input and timing corner cases**
- **Methodical attempt to break design**
  - Black box testing:
    - Verifies functional correctness
    - Written from behavioral specification
    - Uses no knowledge of internal implementation
  - White box testing:
    - Verifies internal paths (e.g. mux forwarding around regfile)
    - Uses knowledge of internal implementation
    - Uses simulation monitors and assertion checkers

Slide 9

Part III: Directed Testing

- Tests run on a complete processor, not a module
- Tests run in multiple modes – hardware interrupts, single step, clock ratios, external sleep mode
- Tests are self-checking assembly and C programs
- Tests cover all processor instructions and features
- Tests are collected into a regression suite:
  - Over 220 assembly and C tests
  - Necessary and sufficient for processor tape-out
  - Used for internal design verification
  - Runs at customer site on customer-specific configuration
  - Runs in <1hr on RTL model

s

Slide 10

Part III: LX4180 Regression Suite Example

C program tests		EJTAG tests	
Multiplier-divider	4	Instruction breakpoints	3
Data cache	5	Data breakpoints	4
Numerical programs	5	Processor breakpoints	7
Demo/benchmark programs	8	TIF/TOF	18
Core tests		Debug exceptions	7
16 bit - load interlock	6	Regular exceptions	6
16 bit - no exceptions	27	General	16
16 bit - exceptions	2		
32 bit - no exceptions	27	Total regression tests	224
32 bit - exceptions	28		
32 bit - reserved inst traps	4		
Custom engine	19		
Co-processor interface	18		
Performance counter	3		
Bus controller	8		

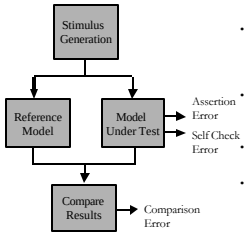
Slide 11

Part III: Pseudo-Random Testing

- Random stimuli constrained to legal interactions
- Complements directed testing, eliminates human bias
- Easy to create complex interactions within the design
- Uncovers bugs without knowing how to expose them
- Trades computer resources for human resources
- Failures harder to debug than in case of directed tests
- Usually activated after directed tests are passing
- Uses and independent reference model

Slide 12

Part III: Pseudo-Random Testing



- Test generator provides user-defined constraints to bias the random test, causing it to exercise the corner cases in the design.
- Stimulus exists as an assembly program which is executed by the reference model and the RTL model.
- The reference model is a high level re-implementation of the design.
- State traces are captured from both models permitting a difference engine to look for errors in the RTL model.

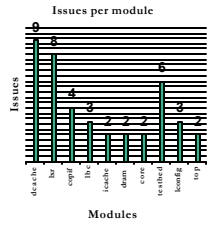
Slide 13

Part III: Pseudo-Random Testing

- Most comparison points between the reference model and RTL model are made at memories
  - General register file
  - Coprocessor registers
  - DRAM
  - System memory
- Some state traces require a complex difference engine to post-process the state traces to detect illegal cases
  - Custom engine registers
  - EJTAG PC trace

Slide 14

Part III: An LX4180 PRT Example



- Most issues related to:
  - Configuration options left untested
  - Exceptions which failed to cancel updates to program state
  - Data cache control
- Issue tracking was initiated after 1st round of directed testing
- Some issues cross module boundaries and appear under multiple modules

Slide 15

Part III: Software Matrix Validation

- Re-validate supported software tool matrix after processor RTL updates:
  - Tool chains: Green Hills, EPI, GNU
  - RTOSs: VxWorks, Nucleus, Thread-X
  - Monitors: PMON, RSS
- Run system level software tests on Lexra's software development board
  - Automated go/no-go tests
  - Use actual silicon to get needed real-time throughput:
    - Test chip - 3 month turnaround from frozen RTL
    - PLD - 1/2 day turnaround closely tracks RTL

Slide 16

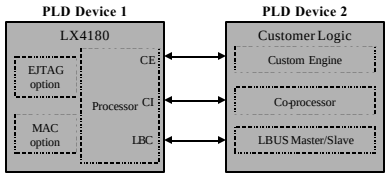
Part III: PLD Prototyping Support

- Development teams are increasingly turning to PLD-based prototype boards for:
  - Hardware-software co-verification before SOC tape-out
  - Hardware platforms to begin software development well before SOC production hardware is available
- Processor IP must support PLD prototyping:
  - Fixed processors in standalone PLD devices
  - Fixed processors in pre-compiled IP blocks integrated with customer-added logic in the same PLD device
  - Configurable processors in synthesizable IP cores that can be compiled with customer-added logic

Slide 17

Part III: PLD Prototyping

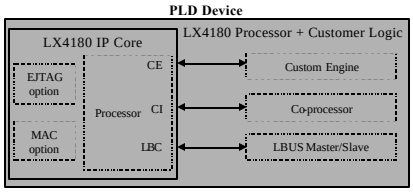
Fixed Processor in Standalone PLD Device



