



# OpenVera Assertion Technology

---

Industrial experience in PCI Express  
OVA Checker IP development

OpenVera Developers Forum

April 10, 2003



Solutions for emerging standards



# Agenda

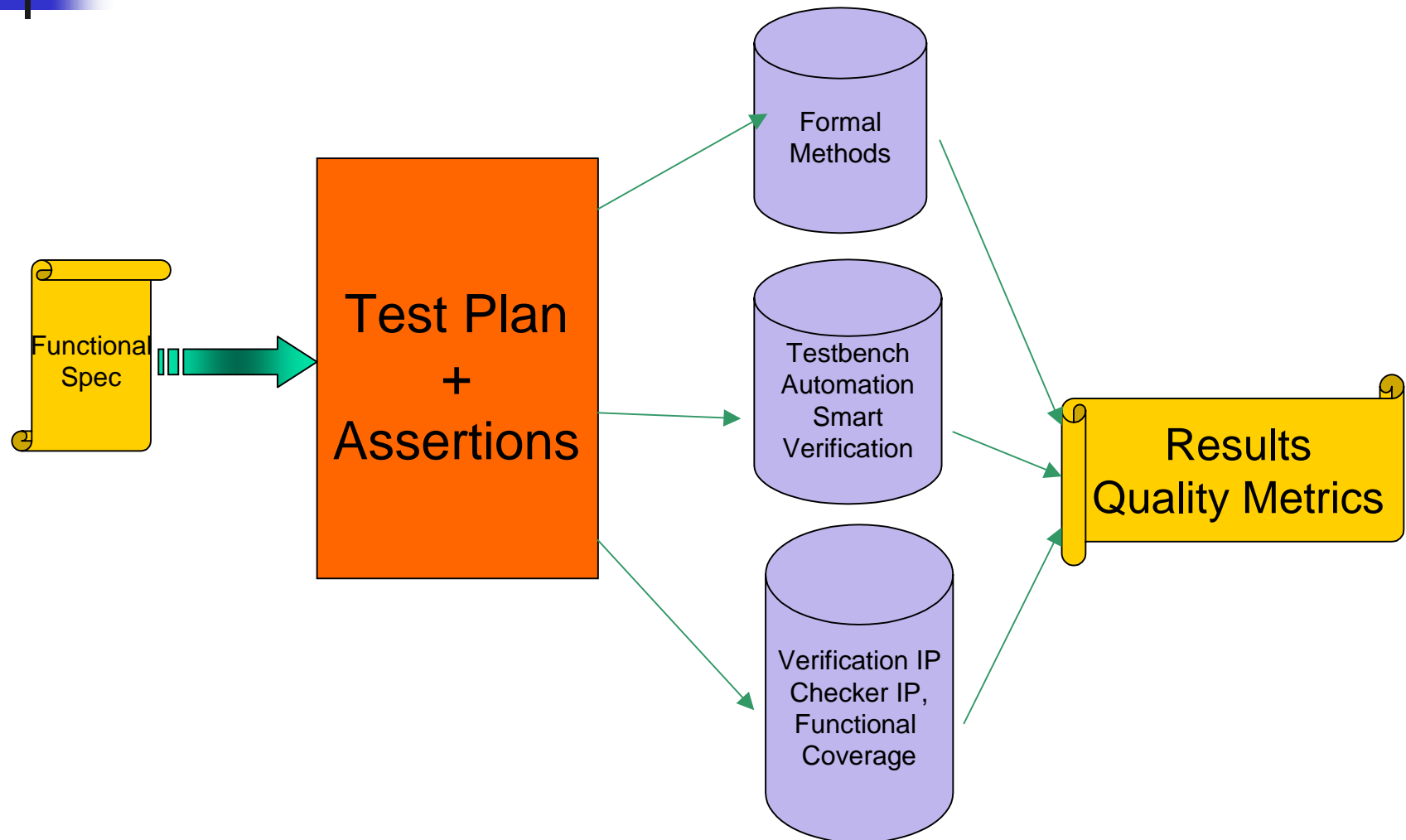
---

- nSys: **Providing complete verification solutions**
- Assertion based Verification and Checker Intellectual Property (IP) for standard protocols
- New connectivity standard protocol: **PCI Express**
- PCI Express Verification
  - **Full Verification IP**
  - **OVA checker IP**
- OVA checker IP: **Architecture and usage**
- Conclusion

