High-Level Design Flow Guide

Renesas Electronics Corporation
Front-end Design Technology Development Department,
EDA & Design Methodology Division

2013/3/26 Rev. 2.1

RENESAS Group CONFIDENTIAL

Table of Contents

1. Introduction

Purposes of this guide

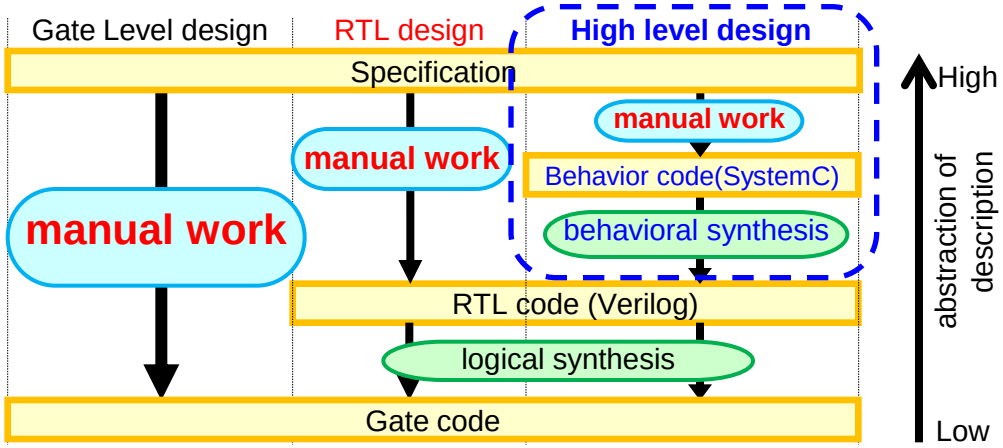
This guide explains the following items that you should understand before applying High-Level Design (HLD):

- HLD flow and what to do in each design phase
- SystemC description creation flow for HLD
- Tools used for HLD
- Documents to refer to for HLD

What is High-Level Design(HLD) ?

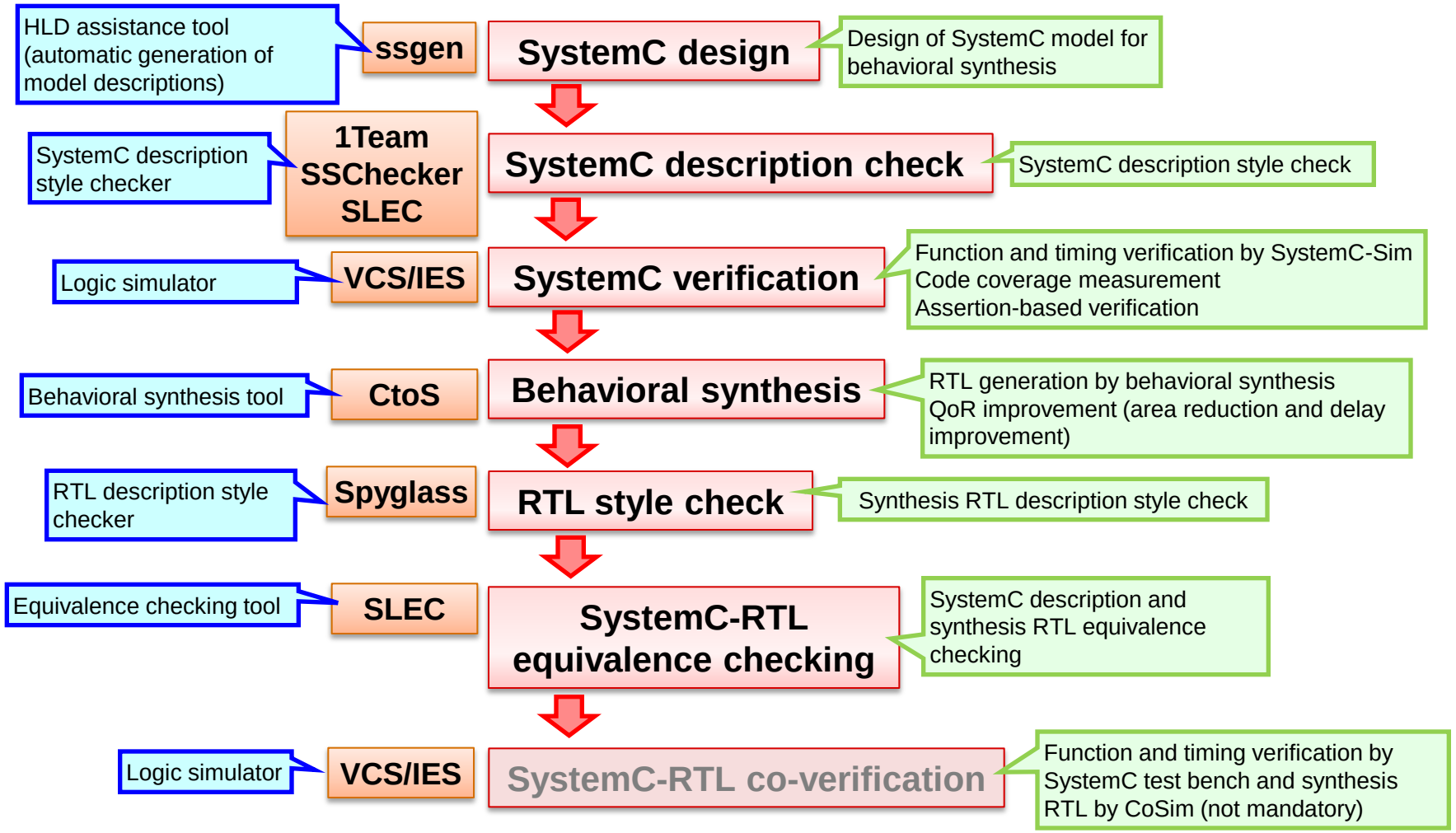
HLD is a design style in which the level of abstraction of hardware design is improved from RTL (register accuracy) descriptions, which are currently used as the mainstream, to SystemC behavioral descriptions and RTL descriptions are automatically generated by inputting appropriate constraints from SystemC behavioral descriptions using the behavioral synthesis tool (CtoS).
HLD has the following two main advantages:

- Improving design productivity: Increase in speed of behavior verification, reduction of the amount of design descriptions, and easier variation design
- Avoiding rework due to the specification: Behavior verification enabled at an early stage through the reduction of the amount of design descriptions



2. High-Level Design Flow Overview (1)

- This chapter explains the HLD flow, what to do in each design step, and tools used. The following shows a conceptual drawing of the entire flow.



2. High-Level Design Flow Overview (2)

SystemC Design

In this design phase, design the SystemC model to be input to the behavioral synthesis tool (CtoS).

- What to do
  You can use ssgen and common HLD Environment to automatically generate fixed SystemC model descriptions and test bench templates. For details of how to design a model, see Chapter3.
- Inputs
  - Hardware specifications (functions, interfaces, timing, etc.)
  - Algorithm C models (if any)
- Output
  - SystemC description
- Tool used
  - ssgen
  - Common HLD Environment

Related documents (See Chapter5)

- C++/SystemC Syntax Rules
- High-Level Design Tutorial SystemC Description for CtoS Edition
- SystemC Description Reference Manual for High-Level Design
- Collection of Sample SystemC Descriptions/Scripts
- High-Level Design SystemC coding style guide
- C++/SystemC coding Rule
- ssgen user's manual, ssgen Tutorial
- HLD Environment user's manual

2. High-Level Design Flow Overview (3)

SystemC Description Check (1)

In this design phase, use the SystemC description rule checker (1Team:System) and SSChecker (REL checker) to check statically if any generated SystemC description violates any behavioral synthesis constraint, may have the possibility of causing a bug, or will cause an error or problem during compilation or simulation.