Slide 18

Part III: PLD Prototyping

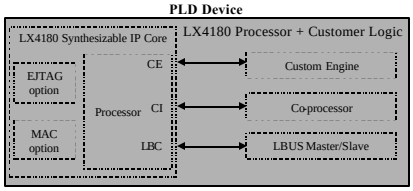
Fixed Processor IP Core with Integrated Customer Logic



Slide 19

Part III: PLD Prototyping

Configurable IP Processor Core Synthesized with Customer Logic



Slide 20

Part III: On-Chip Debug

- Motivation:
  - Hardware bring-up and debug of boards containing SOCs
  - Software development and debug
  - Field diagnostics and upgrades
- Embedded system issues:
  - Target is a configurable processor in a custom SOC
  - On-chip caches and busses are not visible at chip's boundary
  - Code executes at full speed
  - Time-to-market requires standard off-the-shelf interfaces and tool chains

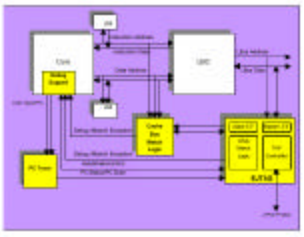
Slide 21

Part III: On-Chip Debug: EJTAG Solution

- Synthesizable EJTAG option works with all Lexra processor IP cores in customer's SOC
- Monitors internal D-cache, I-cache, and system busses
- Works at full processor clock speed
- Provides full-featured debug capabilities:
  - Data, instruction, and processor complex hardware breakpoints
  - Access to system memory through EJTAG probe
  - Single stepping
  - PC Trace
  - Debugger interfaces
- Uses a standard interface and tool chains
  - 5 JTAG pins + 0-10 optional PC Trace pins
  - Off-the-shelf probe and debuggers

Slide 22

Part III: Lexra EJTAG Implementation



Slide 23

Part III: HW-SW Co-verification

- Verify that hardware and software work together before tape-out of the SOC(s)
- Run and debug code at near real-time speed:
  - Fast simulation models
  - PLD hardware prototype
- Fast simulation models:
  - Customer-specific C simulation environment
  - Mix of RTL and C models in a commercial simulation environment
- Simulation approach requires a fast Instruction Set Simulator (ISS) for the processor model

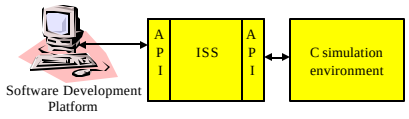
Slide 24

Part III: Instruction Set Simulator

- Fast C model of the processor
  - Executes compiled code
  - Faster than RTL model by 2-3 orders of magnitude
  - Front-end interface to tool chain debuggers
  - Back-end interface to simulation environments
- Front-end interface enables full software development and debug capabilities of commercial tool chains
- Back-end interface enables connections to:
  - Customer-specific C modeling environment
  - Standard Verilog simulation environment
  - Commercial HW-SW co-verification tools

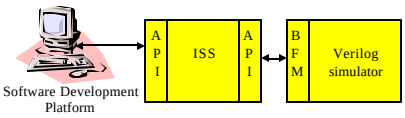
Slide 25

Part III: ISS Integration Options



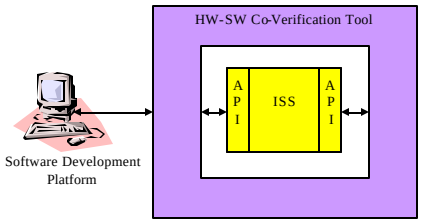
Slide 26

Part III: ISS Integration Options



Slide 27

Part III: ISS Integration Options



Slide 28

Part III: Lexra Instruction Set Simulators

- **Asym-LX**
  - LX4180, LX4280 & LX520 instruction accurate
  - MIPS I and Radiax instruction sets
  - 500K 1M instructions per second
- **PASS2-LX1**
  - LX4180 single issue pipeline accurate
  - MIPS I instruction set
  - 100K - 200K instructions/second
- **PASS2-LX2**
  - LX4280 & LX5280 dual issue pipeline accurate
  - MIPS I and Radiax instruction sets
  - 100K - 200K instructions/second

Slide 29

Part III: Lexra Instruction Set Simulators

- **Debuggers**
  - CLUE built-in standalone debugger
  - Green Hills, EPI, GNU
- **ARIES back-end interface**
  - ABMI: generic API for hardware interfaces (e.g. LBUS BFM)
  - ACOPI: specialized API for behavioral co-processor models
  - AUDI: specialized API for behavioral custom engine models
- **Memory**
  - Models cache size, line size and tracks cache misses
  - Memory usage monitors and breakpoints
- **Pipeline (PASS2)**
  - Model of the hardware pipeline advances on each cycle according to stalls and interlocks
  - Accurate modeling of register bypasses and multiple exceptions

