Native SystemC Assertion for OCP property checking

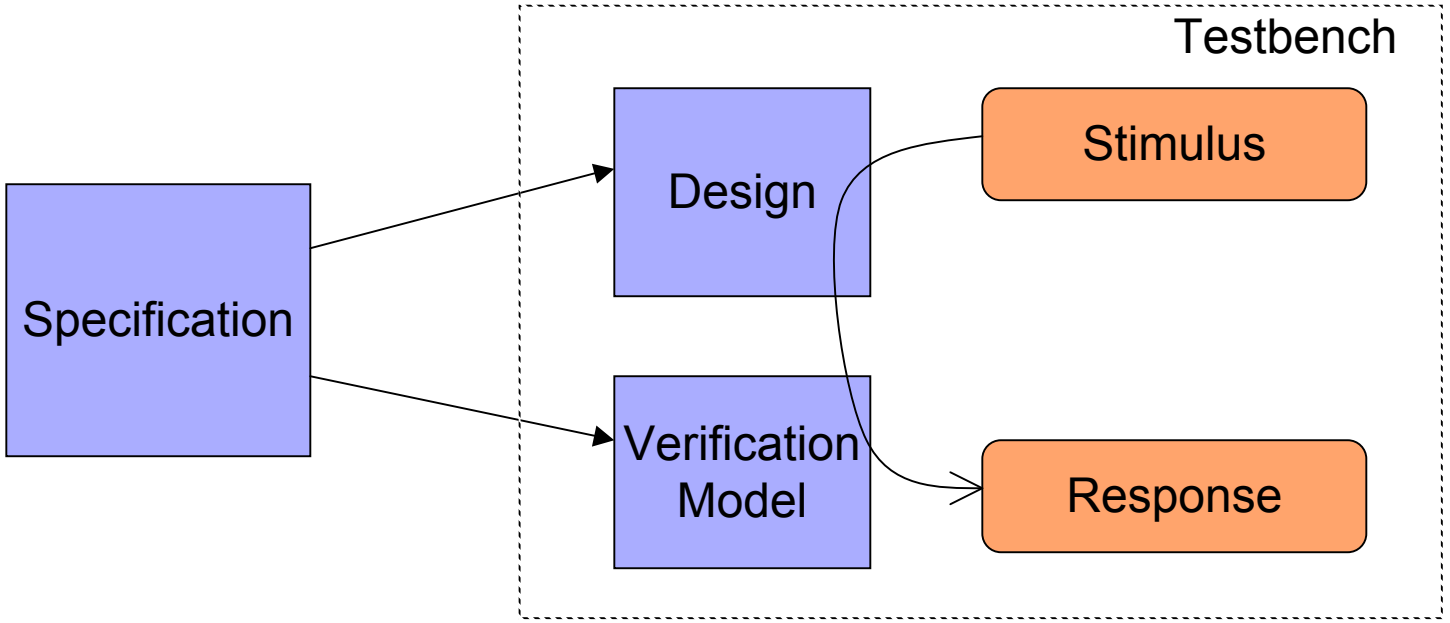
Joe Chou, Sonics Inc.,
Atsushi Kasuya, JEDA Technologies Inc.,
Shanshan Li, JEDA Technologies Inc.,
David Yu, JEDA Technologies Inc.

Agenda

- Assertion Verification Methods
- NSCa defined
- NSCa syntax
- Case study, OCP Assertion

What is Verification?

