

Formal Verification IPs

Automated SoC protocol checking

DATE'07 16-19 april 2007

OCP-IP / AerieLogic presentation



The potential ABV Answer ...

- Assertion-based Verification (ABV) offers the opportunity to **unify the functional verification**
 - ◆ Assertions are standard (PSL, SVA)
 - ◆ Assertions are reusable all along the design cycle
 - ◆ Assertions can be verified dynamically during simulation and statically with **Formal Verification**
- ABV introduction **eases debug tasks**
 - Functional verification can **start sooner**
 - **Controllability** is enhanced
 - **Observability** is high



... and Formal Verification ...

- Enhances functional verification
 - ♦ **Static** vs. dynamic - No test vectors needed
 - ♦ Allows detection of bugs that are **difficult to find** with a simulator
 - ♦ **Exhaustive** formal computation - 100% functional coverage
- Usable with assertions all along the design flow
 - ♦ During the **RTL design phase** by design engineers to compute expected behaviors and to catch bugs quickly and easily
 - ♦ During the **verification phase** by verification engineers to exhibit complex behaviors and to find remaining difficult bugs
 - ♦ During the **integration phase** by system-level engineers to isolate interaction problems and detect functional performance issues

... lead to ABV Problems

- **Assertions are not so easy to write**

- ◆ High-value assertions are at block-interface level
- ◆ Extract from the spec (how many ?)
- ◆ Code, debug and package them



- **Assertions are not so easy to measure**

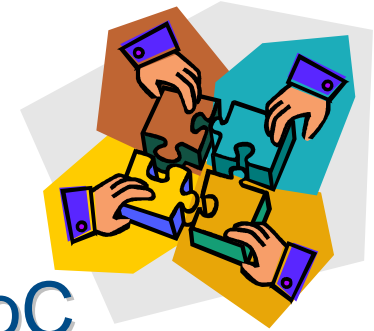
- ◆ Link to the specification, to the design, to the test-plan
- ◆ Measure the quality of the verification achieved

- **Formal Verification needs additional work**

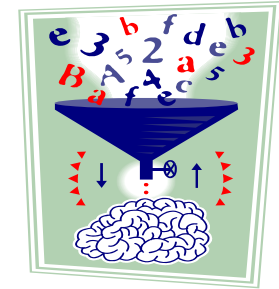
- ◆ Create a sound and complete formal environment
 - Takes usually 80% of the effort

The Need for Protocol Checking

- Protocols are commonly used today to connect **IP blocks** on structured **SoCs**.
- The Protocol is the **back-bone** of the SoC
 - ◆ A protocol failure usually leads to a non-functional chip
 - ◆ IP blocks functional bugs often lead to protocol failures
 - ◆ Functional integration problems are well monitored at protocol level
- Exhaustive simulation of the protocol interface of an IP block is **difficult and time-consuming**



The Challenge



- **Enhance the functional verification flow**
 - ♦ Using **Assertions** advantages
 - ♦ Using **Formal Verification** advantages
 - ♦ Focusing on **global functionalities** at block's interface
 - ♦ **Measuring** the achieved functional verification

- **Enable non-expert users to take advantage of ABV and Formal Verification**
 - ♦ With a **high level of automation**
 - ♦ With a **straightforward verification flow**
 - ♦ With strong support and maintenance

What is a Formal-VIP ?

- It's a **Verification IP** working seamlessly with Formal Verification tools
 - ♦ Automatic usage
 - ♦ Complete documentation, scripts and report generation
 - ♦ Easy setting via GUI or CLI interface
- Provides **automation** to Formal Verification
 - ♦ Builds the formal environment
 - ♦ Instanciates all assertions on the design
 - ♦ Creates regression scripts and coverage analysis

Who uses Formal-VIPs ?