# nSys at a glance

- Providing complete verification solution for emerging standards
- nVS family components: **BFM, Checkers, Test Suites**
- nVS family features:
  - Exhaustive Functional testing without writing any code
  - Extensive error injection/detection capabilities
  - Random traffic as well as random response generation
  - Consistency of interface, installation, operation and documentation across nVS family
- nVS including OVA Checkers available for: **PCI Express, PCMCIA, IEEE1284, UART,...**
- <http://www.nsysinc.com>

# Assertion-based verification



# OVA Checker IP Assertions

- Assertion based verification
  - Develop re-usable assertion for standard protocol
  - Encapsulation and parameterized library for re-use
  - Design context specification
  - Protocol coverage directives
  - User specified signal and sampling clock(s)
  - Use Layered Architecture Model
    - OpenVera Modeling Architecture for Verification IP concept

# OVA Language Simple Example

- Concise and abstract
- Declarative representation

```
unit arbiter_check (logic req, logic reset, logic
  clk, logic ack);
clock posedge clk {
bool valid : posedge req && (! reset);
  event ack_after_req :
    if valid then (! ack) *[2..4] #1 ack;
}
assert ack_after_req_2_4: check( ack_after_req );
endunit
```

# OVA Language

## Simple Example

- Concise and abstract
- Declarative representation

```
unit pciex_chk (logic tx, logic rx, logic clk);  
  clock posedge clk {  
    event ex_io_tc_ne_0 :  
      matched ex_correct_header_byte1_received  
    &&  
      ex_type_byte0 === `EX_IO_PKT &&  
      ex_tc_byte1 !== 3'b0;  
  }  
  assert ex_io_tc_ne_0_assert :  
  forbid (ex_io_tc_ne_0);  
endunit
```

# PCI Express

## Brief overview

PCI Express

- New connectivity standard
- Backed by PCISIG
- Enabled by Intel
- PCI shelf life >10 years
- Supports multiple market segments & emerging applications:
  - **Unifying I/O architecture for desktop, mobile, workstation, server, communications platforms, and embedded devices**
- Ability to deliver low cost, high volume solutions:
  - **Cost at or below PCI cost structure at the system level**
- Dual Simplex point-to-point