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



Primitive Elements for Verification

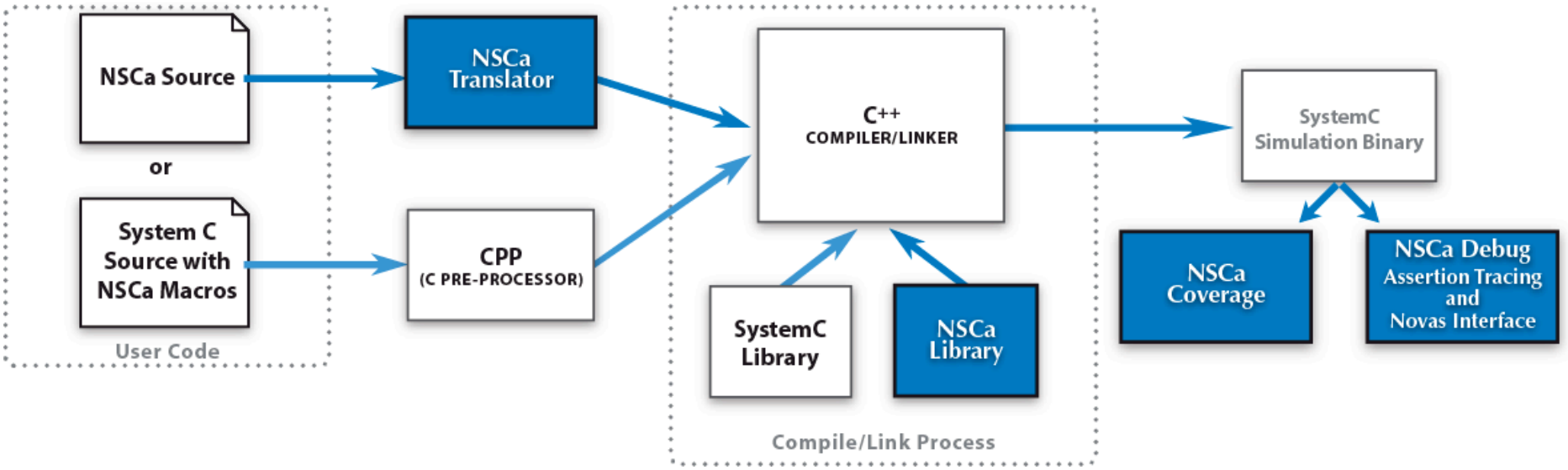
- Assertion
- Timed (temporal) Expression
- Flexible (Dynamic) Multi-threading
- Garbage Collection
- Pattern Generation

NSCa is

- SVA Equivalent Functionalities
  - Property Expression
  - Sequence Expression
