UML to SystemC Utility Development
Rev. 1
Sep. 11th, 2009

Hiroyuki Yagi

Sec. 2, System Design Solution Dept.,

DPD, SBG, CPDG, Sony Corporation

Agenda

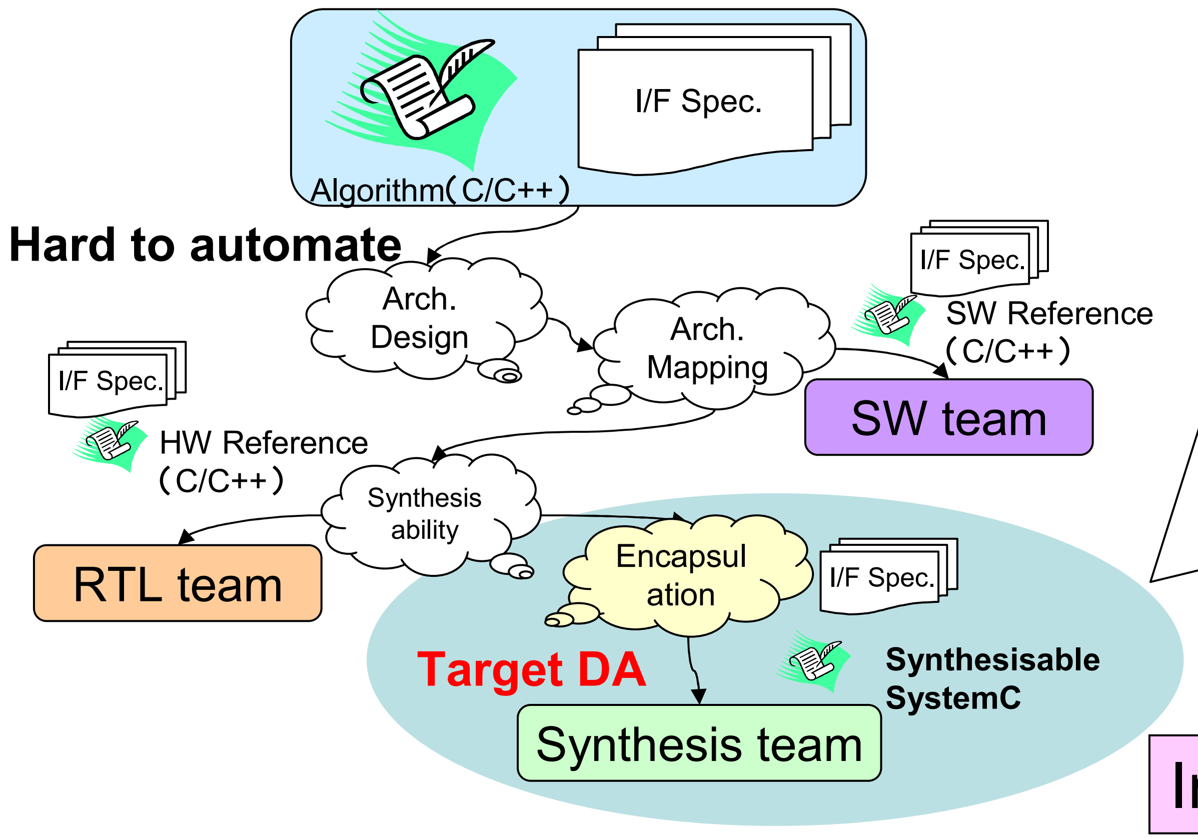
秘 CONFIDENTIAL

- What is the idea?
- Minimum requirements…from Sony side
  - Wish list
  - Block diagram under concern
  - Example: Expected auto-generated SystemC structure description
- Discussion
  - Realizability
  - Schedule/Milestone

What is the idea?