# PCI Express Verification

## Full verification IP

**Layer 3**

**High level Test Scenario Generator**  
(Basic/Directed/Compliance/Error/Random tests)

**Layer 2**

**Transaction/Traffic generator**

**Layer 1**

**BFM tasks**  
(Command Generation, IP Configuration )

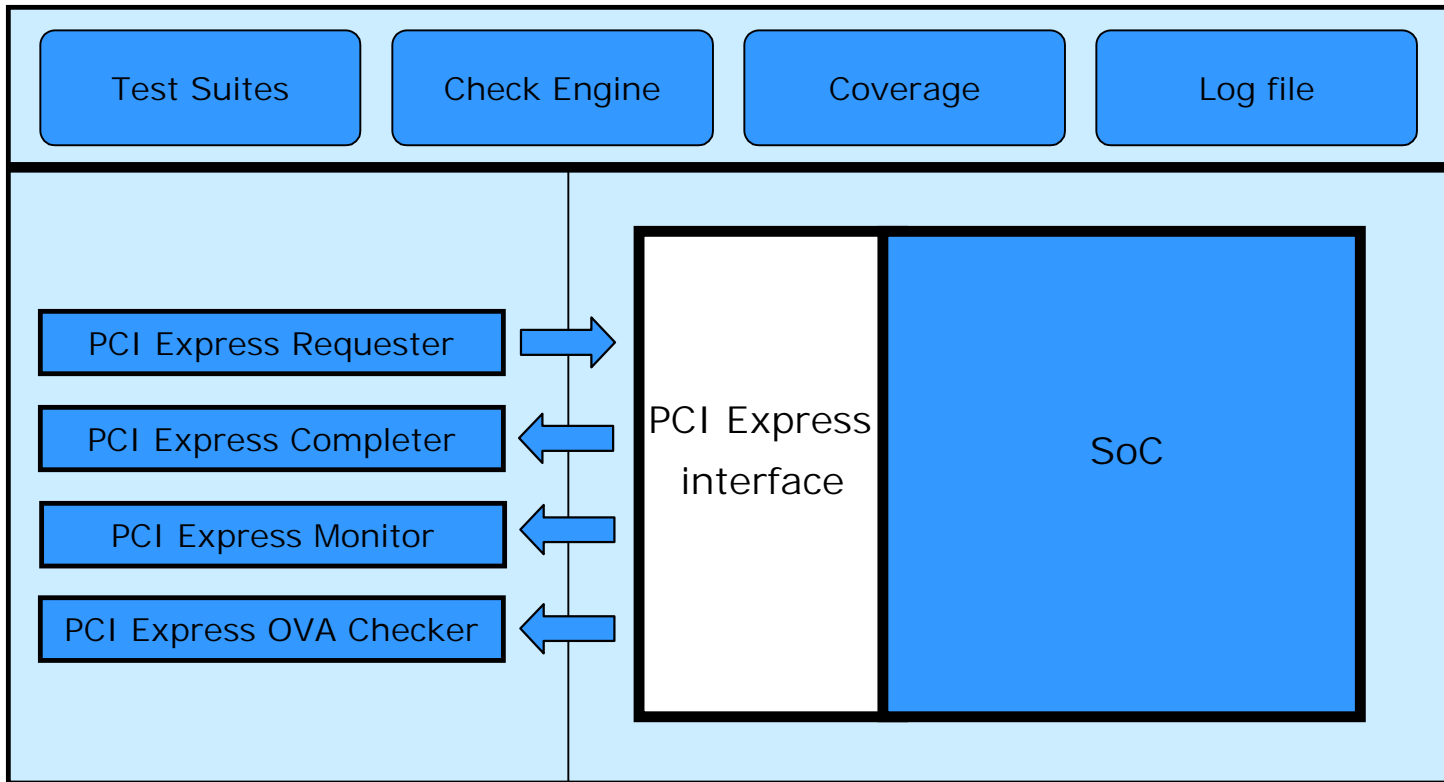
**Layer 0**

**Physical Signal manipulation**  
(.if.vrh)

**VHDL  
DUT**

**Verilog  
DUT**

# OVA checker IP for PCI Express





# Assertion IP reusability

---

- Re-use for Verification
  - Adaptation to a specific instance in design
  - Parameterization
  - Hierarchical construction and encapsulation to hide internal structure
  - User controlled failure message definition
  - Configurability
    - Allows use in dynamic simulation and formal