  - Delay
  - Repetitions (consective, non-consective, 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
I->	NSC_IMPLY(..)	implication
I=>	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

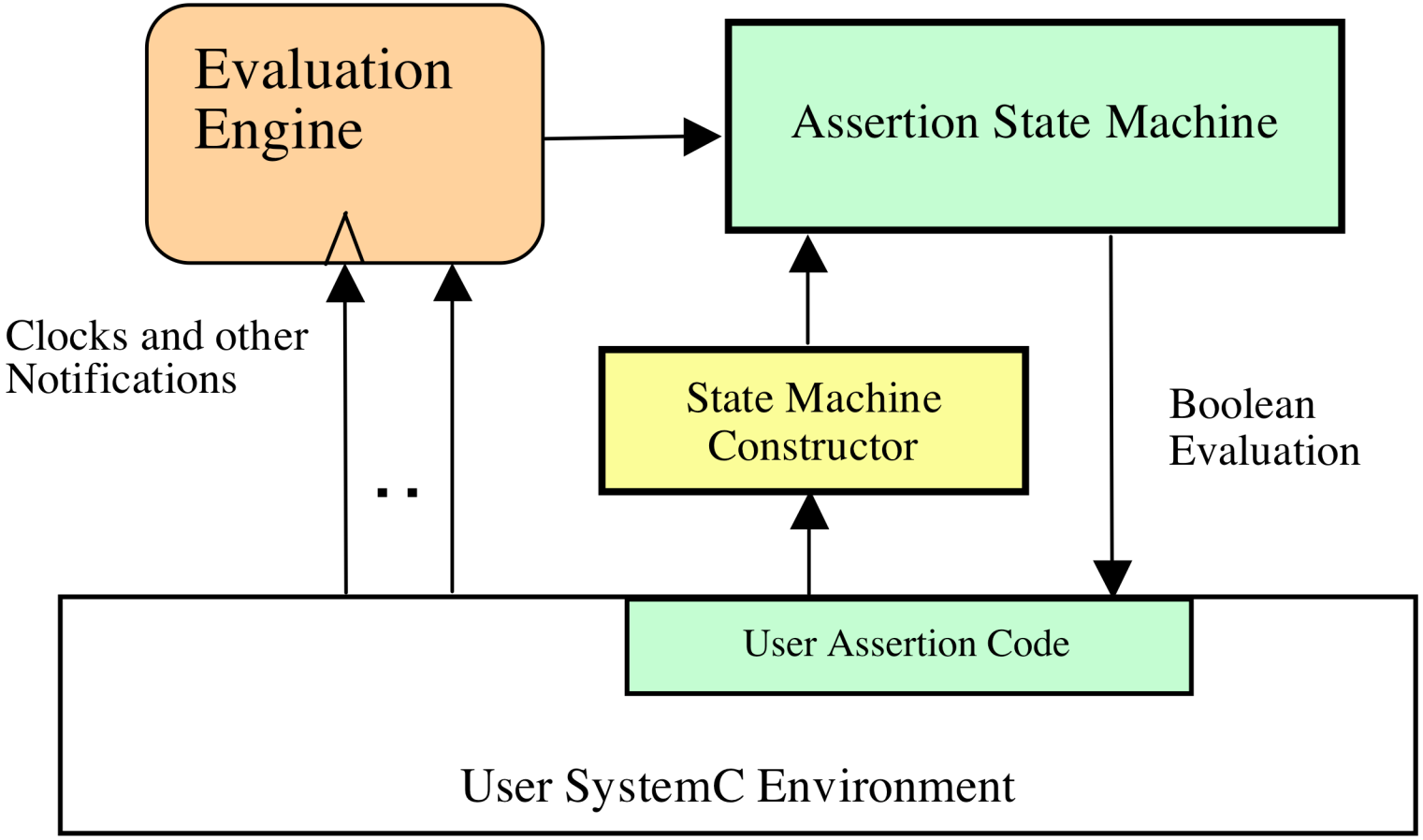
Example Code (NSCa Syntax)

```
// 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  ;
}
```