CONFIDENTIAL

- Many hardware developments are motivated to accelerate/minimizing the power/minimizing the area to execute an EXISTING algorithm usually written in C/C++



Encapsulation: Writing SystemC codes from I/F specification is a hard task even for the specialists

•Error-prone: tedious copy-paste based code modification

•Need a lot of efforts: the property of the each ports of the connection should be matched

Impact of this DA is a big

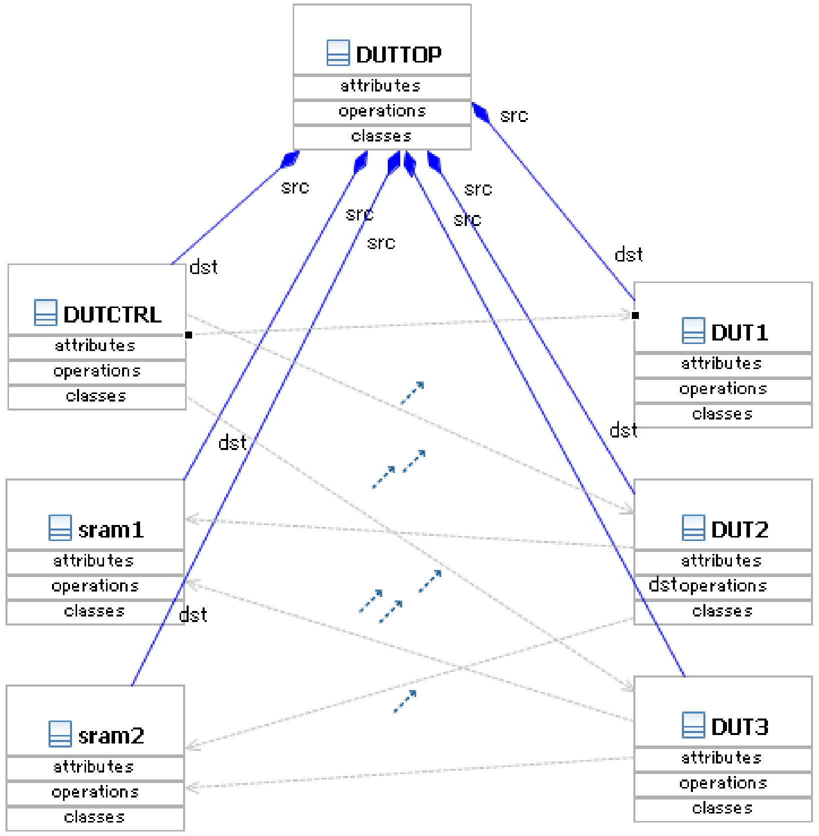
Wish list（revised)

1. Generate structural description（=wrapper+connection)：from UML（SysML) model to SystemC
   - Class diagram: SystemC module generation（device with or without the test bench（TB) module)
   - Component diagram: SystemC module generation（package of modules with or without TB module)
     - Table input form for external interface specification
       - Don't care about the file format
       - Want to have the sorting feature
       - Fields: port, type, name… explained later
       - Need to handle the modular interfaces
     - Need to handle the multi-ports（1-output / multi-input) connection
2. Static code checking of SystemC module
   - Detection of unconnected ports
   - Automatic ports connections selected by user（stub, logical true or false)

Behavioral description is NOT in current scope

Example1: class diagram (revised)

- Class diagram and equivalent SystemC codes for DUT1
- Problem: We can't see the transaction data while the relations between classes are visible



DUT1.h

```
#include <systemc.h>
#include "my_iflib.h"

SC_MODULE(DUT1) {
  my_stream_in<sc_uint<16> >  din;
  my_stream_out<sc_uint<16> > dout;
  my_event_in              start;
  my_event_out              done;

  sc_in<bool>  clk;
  sc_in<bool>  rst_X;

  SC_CTOR(DUT1) {
    SC_CTHREAD(ct_main, clk.pos());
    reset_signal_is(rst_X, false);
  }

  void ct_main(void);
};
```