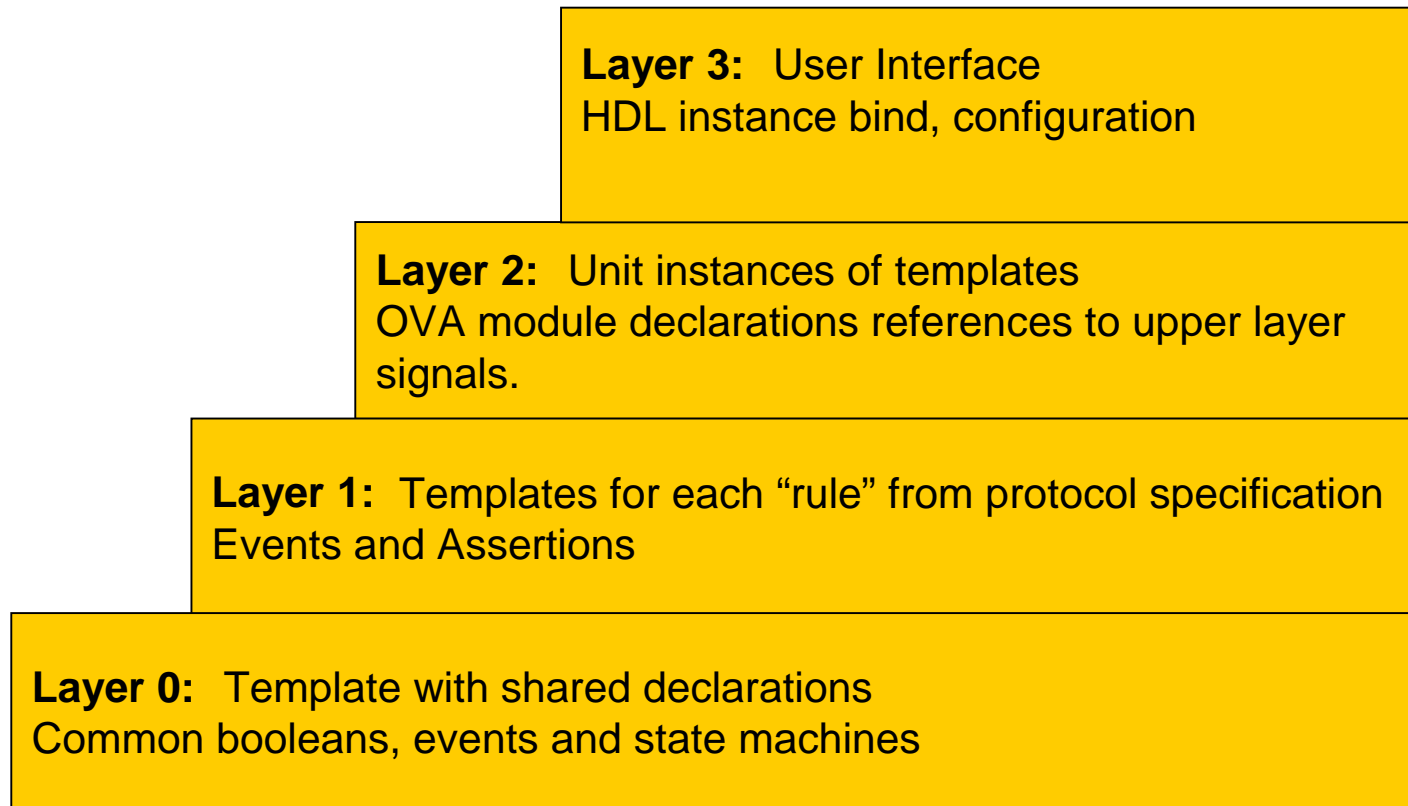
# OVA checker IP

## Common guidelines

- Properties and rules to model
  - Set of rules from timing diagram or specification
  - Identify minimal set of non-redundant and non-overlapping properties
  - Simplify rules if descriptions are complex
  - Rules categorization
  - Make sure rules do not contradict each other

# PCI Express OVA checker IP Layered Architecture

OVA Checker IP  
Architecture



Layered structure eases development and debugging

# PCI Express OVA Checker Features

PCI Express OVA IP

- Provides for checking of:
  - Endpoint/Legacy Endpoint
  - Switch/RC
  - Bridges
- Checking of over 200+ rules
- Individual rules can be enabled/disabled
- Real-time reporting of errors on the clock in which the End symbol of a packet is received
- Stop testing based upon the following: **Maximum Error count/Warning count/Packet count/Time**
- Programmable message levels: **Error/Warning/Info/Debug**