- What to do
  Check the execution log files of 1Team:System and SSChecker and take appropriate action for the description for which a Syntax, Fatal, or Error message is output in the report file. The designer should determine whether to take action for the description for which a Warning message is output in the report file. A pseudo error message may be output to the report file due to 1Team tool defects. Check this type of error using the Checking known-bugs (Mis-detection) Script.
  You can easily execute 1Team:System by using Common HLD environment.
- Inputs
  - SystemC description
  - Rule set for behavioral synthesis (Renesas/Synth)
- Outputs
  - Execution log file (spyglass.log): Contains summary information at the end.
  - Analysis result file (moresimple.rpt): Detailed report on error/warning messages
  - SSChecker analysis result file: Detailed report on error/warning messages
- Tools used
  - 1Team:System
  - SSChecker
  - Checking known-bugs (Mis-detection) Script
  - Common HLD Environment

Related documents (See Chapter5)
- 1Team:System user's Manual
- Checker list of High-Level Design SystemC coding style guide
- Checking known-bugs (Mis-detection)Script User's Manual
- SSChecker user's manual
- HLD Environment user's manual

2. High-Level Design Flow Overview (4)

SystemC Description Check (2)

Use the equivalence checking tool (SLEC) to see if there is any uninitialized variable or access to the outside of the array that cannot completely be checked with static analysis using 1Team:System. This check can also be performed with "SystemC verification" or "SystemC-RTL equivalence checking", which is a later design stage, but it is recommended to perform this check in this design phase for early detection and rapid measures.

- What to do
  Check the SLEC execution log file. If you find Falsified, there may be an uninitialized variable or access to the outside of the array. In this case, check the relevant signal information and waveform data, determine the faulty location, and correct the SystemC description. If there is not Falsified, but Unresolved, the possible cause may be an SLEC defect.
- Inputs
  - SystemC description
  - SLEC execution script (.tcl)
- Outputs
  - SLEC execution log file (slec.log): Contains summary information of the check results at the end
  - Waveform data until the cycle in which a mismatch occurs (spec(impl).vcd): If there is a mismatch
- Tool used
  - SLEC

Related Documents (See Chapter5)
- Sequential Equivalence Checking Tool SLEC User's Manual
- SLEC Know-How

2. High-Level Design Flow Overview (5)

SystemC Description Check (3)

Use the Overflow Checker which is supplement utility of the behavioral synthesis tool (CtoS) to see if there is any bit overflow in calculation or assignment description that cannot be checked with static analysis using 1Team:System.

- What to do
  Check the CtoS execution log file. If you find WARNING message of the Overflow checker, please make code review to check whether there is overflow or not. If a problem of overflow exists, correct the SystemC description. Since there is a possibility of miss detection, you don't have to always modify the code.
- Note
  RTL style checker (HAL) with built-in IES is used for execution of Overflow Checker. It is necessary to read the environment setting file of IES first. Moreover, it is necessary to be synthesizable description.
- Inputs
  - SystemC description
  - CtoS execution script (.tcl)
- Outputs
  - CtoS execution log file (slec.log)
- Tool used
  - CtoS
  - Overflow Checker
  - Simulator (IES)

Related Documents (See Chapter5)
- Overflow Checker Utility User's Manual

2. High-Level Design Flow Overview (6)

SystemC Verification

In this design phase, simulate the operation of the SystemC model subject to behavioral synthesis, using a test bench to verify that there are no problems with the functions and timing. Also measure the code coverage to guarantee the coverage of verification. As required, embed the assertion-based checker (SVA: SystemVerilog Assertions) to verify that the operation is as specified.

- What to do

  Confirm items including that the simulation results match the expected results, that the code coverage is 100%, and that no assertion violations have occurred. If a bug is found, debug it using a debugger.

  You can easily execute simulation by using ssgen and Common HLD environment.
- Inputs
  - SystemC description
  - SVA (assertion check description)
  - Simulation execution script
- Outputs
  - Simulation execution results and waveform file
  - Coverage measurement results (analysis results: *.gcov)
- Used tools
  - Simulator (such as VCS-MX, IES, or gcc)
  - Debugger (such as gdb or Visual C++)
  - Coverage tool (gcov)
  - ssgen
  - dummyins, mcpp
  - Common HLD Environment

Related documents (See Chapter5)
- VCS-MX User's Manual
- NC-SC Tool Manual
- Collection of SystemC Code Debug Know-How
- Easily-Measurable SystemC Code Coverage Guide
- SVA Application Guide
- SVA Training
- ssgen user's manual, ssgen Tutorial
- HLD Environment user's manual

2. High-Level Design Flow Overview (7)

Behavioral Synthesis

In this design phase, input the created SystemC model to the behavioral synthesis tool (CtoS) to automatically generate an RTL.