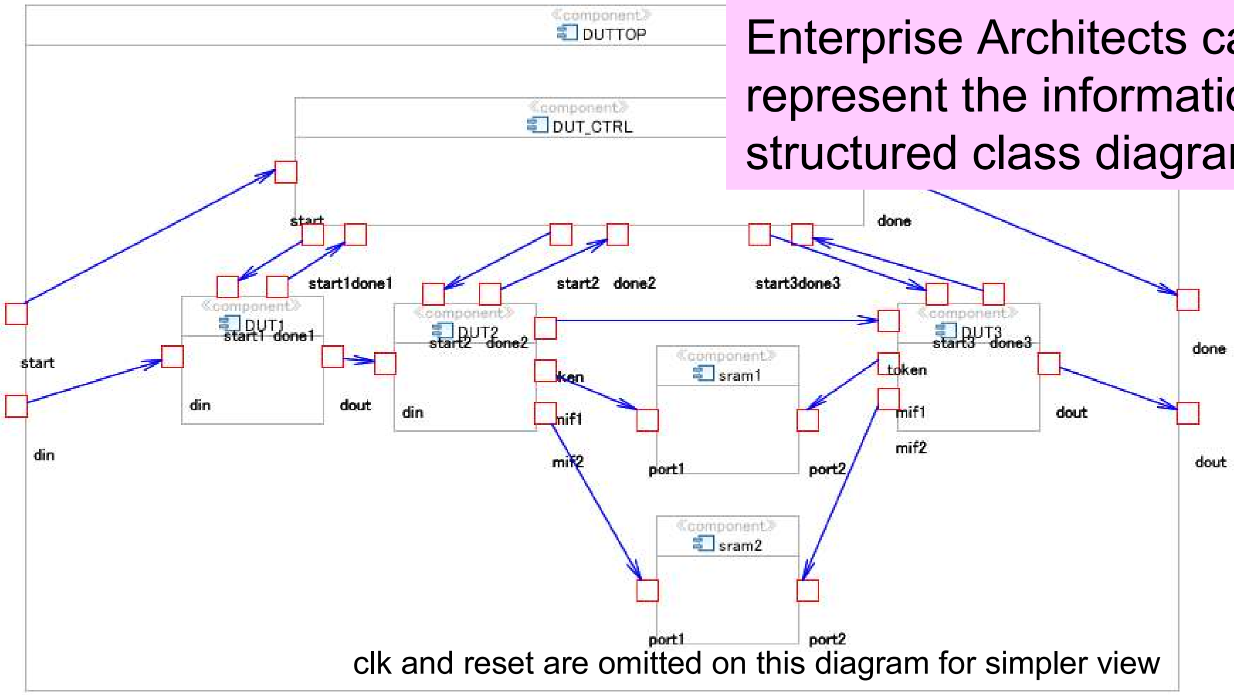
DUT1.cc

```
#include "DUT1.h"

void DUT1::ct_main(void) {
  start.reset();
  done.reset();
  din.reset();
  dout.reset();
  wait();
  while(true) {
    start.wait();
    for(int i=0 ; i<N ; i++) {
      ... = din.get();
      ...
      dout.put(...);
    }
    done.notify();
  }
}
```

Example2: Component diagram（revised)

Enterprise Architects can represent the information with structured class diagram



clk and reset are omitted on this diagram for simpler view

UML component diagram (or composite structure diagram)

Each should be synchronized as they are semantically identical

- Optional : We want to manage the data with a spread sheet like this.

source	# of dest.	destination	dest.2	name	kind	type	temp. arg.	array
DUTCTRL	0	DUT1		START	port	bool	0	0
DUT1	0	DUTCTRL		done	port	bool	0	0
DUT1	0	DUT2		din	mystreamIF	sc_uint	16	0