Slide 30

Part IV: Factory Verification of Processor IP

Lexra Verification Objectives

- Instruction set
- Exceptions and interrupts
- Interfaces (LMI, CE, CI, LBUS)
- Processor configuration options
- ASIC / SOC technology mapping

Slide 31

Part IV: Factory Verification of Processor IP

1. **Directed testing**
   - Goal is full line coverage and state-machine coverage
   - Only a limited number of processor configurations are tested
   - Design is believed correct before starting PRT (typically 90%)
   - Directed tests are starting point for the regression suite
2. **Pseudo-random testing**
   - Tests interactions not caught by directed tests
   - Randomly varies processor configurations
   - 10% of total bugs are typically found in PRT
   - 50% of total debug effort is typically spent in PRT
   - Random tests are run until bug rate goes to zero
   - Processor is verified after 48 hrs of zero bugs
   - Run with fixed configuration before tape-out

Slide 32

Part IV: Factory Verification of Processor IP

3. **Regression suite**
   - Updated with gaps in directed tests discovered during PRT
   - Used internally as a quick check on RTL changes & updates
   - Used to verify technology mappings with SmoothCores & PLDs
   - Not sufficient for RTL verification (need regression suite + PRT)
4. **Hardware prototyping**
   - Processor RTL is mapped into a PLD
   - PLD is plugged into test chip socket on development board
   - EJTAG is used to download code and debug new test chips and development boards
5. **Software matrix validation**
   - All supported software is re-validated on a development board
   - Sanity check and confidence builder – no way to measure coverage
   - Run before every RTL release, PLD eliminates delay of test chip fab

Slide 33

Part IV: Factory Verification of Processor IP

Summary

Directed Testing
+
Pseudo-Random Testing
+
Software Matrix Re-validation

Slide 34

Part V: Field Verification

Customer Verification Objectives

- Memory system
- Co-processors
- Custom engines
- System and software
- ASIC / SOC technology mapping

Slide 35

Part V: Field Verification Strategy

1. **RTL verification of standalone processor core**
   - Use same regression suite used by Lexra (shipped with release)
   - Verifies correct setup of processor database release
   - Verifies correct connection of memories
   - Verifies correct connection of custom engines and co-processors
   - Verifies correct technology mapping of Lexra processor
   - Can be extended by customer to functionally verify custom engines and co-processors
2. **RTL verification of integrated processor**
   - Customer specific SOC tests

Slide 36

Part V: Field Verification Strategy

3. **HW-SW co-verification – simulation based**
   - Use instruction accurate or pipeline accurate Lexra ISS
   - Connect Lexra ISS to simulation environment:
     - Option 1: connect to customer-specific C simulation environment directly through the ABMI API
     - Option 2: connect to Verilog simulation environment through a generic Lexra bus functional model for the LBUS
     - Option 3: connect Lexra ISS to a HW-SW co-verification tool through a proprietary bus functional model for the LBUS
   - Model custom engines and co-processors:
     - Option 1: connect behavioral models through the AUDI (custom engine) or ACOPI (co-processor) APIs
     - Option 2: connect Verilog modules through the generic ABMI API using bus functional models for the CE and CI interfaces
   - Write, simulate, and debug code using off-the-shelf tool chains

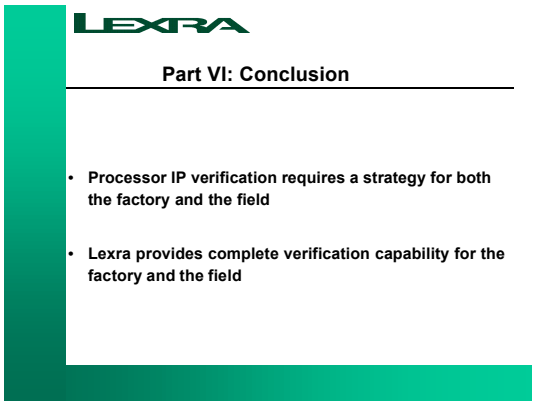
Slide 37

Part V: Field Verification Strategy

3. **Cont'd**

   HW-SW co-verification – HW prototype based
   - Use a PLD-compiled processor core
   - Compile custom engines, co-processors, bus elements, and other required system modules into one or more PLDs
   - Use EJTAG to bring up and debug the prototype system hardware
   - Write, simulate, and debug code using off-the-shelf tool chains
   - Use EJTAG for difficult software debugging of real time embedded systems

Slide 38


LEXRA
Part VI: Conclusion
• Processor IP verification requires a strategy for both the factory and the field
• Lexra provides complete verification capability for the factory and the field

Slide 39