- Formal-VIPs are targetted at **designers** and **non-expert** verification engineers
 - ♦ No knowledge in assertions
 - ♦ No knowledge in formal verification
 - ♦ Possibly limited knowledge in protocols
 - ♦ No time to learn a new tool/methodology
 - ♦ Possibly no knowledge of the DUT internal details
- They also help **expert users**
 - ♦ Provide regression results quickly
 - ♦ Reduce the assertion development effort

Necessary features of a Formal-VIP

1. Ease of Use

- ♦ Same use-flow for all protocols
- ♦ GUI, inline help, automatic fill-in, user-doc linked to protocol spec

2. Automation

- ♦ Wrapper, formal environment, properties, coverage scenarios, regression script, report

3. Flexibility

- ♦ Adapt to user's protocol usage and specific requests
- ♦ Work automatically on several formal tools of the market

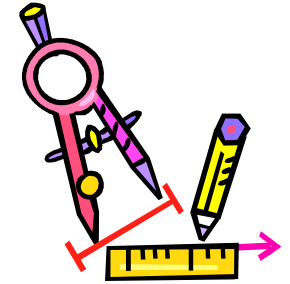
4. Quality

- ♦ Support 100% of protocol
- ♦ Have protocol experts, follow the working groups, formally verify the Formal-VIP on many designs with many configurations

5. Evolution

- ♦ Follow quickly the protocol evolutions
- ♦ Enhance with other automatic properties

Formal-VIP for OCP 2.x (1)



- For IP blocks based on the OCP 2.2/2.1/2.0/Sonics2.4/1.0 protocols
 - ♦ Automatic **instanciation** of assertions and formal environment
 - ♦ Automatic **formal verification** of properties and coverage scenarios
 - ♦ Formal **Proof**, or automatic VCD and testbench in case of **violation**
- Provides **100% compliance** against the OCP protocol specification
 - ♦ Fully aligned on the OCP-IP Compliance document
 - ♦ including multi-threading, tagging, datahandshake, ...
 - ♦ More than 110 properties and hundreds of coverage scenarios
- Provides **100% functional performance** analysis
 - ♦ For detection of dead-locks, loss of data, back-pressure, ...
 - ♦ *Ex: A thread will be busy at most N consecutive cycles*
 - ♦ More than 15 properties dedicated to performance analysis

Formal-VIP for OCP 2.x (2)



- OCP compliance and performance results are **precisely measured**
 - ♦ with **functional metrics** against the OCP protocol specification and the OCP-IP compliance documentation
- Formal-VIP for OCP 2.x is **easy to use and to deploy**
 - ♦ **No knowledge** in assertions or formal verification needed
 - ♦ User sets the design's specific parameters via GUI wizards or a simple configuration file
 - ♦ Choice to check the full protocol or select from available properties to create an automatic script
- **Qualification** of the Formal-VIP for OCP 2.x solution
 - ♦ We participate to the OCP-IP Functional Verification Working Group
 - Co-author of the *OCP-IP Protocol Compliance* document
 - We worked with MIPS, Sonics, Texas-Instruments and Toshiba
 - ♦ Was **formally verified** on more than 60 industrial blocks

Ease of Use of a Formal-VIP (1)

Reload/Save configurations
Create <rtl.conf> for OCP

syntax and semantic checks

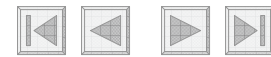
Multi-interfaces support
Straightforward to use

inline help

The screenshot shows the HPKWizard v.1.5.10 configuration window. The title bar includes 'File Configuration' and the AerieLogic logo. The main area is titled 'HPK-AHB Master 1 (1)'. A message box at the top states 'not supported SUPPORT_BURST_INCR=NO and ALWAYS_RESTART_BURST_WITH_INCR=YES'. Below this, a tree view on the left shows 'OCP_2.1 interface' and 'AHB_interface' selected. The right pane, titled 'Master Parameters', contains several sections:

- REAL DESIGN**:
- DATA BUS WIDTH**: 32 (dropdown), **SUPPORT SINGLE TRANSFER**:
- Support Burst**:
 - INCR: INCR4: INCR8: INCR16:
 - WRAP4: WRAP8: WRAP16:
- Support Size**:
 - SIZE: SIZE16: SIZE32: SIZE64:
 - SIZE128: SIZE256: SIZE512: SIZE1024:
- Others**:
 - ALWAYS RESTART BURST WITH INCR: ALWAYS RESTART BURST WITH SINGLE:
 - ALWAYS CANCEL BURST ON ERROR: NEVER CANCEL BURST ON ERROR:
 - CAN CANCEL: (Indicates if the master always cancels its burst transfers after receiving an error response)
 - IS DEFAULT MASTER: MASTER MAX BUSY CYCLES: 3
 - MASTER MAX LOCKED CYCLES: 7 BURST FOUR MAX CYCLES: 12
 - BURST EIGHT MAX CYCLES: 16 BURST SIXTEEN MAX CYCLES: 26

 The bottom of the window has buttons for 'Add', 'Rename', 'Remove', 'Save', and 'Close'.



Ease of Use of a Formal-VIP (2)

The screenshot shows the HPK-Wizard Export dialog box with the following sections and callouts:

- Export of multi-interfaces designs:** Points to the 'Configuration' section at the top of the dialog.
- Select which assertions to export:** Points to the 'Compliance properties' and 'Performance properties' sections, which contain various assertion checkboxes.
- Select the target tool:** Points to the 'Formal Verification', 'Simulation', and 'Close' buttons at the bottom of the dialog.
- Log results:** Points to the 'Log' section at the bottom of the dialog, which displays a list of messages and warnings.

The dialog box includes the following configuration options:

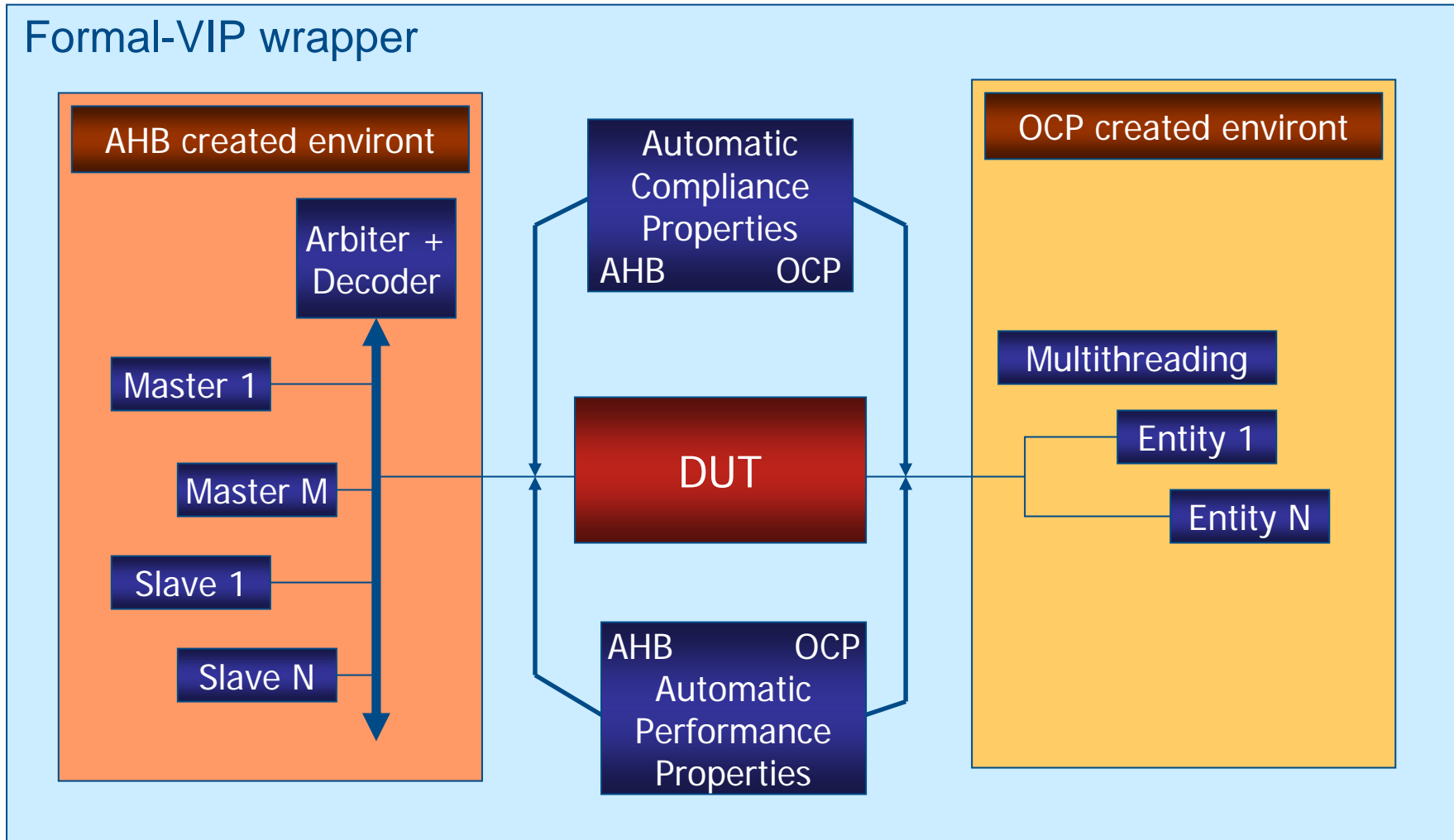
- Interface : OCP 2.2 Port : demo_ Protocol : OCP 2.2
- Compliance properties:
 - burst_value_SRespLast_MRMD_SRMD_0_0
 - burst_value_SRespLast_MRMD_SRMD_0_1
 - burst_value_SRespLast_SINGLE_0_0
 - burst_value_SRespLast_SINGLE_0_1
 - hpk_signal_value_SCmdAccept_during_reset
 - hpk_signal_value_SResp_during_reset
- Performance properties:
 - hpk_performance_latency_SResp_request_to_response_...
 - hpk_performance_latency_SResp_response_to_response_...
 - hpk_performance_max_cycles_SCmdAccept_inactive_2_0...
 - hpk_performance_pipeline_response_slave_10_0
- Coverage scenarios:
 - hpk_coverage_burst_16_incr_RD_0
 - hpk_coverage_burst_16_incr_WRNP_0
 - hpk_coverage_burst_16_incr_WR_0
 - hpk_coverage_burst_16_wrap_RD_0
 - hpk_coverage_burst_16_wrap_WRNP_0
 - hpk_coverage_burst_16_wrap_WR_0
 - hpk_coverage_burst_2_incr_RD_0

