



**J E D A**  
Technologies

# NSCa Native SystemC Assertion

**Stephen R. Pollock**  
**VP, Marketing and Sales**  
**JEDA Technologies Inc.**

# Agenda

- Introduction to JEDA
- Assertion Verification Methods
- NSCa defined
- NSCa syntax

# About JEDA

- 2002 – Founded
- Mission: to provide world-class SystemC verification automation tools
- > 25 employees
  - ▶ Headquarters – Los Altos, CA
    - 8 employees
  - ▶ Development Center – Beijing, China
    - 17 employees
- Venture Funded
- Experienced Team

# Management Team

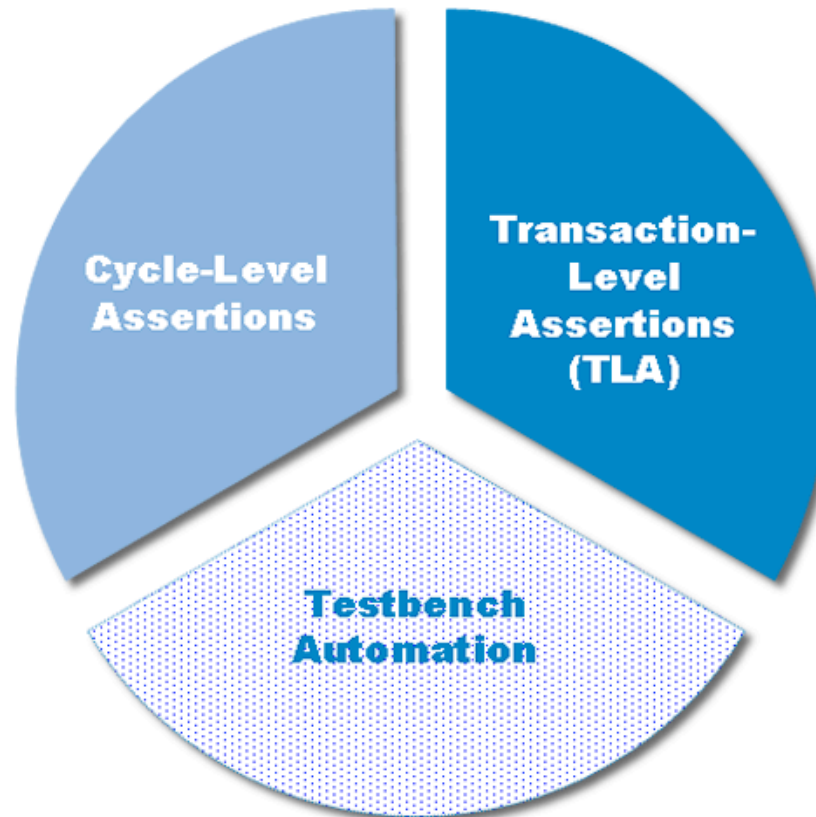
- **Eugene Zhang – President**
  - ▶ Original Vera developer and JEDA contributor
  - ▶ 14+ years management experience: Juniper, Cisco, SUN, Amdahl
  - ▶ MSCE Syracuse U.; MSEE/BSEE Tsinghua U.
  
- **Atsushi Kasuya – CTO and Chief Architect**
  - ▶ Vera language patent author, Vera inventor
  - ▶ JEDA language architect and inventor
  - ▶ 20+ years experience: Juniper, SUN, Xerox PARC
  - ▶ MSCE Santa Clara U.; BSEE Seikei U.
  
- **Tesh Tesfaye – Director, Products**
  - ▶ JEDA contributor
  - ▶ 14+ years experience: Juniper, SUN
  - ▶ Multiple patents author
  - ▶ BSCE UC Santa Barbara
  
- **Stephen R. Pollock – VP, Marketing and Sales**
  - ▶ 25+ years experience in EDA marketing and sales
  - ▶ Launched numerous EDA companies
  - ▶ 10 years IC design experience
  - ▶ BSEE Drexel University

## Experienced Verification Automation Team

- Sun (1993)
  - ▶ Invented **1st generation** HW verification automation language – VERA
- Juniper (1996)
  - ▶ Developed **2nd generation** testbench tool – JEDAX
- JEDA Technologies founded (2002)
  - ▶ Contributions to IEEE 1364 & System-Verilog
  - ▶ Inventor of ESL ABV
  - ▶ Active member of OCP-IP and OSCI
- JEDA Technologies (2006)
  - ▶ Released the **3<sup>rd</sup> generation** – NSCa