- What to do
  The CtoS execution script can automatically be generated using ssgen. Confirm with the execution log file that no errors have occurred, and confirm that an RTL has been generated. A separate confirmation is necessary for determining that the RTL has been generated properly. See "SystemC-RTL Equivalence Checking", and "SystemC-RTL Co-simulation". Note that logic synthesis must be performed separately to confirm that the path delay in the RTL generated by CtoS meets the timing constraints and that the area (circuit scale) meets the desired size.
  You can easily execute behavioral synthesis by using ssgen and Common HLD environment.
- Inputs
  - SystemC description
  - Synthesis script (.tcl)
  - Cell library (.lib)
- Outputs
  - RTL(Verilog-HDL)file
  - CtoS execution log file (ctos.log)
- Tools Used
  - CtoS
  - ssgen
  - Common HLD Environment
  - Logic Synthesis tool (Design Compiler and RTL Compiler)

Related documents (See Chapter5)
- High-Level Design SystemC coding style guide
- CtoS FAQ
- Collection of High-Level Design TIPS
- Collection of CtoS Tips (provided by Cadence)
- CtoS Defect DB
- ssgen user's manual, ssgen Tutorial
- HLD Environment user's manual

2. High-Level Design Flow Overview (8)

RTL Style check

In this design phase, check the RTL descriptions generated by CtoS, using SpyGlass, to see if they violate the style rules of Renesas. Basically, no problems should be found, but descriptions violating the style rules may be generated due to CtoS defects, so be sure to perform the check.

- What to do
  Execute SpyGlass with the rule set for Renesas only and file for suppressing reports for HLD. Check the execution log file to see the report. If there is a problem, contact the Front-end Design Technology Development Department.
- Inputs
  - RTL(Verilog-HDL)file
  - Rule set for Renesas only (renesas2001)
  - File for suppressing reports for HLD (hls_design_waiver.txt)
  - Clock reset constraint file (clock_reset.txt)
- Outputs
  - Execution log file (spyglass.log) : Contains summary information at the end.
  - Analysis results (moresimple.rpt) : Detailed Fatal/Error/Warning messages
- Used tool
  - Spyglass

Related documents (See Chapter5)
- SpyGlass Renesas Policy User Guide
- RTL style check guide on High-Level Design

2. High-Level Design Flow Overview (9)

SystemC-RTL Equivalence Checking

In this design phase, verify that SystemC description is functionally equivalent to the RTL generated by CtoS, using the equivalence checking tool (SLEC).

- What to do
  Check the SLEC execution log file. If you find Falsified, there may be an uninitialized variable or access to the outside of the array or the RTL generated by CtoS may functionally differ (due to CtoS defects). In this case, check the relevant signal information and waveform data, determine the faulty location, and examine whether the problem can be worked around by correcting the SystemC description. If there is not Falsified, but Unresolved, the possible cause may be an SLEC defect. When the possible cause may be a CtoS/SLEC defect, contact the Front-end Design Technology Development Department.
- Inputs
  - SystemC Description
  - RTL(Verilog-HDL)file
  - SLEC Execution script(.tcl) :Can be generated from the SLEC xml file generated by CtoS, using the utility.
- Outputs
  - Execution log file (slec.log): Contains summary information of the check results at the end
  - (If there is a mismatch) waveform data (spec (impl).vcd) until the cycle in which a mismatch occurs
- Used tool
  - SLEC

Related Documents (See Chapter5)
- Sequential Equivalence Checking Tool SLEC User's Manual
- SLEC Know-How

2. High-Level Design Flow Overview (10)

SystemC-RTL Equivalence Checking (2)

- Note

  If the execution conditions for SystemC-RTL equivalence checking and the results are as below, it is necessary to perform RTL-RTL equivalence checking additionally to see if X propagation occurs.
  - A Full Proof match can be confirmed, but input constraints (such as fixed value inputs) are set for the verification.
  - Matches are confirmed with Bounded verification only, and matches are not confirmed with Full Proof match verification.

2. High-Level Design Flow Overview (11)

SystemC-RTL Co-Simulation

In this design phase, verify (using VCS-MX/IUS) the RTL generated by CtoS, using the test bench (SystemC) used in SystemC verification, to see if it has any problems in functions and timing.