NSCa Evaluation Engine



Case study: OCP property checking

Illustration: simple timing diagram

- When MCmd (non-IDLE), must be accepted in 3 cycles

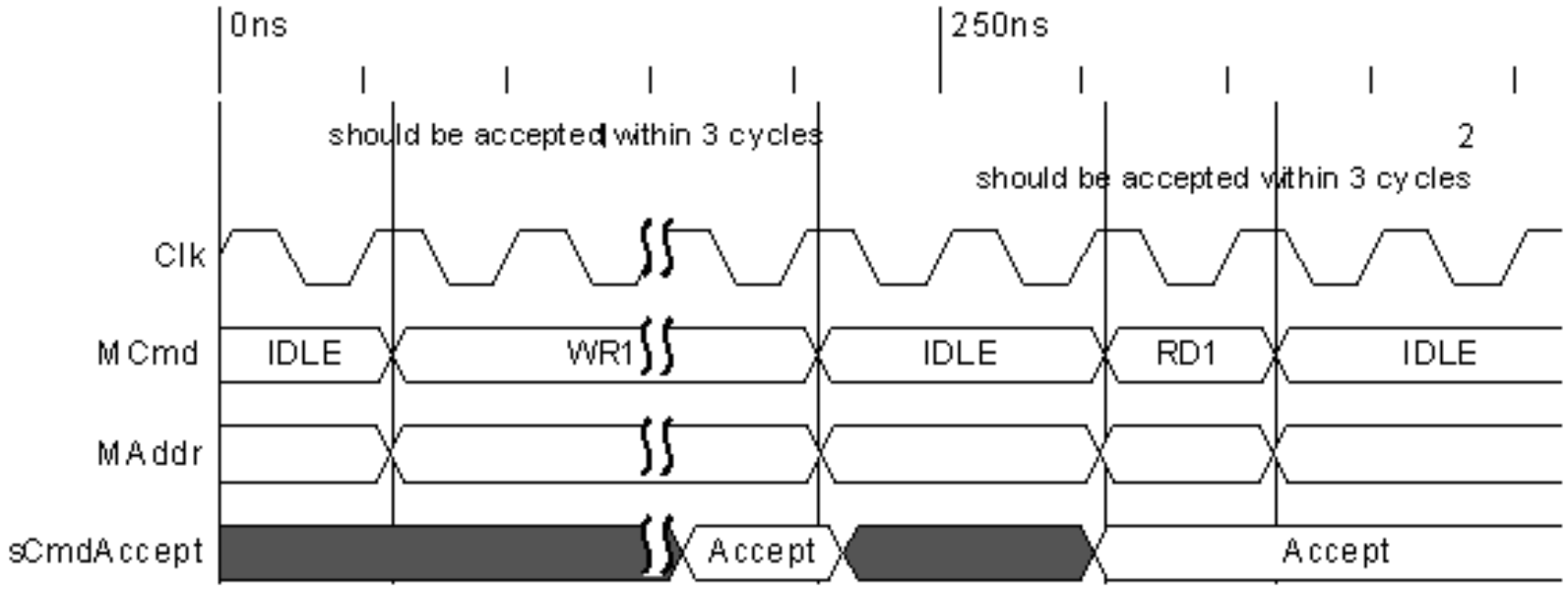


Illustration: checks using SystemC

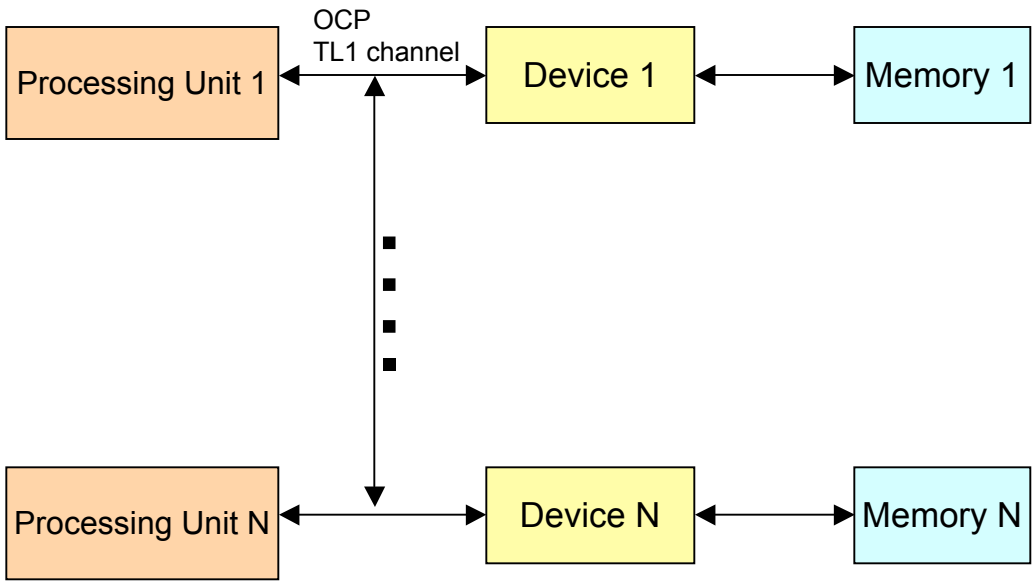
- When MCmd (non-IDLE), must be accepted in 3 cycles

```
int NSCaOCPChecker::SCmdAcceptWithin_3 () {
  int cnt =0 ;
  int const old_Mcmd  = MCmd.read();
  while(cnt++ < 3 ) {
    if(SCmdAccept.read()) {
      cout << "SCmdAcceptWithin_3 succeed @"<< sc_time_stamp()<<endl;
      return 1;
    } else if ( old_Mcmd != MCmd.read()) {
      cout << "SCmdAcceptWithin_3 failed (0) @"<< sc_time_stamp()<<endl;
      assert(0);
    }
    wait();
  }
  cout <<  "SCmdAcceptWithin_3 failed(1) @"<< sc_time_stamp()<<endl;
  assert(0);
  return 0;
}
void NSCaOCPChecker::manual_checker () {
  if ( m_last_MCmd == 0 && MCmd.read()!=0){
    sc_spawn_options op;
    op.set_sensitivity(&CLK.posedge_event());
    sc_spawn(sc_bind(&NSCaOCPChecker::SCmdAcceptWithin_3, this), NULL, &op);
  };
  m_last_MCmd = MCmd.read() ;
}
```