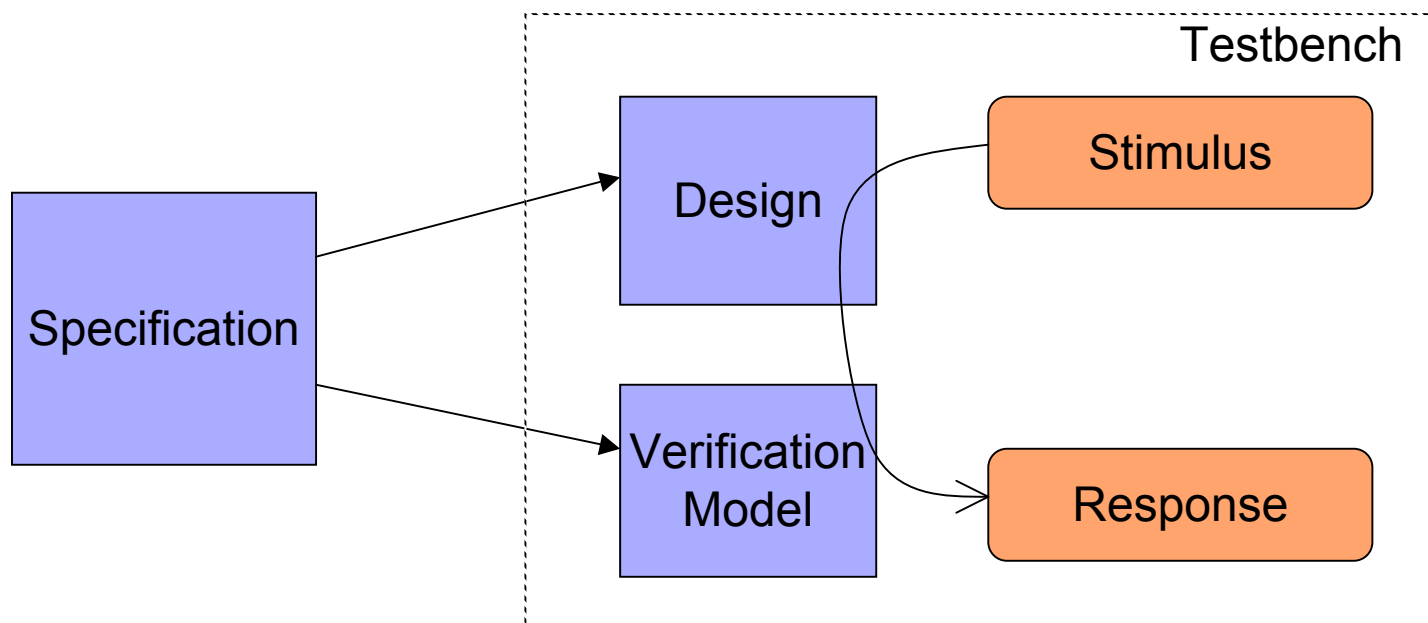
# JEDA's Focus

- SystemC Verification Automation



# What is Verification?

- Process to confirm that the design behaves according to its specification
- Checked by two (or more) brains



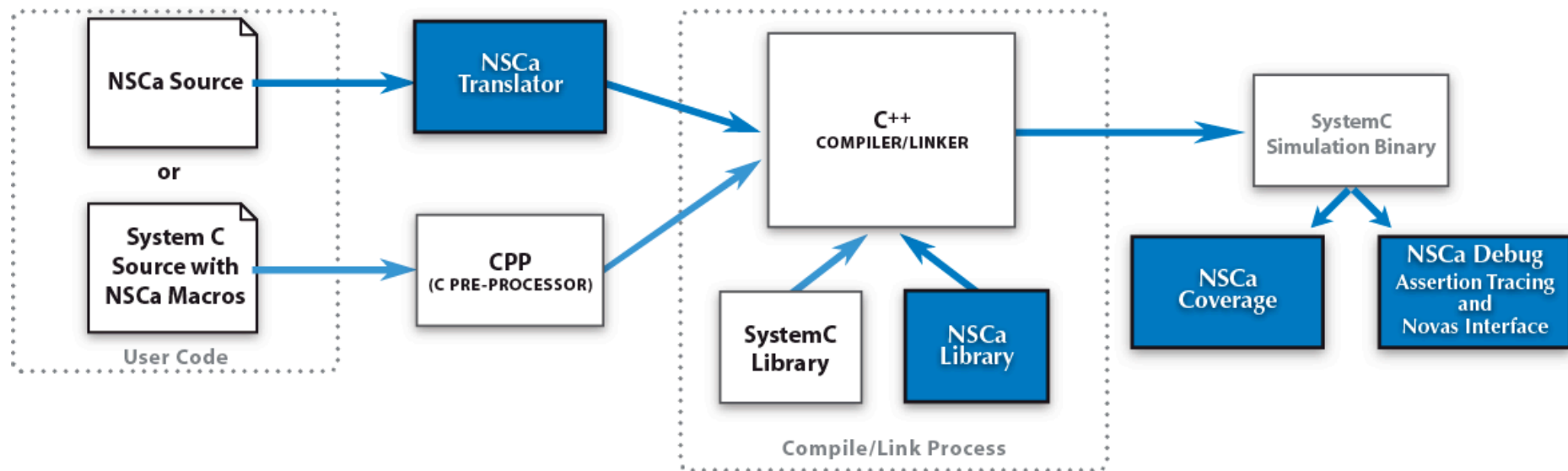
# Primitive Elements for Verification

- Assertion
- Timed (temporal) Expression
- Flexible (Dynamic) Multi-threading
- Garbage Collection
- Pattern Generation