The Log section contains the following text:

```
OCP 2.2> and design MUST NOT include combinational cycles
OCP 2.2> WARNING: Parameter empty_design is set to 1, so DUT MUST be an empty one
OCP 2.2> WARNING: pec_ocp_port = pec_slave_port
OCP 2.2> the interface to be tested MUST be a slave one

OCP 2.2> BUILDING LIBRARY FOR AN ENVIRONMENT OCP CORE MASTER INTERFACE --- OCP 2.2 HPK v1.7.0 ...
OCP 2.2> BUILDING LIBRARY FOR THE DUT OCP CORE SLAVE INTERFACE --- OCP 2.2 HPK v1.7.0 ...
```

AHB-OCP Bridge Verification



Formal-VIP Success-Story



- Background
 - ◆ 38 on-the-shelf protocol-based IPs
 - Had been heavily simulated with simulation VIPs
 - ◆ 2 people part-time during 3 months
 - ◆ World-wide semi-conductor company
- Results of F-VIP with Formal Verification
 - ◆ 9 functional bugs automatically found
 - ◆ Tens of wrong protocol documentations
 - ◆ 2 hours of work per IP on average

Conclusion

- Providing Formal-VIPs is key to **broaden the usage of formal verification**
 - ♦ Designers and non-experts need automated solutions
- Providing enough automation for **non-expert users is tedious**
 - ♦ It needs a full-time support team or company
- AerieLogic provides a **robust Formal-VIP solution**
 - ♦ Products already used in the industry for years
 - ♦ Formal-VIP work automatically with **several formal verification tools** of the market