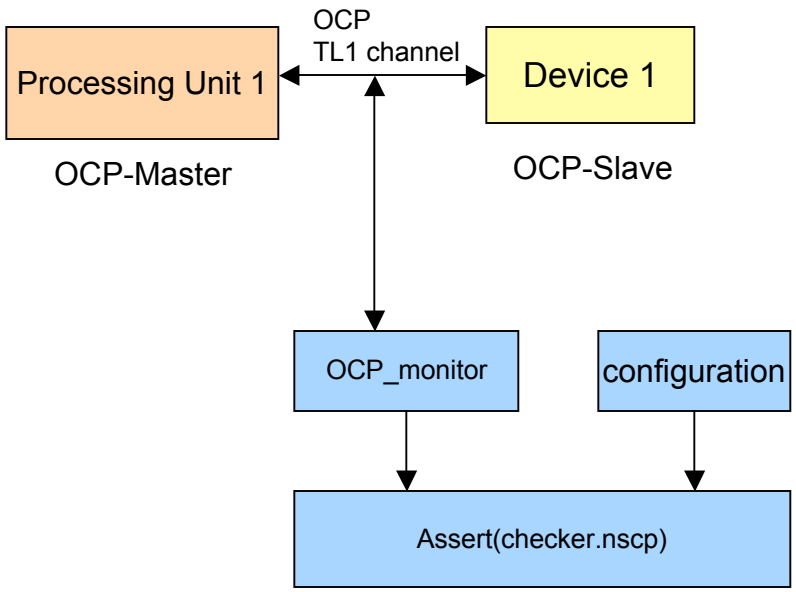
Illustration: checks using NSCa

```
nsc_property NSCaOCPChecker::prop_scmdaccept_asserted_within_three_cycles() {
  int tmp ;
  ( ( (MCmd.read()==OCP_MCMD_IDLE) @1 (MCmd.read()!=OCP_MCMD_IDLE) ),
    tmp=MCmd.read())
    |-> (tmp==MCmd.read()) [*0:2] @1 SCmdAccept.read());
}
```

Customer challenge



Testbench environment



Example(1) Assertion spec.

3.3.1 request_exact_SThreadBusy

Nomenclature

Protocol hierarchy	Request
Signal group	Dataflow – Thread Extensions
Critical signals	MCmd
Assertion type	Value
Enable parameter	sthreadbusy_exact=1

Description

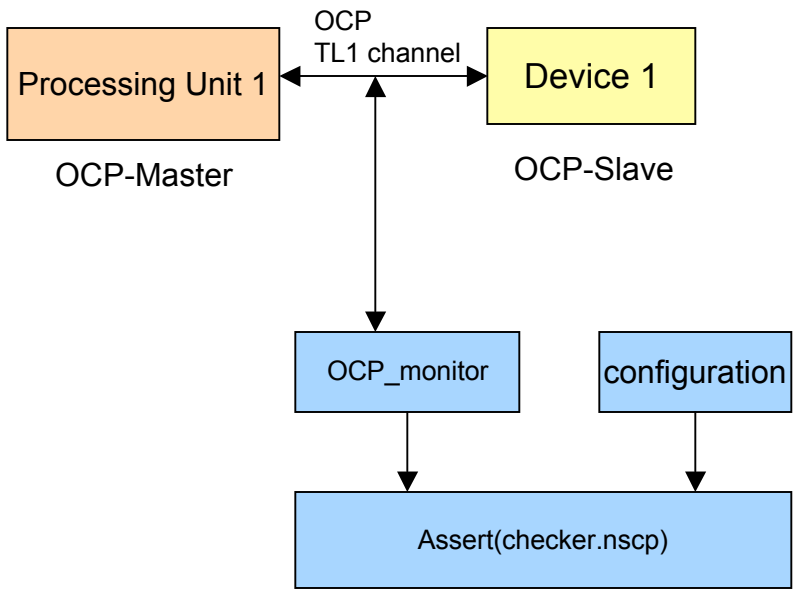
If sthreadbusy_exact = 1, when a given slave thread is busy, the master must stay idle on this thread.