# NSCa is

- SVA Equivalent Functionalities
  - Property Expression
  - Sequence Expression
  - Delay
  - Repetitions (consecutive, non-consecutive, goto)
  - Sequence Ops ( AND, OR, Intersect, First-match, Throughout, Within, etc.)
  - Sequence Match Items
- NSC syntax (extended C++ syntax)
- Macro support for standard C++ compilation
- Natively evaluated in SystemC Thread

# Compile/Link flow



# Temporal Primitives

| NSCa syntax              | Macro             | Function                             |
|--------------------------|-------------------|--------------------------------------|
| nsc_property             | NSC_PROPRTY(..)   | property declaration                 |
| nsc_always               | NSC_ALWAYS(..)    | always property declaration          |
| nsc_assert               | NSC_ASSERT(..)    | property invocation (spawn a thread) |
| nsc_pand                 | NSC_PAND( .. )    | property-and operation               |
| nsc_por                  | NSC_POR( .. )     | property-or operation                |
| nsc_not                  | NSC_NOT( .. )     | property-not operation               |
| ->                       | NSC_IMPLY( .. )   | implication                          |
| =>                       | NSC_NOIMPLY( .. ) | non-overrap implication              |
| <property/sequence name> | NSC_CALL( .. )    | property/sequence instance call      |
| nsc_sequence             | NSC_SEQUENCE(..)  | sequence declaration                 |

# Temporal Primitives (cnt.)

| NSCa syntax     | Macro               | Function                          |
|-----------------|---------------------|-----------------------------------|
| @ [ m : n ]     | NSC_SEQ(m,n, .. )   | m to n cycle delay                |
| ( <s> , <m> )   | NSC_MATCH(..)       | sequence match item               |
| [ * m : n ]     | NSC_CREP(m,n,.. )   | m to n consecutive repetition     |
| [ -> m : n ]    | NSC_GOTO(m,n,..)    | m to n goto repetition            |
| [ = m : n ]     | NSC_NREP(m,n,..)    | m to n non-consecutive repetition |
| nsc_and         | NSC_AND( .. )       | sequence and operation            |
| nsc_or          | NSC_OR( .. )        | sequence or operation             |
| nsc_intersect   | NSC_INTERSECT(..)   | sequence intersection             |
| nsc_within      | NSC_WITHIN(..)      | sequence within                   |
| nsc_throughout  | NSC_THROUGHOUT(.)   | sequence throughout               |
| nsc_first_match | NSC_FIRST_MATCH(..) | sequence first match              |

# Example Code (NSCa Syntax)

```
// req , then gnt within 5 cycle,  
//   then req = 0  
if (  
    ! nsc_sequence(  
        req.read() == 1 @[1,5] gnt.read() == 1  
        @1 req.read() == 0  
    )  
)  
{  
    cout << "Error: request/grant sequence broken!"  
        << endl ;  
}
```

## Example Code (Macro)

```
// req , then gnt within 5 cycle,  
// then req = 0  
if (  
    ! NSC_SEQUENCE(  
        NSC_BOOL( req.read() == 1 ) &&  
        NSC_SEQ( 1, 5, NSC_BOOL(gnt.read() == 1) ) &&  
        NSC_SEQ( 1, 1, NSC_BOOL(req.read() == 0) )  
    )  
)  
{  
    cout << "Error: request/grant sequence broken!"  
        << endl ;  
}
```

# Assertion coverage

Assertion Browser - Jeda NSCa IDE

**Assertion Summary**

| Assertion Name                 | Attempted | Passed | Vacuous | Failed | Percentage | AttempTime(Firs... | DoneTime(First Failed) |
|--------------------------------|-----------|--------|---------|--------|------------|--------------------|------------------------|
| monitor.prop_packet_start      | 16        | 14     | 13      | 2      | 87%        | 21ns               | 21ns                   |
| monitor.prop_packet_end        | 16        | 14     | 14      | 2      | 87%        | 131ns              | 131ns                  |
| monitor.prop_ready_after_rst   | 16        | 16     | 15      | 0      | 100%       | null               | null                   |
| monitor.prop_ready_after_flush | 16        | 14     | 14      | 2      | 87%        | 31ns               | 41ns                   |
| monitor.prop_packet_err        | 16        | 15     | 14      | 1      | 93%        | 101ns              | 101ns                  |
| monitor.prop_packet_ebn        | 16        | 14     | 14      | 2      | 87%        | 131ns              | 131ns                  |
| monitor.prop_packet_process    | 15        | 15     | 15      | 0      | 100%       | null               | null                   |
| monitor.prop_data_resend       | 16        | 15     | 14      | 1      | 93%        | 41ns               | 51ns                   |

OK

# Summary

- JEDA focus is to develop verification automation tools for SystemC
- Experienced Team
  - ▶ The core team invented Vera at Sun 14 years ago
  - ▶ This is our 3rd generation verification automation product
- Our native SystemC assertion approach gives the user the full power of C/C++ to develop assertions
- Free NSCa demo version available for download at JEDA website
  - ▶ <http://www.jedatechnologies.com>