DESIGN PLATFORM DIVISION
SEMICONDUCTOR BUSINESS GROUP

Generated SystemC structural description

秘 CONFIDENTIAL

Expecting code example

DUTTOP.h

```
#include <systemc.h>      } including libs for synthesis
#include "my_iflib.h"
#include "DUT1.h"
#include "DUT2.h"
#include "DUT3.h"
#include "DUT_CTRL.h"

SC_MODULE(DUTTOP) {
  my_stream_in <sc_uint<16> >   din;    }
  my_stream_out<sc_uint<16> >   dout;   } synthesisable ports
  my_event_in                   start;  }
  my_event_out                  done;   }
  my_sram                       sram1;  } instantiation of memories
  my_sram                       sram2;  }

  sc_in<bool>                   clk;    } clock and reset signal
  sc_in<bool>                   rst_X;  }

  DUT1                          iDUT1;     }
  DUT2                          iDUT2;     } instantitation of sub-
  DUT3                          iDUT3;     } modules
  DUT_CTRL                      iDUT_CTRL; }

  my_stream<sc_uint<16> >       DUT1_dout;
  my_stream<sc_uint<1> >        DUT2_token;

  my_event                      DUT1_start; }
  my_event                      DUT1_done;  }
  my_event                      DUT2_start; } event definition
  my_event                      DUT2_done;  }
  my_event                      DUT3_start; }
  my_event                      DUT3_done;  }
```

```
SC_CTOR(DUTTOP) {
    iDUT1.din    (din);
    iDUT1.dout   (DUT1_dout);
    iDUT1.start  (DUT1_start);
    iDUT1.done   (DUT1_done);
    iDUT1.clk    (clk);
    iDUT1.rst_X  (rst_X);

    iDUT2.din    (DUT1_dout);
    iDUT2.token  (DUT2_token);
    iDUT2.mif1   (sram1);
    iDUT2.mif2   (sram2);
    iDUT2.start  (DUT2_start);
    iDUT2.done   (DUT2_done);
    iDUT2.clk    (clk);
    iDUT2.rst_X  (rst_X);

    iDUT3.token  (DUT2_token);
    iDUT3.mif1   (sram1);
    iDUT3.mif2   (sram2);
    iDUT3.dout   (dout);
    iDUT3.start  (DUT3_start);
    iDUT3.done   (DUT3_done);
    iDUT3.clk    (clk);
    iDUT3.rst_X  (rst_X);

    iDUT_CTRL.start   (start);
    iDUT_CTRL.start1  (DUT1_start);
    iDUT_CTRL.start2  (DUT2_start);
    iDUT_CTRL.start3  (DUT3_start);
    iDUT_CTRL.done1   (DUT1_done);
    iDUT_CTRL.done2   (DUT2_done);
    iDUT_CTRL.done3   (DUT3_done);
    iDUT_CTRL.done    (done);
    iDUT_CTRL.clk     (clk);
    iDUT_CTRL.rst_X   (rst_X);

    sram1.clk    (clk);
    sram2.clk    (clk);
  }
};
```

connecting modules in the constructor

note:

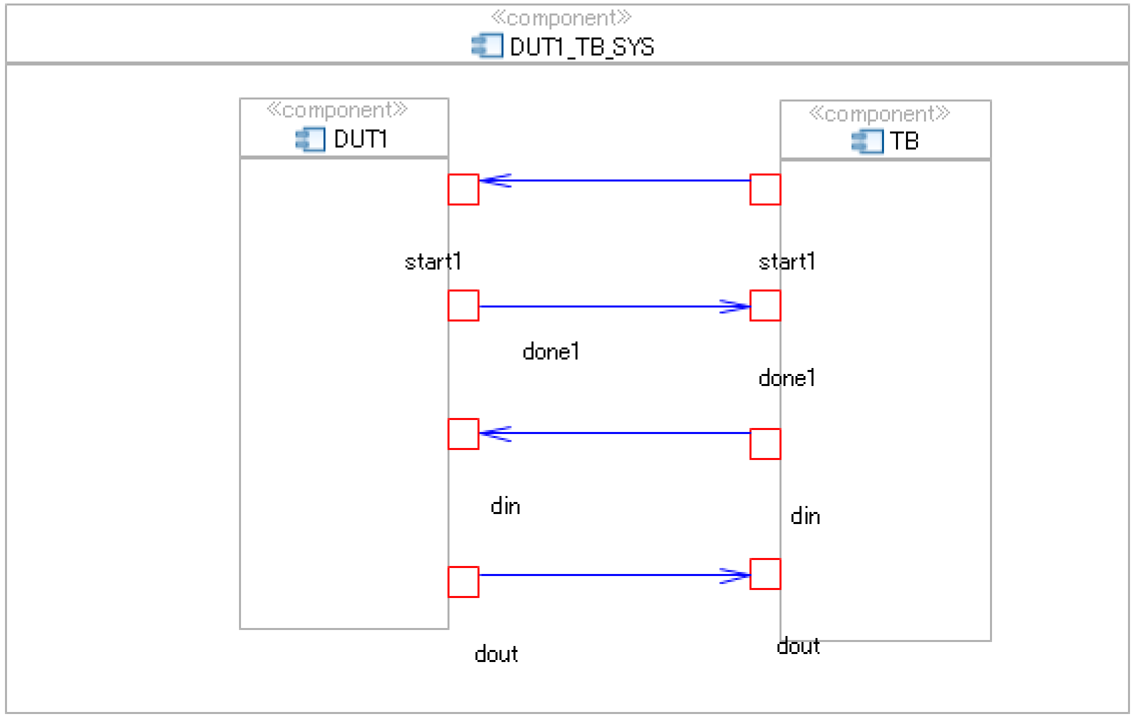
•Hierarchical modules can assume there is no thread inside of the brocks