Example (1) implementation

```
//-------------------------------------------------------------------------
// If sthreadbusy_exact = 1, when a given slave thread is busy, the master
// must stay idle on this thread.
//-------------------------------------------------------------------------
 nsc_property NSCaOCPChecker::request_exact_SThreadBusy () {
   nsc_always
   (
     (
       (MCmd.read()!=OCP_MCMD_IDLE)
       &&(!MCmd_X_Z())
       &&(!is_reset_asserted())
     )
     |-> (
       !is_MCmd_when_SThreadBusy()
     )
   );
 }

//check if the cmd thread id is busy
 bool NSCaOCPChecker::is_MCmd_when_SThreadBusy() {
   sc_lv<THREADS_WIDTH> id = MThreadID.read();
   sc_lv<THREADS> busy = SThreadBusy.read();
   sc_lv<3> cmd = MCmd.read();

   int tmp = (sc_uint<THREADS_WIDTH>)id;
   if((busy[tmp] == '1')&&(cmd != OCP_MCMD_IDLE)&&(!MCmd_X_Z())){
     return true;
   }else{
     return false;
   }
 }
```



Example (2) Assertion spec.

3.4.5 burst_sequence_MAddr_INCR

Nomenclature

Protocol hierarchy	Burst
Signal group	Dataflow – Basic Signals
Critical signals	MAddr
Assertion type	Ordering
Enable parameter	addr=1 burstseq_incr_enable=1 burstlength=1

Description

Within an incrementing burst, the address increases for each new master request by the OCP word size.

Example (2) implementation

```
/*
 * Within an incrementing burst, the address increases for each new master request by the OCP
 * word size.
 */
nsc_property NSCaOCPChecker::burst_sequence_MAddr_INCR() {
  sc_lv<ADDR_WIDTH> old_addr;
  sc_lv<3> old_cmd;
  nsc_always
  (
    (
     (MCmd.read()!=OCP_MCMD_IDLE) && (!MCmd_X_Z())
      && ((sc_uint<BURST_LENGTH_WIDTH>)MBurstLength.read() > 1)
      &&(MBurstSeq.read() == OCP_MBURSTSEQ_INCR),
      old_addr = MAddr.read(),
      old_cmd = MCmd.read() //,cout<<sc_time_stamp()<<" 1 "<<old_addr<<"   "<<old_cmd<<endl
    )
    |=> (
      (
       ((sc_uint<ADDR_WIDTH>)(MAddr.read())-(sc_uint<ADDR_WIDTH>)old_addr == DATA_WIDTH/8)
         &&(MCmd.read() == old_cmd)
       )
       ||(MCmd.read() != old_cmd)

    nsc_throughout
      (nsc_first_match( @[0:$] MCmd.read() == old_cmd))
    )
  );
}
```

Assertion coverage

Property name	attempted	passed	failed	percentage	First error attempt time	First error done time
::monitor::prop_reset_status	31	31	0	100%	(null)	(null)
::monitor::prop_strb_length	31	31	0	100%	(null)	(null)
::monitor::prop_din_hold	31	31	0	100%	(null)	(null)
::monitor::prop_ack_length	31	30	1	96%	110 ns	120 ns
::monitor::prop_dout_check	31	31	0	100%	(null)	(null)
::monitor::prop_transaction_length	31	31	0	100%	(null)	(null)
::monitor::prop_reset_status	0	0	0	0%	(null)	(null)

OCP-IP assertion package

- Based on the OCP-IP Functional Verification Working Group(FVWG) compliance check documentation
- JEDA implemented entire checks, in excess of 70+ checks
- JEDA donated OCP-IP2.0 Dataflow Phase NSCa assertion checks to the OCP-IP community

Take NSCa for a test drive

- Free NSCa demo version available for download at JEDA website
  - http://www.jedatechnologies.net
- The package contains
  - A Free NSCa demo engine package
  - Examples of simple assertion checks
  - An OCP-IP assertion check tutorial using the OCP-IP-master and OCP-IP-slave SystemC models from http://www.ocpip.org