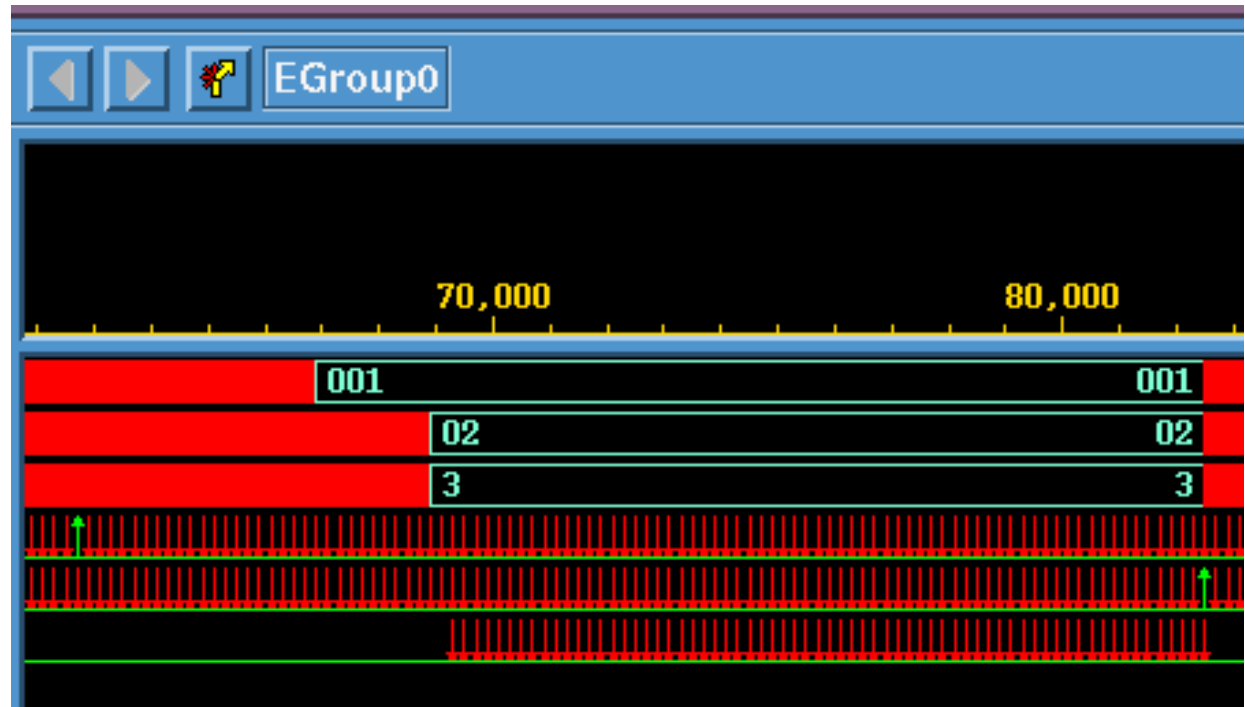
# OVA checker IP example

---

```
clock posedge clk{
bool ex_fmt_00   :   ex_fmt_byte0 === 2'b00;
bool ex_fmt_01   :   ex_fmt_byte0 === 2'b01;
bool ex_fmt_10   :   ex_fmt_byte0 === 2'b10;
bool ex_fmt_11   :   ex_fmt_byte0 === 2'b11;
var ex_4dw;
    ex_4dw <= (ex_fmt_01 || ex_fmt_11) ? 1'b1:1'b0;
event ex_io_request:
    ex_type_byte0 === `EX_IO_PKT;
event ex_io_fmt_ne_3dw :
    matched ex_io_request && ex_4dw;
}
assert ex_io_fmt_ne_3dw_assert : forbid(ex_io_fmt_ne_3dw);
```

# OVA checker IP example: Waveform

seq\_num  
pkt\_type  
pkt\_fmt  
check(ex\_stp\_rcvd)  
check(ex\_end\_rcvd)  
forbid(ex\_io\_fmt\_ne\_3dw)



check(ex\_stp\_rcvd), checks for the start of a TLP &  
check(ex\_end\_rcvd) checks for the end of a TLP

For every I/O TLP type, it is forbidden to have more than 3 Dwords in a packet

# Concluding remarks

- OVA checker IP
  - Reduces time to first test
  - Raises quality and confidence in design
- Use structured architecture for development
  - layers, coding, ...
- Applications
  - Dynamic simulation using standard and proprietary protocols
  - Formal verification using assertion based techniques
- Shortening development life-cycle by months
- nSys: Validated solutions for emerging standards