•Addition of the glue logic is possible between leaf-modules

•Reset action is a must for leaf modules

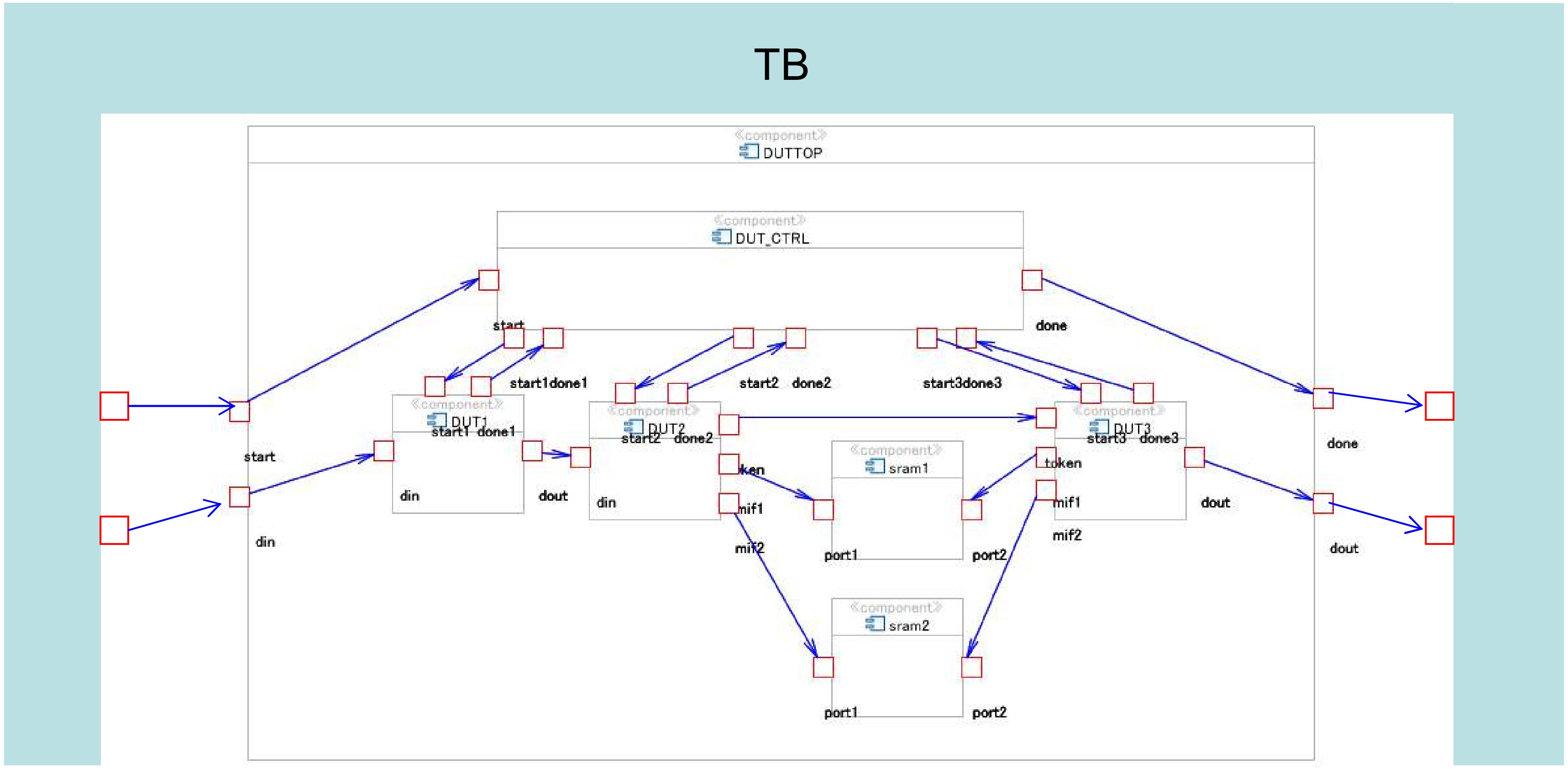
Device with a test bench

- Device can't be tested stand alone
  - Passing the compilation is a must.
- Test bench is a mirror module of the target device, except the attribute for the I/O directions are reversed



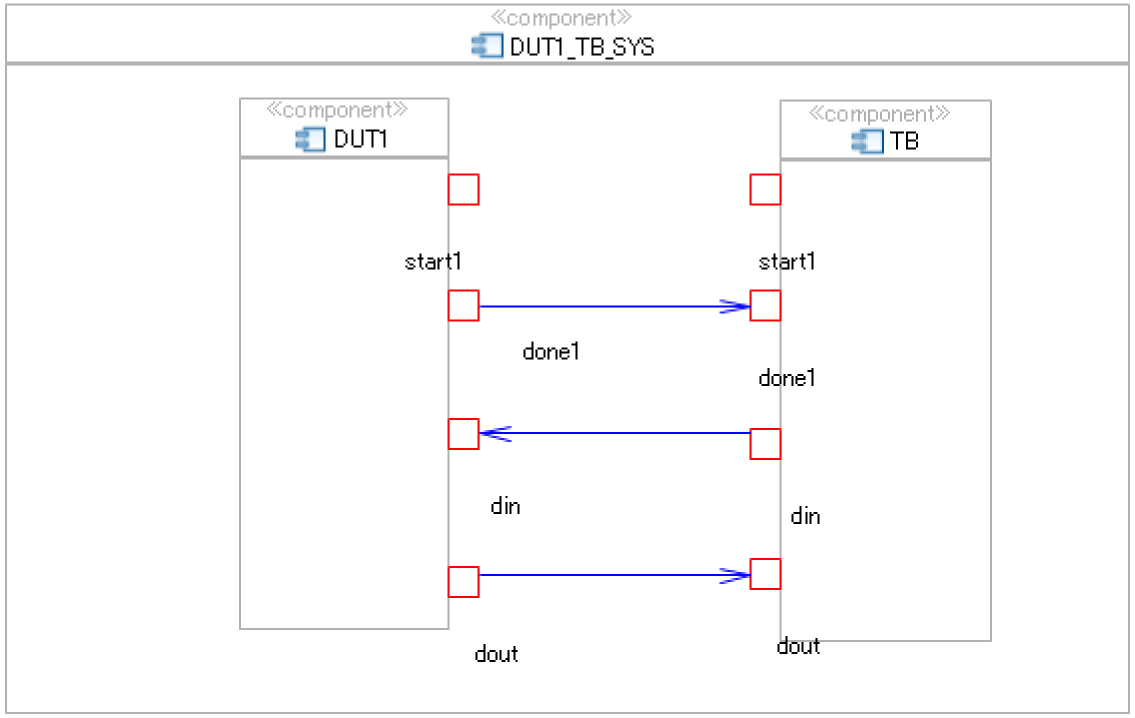
System with a test bench

- Drive a package with the test bench
  - Passing the compilation is a must
- Test bench is a mirror module of the target system, except the attribute for the I/O directions are reversed



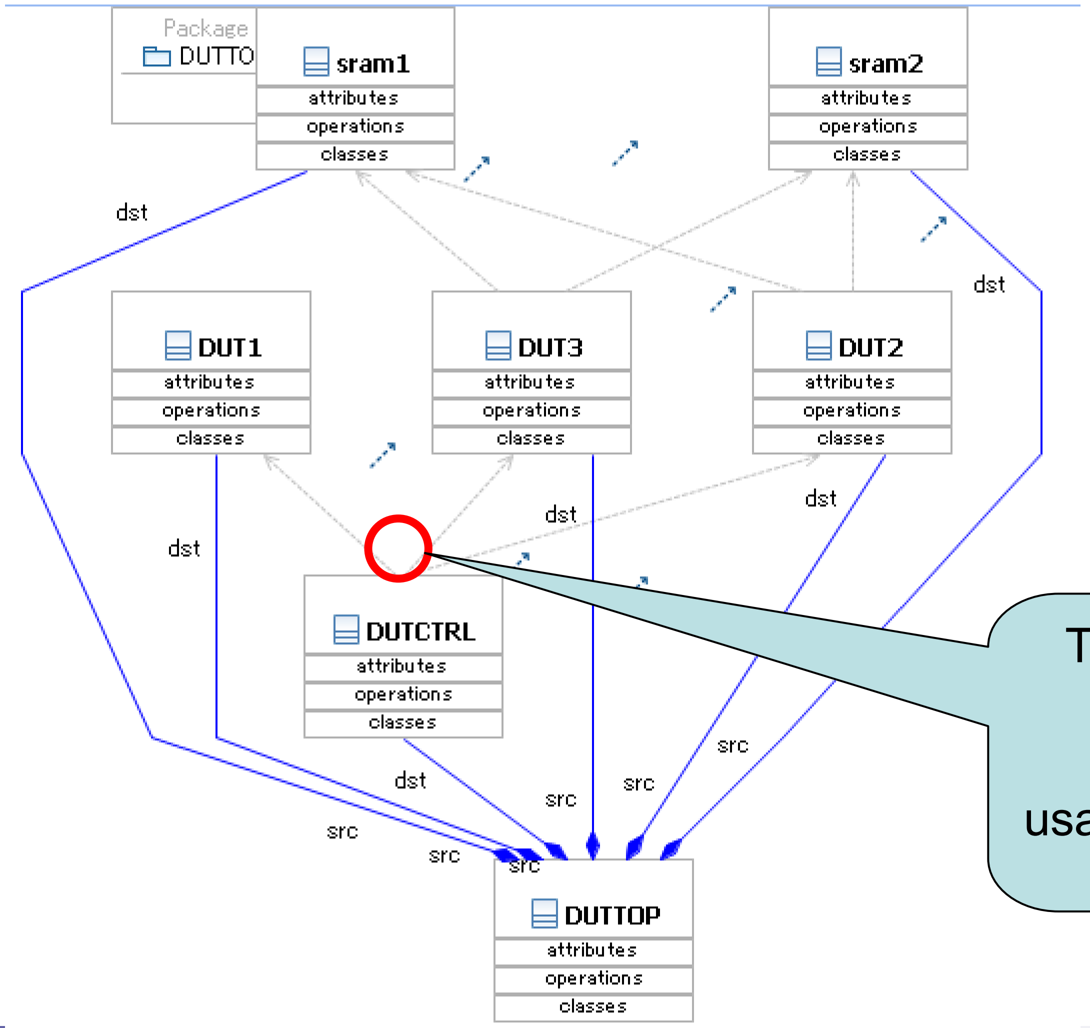
Static code checking of SystemC module

- Reports if there is any un-connected （floating） ports
  - Show only un-connected ports
  - Specify the user's choice with the property（STUB、0、1）



Multi-ports connection

- Example: broadcasting the control signal to each modules



The Modular interfaces should be usable adding to the ports

Discussion

- Minimum requirements are shown
- Realizablity?
- When?