- What to do

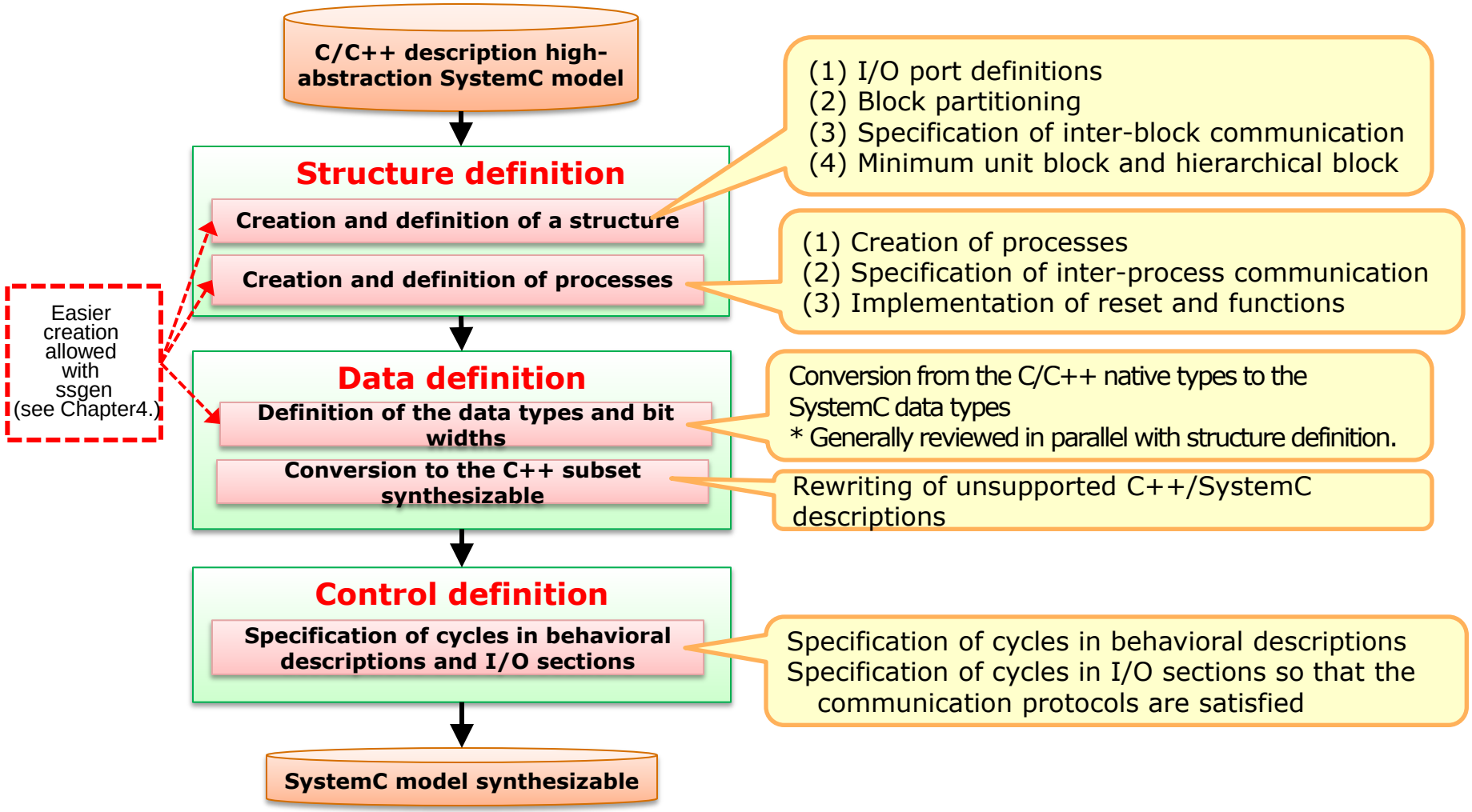
  Confirm that the simulation results match the expected values. This design phase is not mandatory, however. In either of the following cases, it is necessary to perform this phase:
  - The problem that SystemC-RTL equivalence checking takes a long time cannot be resolved, and the match cannot be confirmed.
  - When pipeline synthesis is used, the data output timing differs from that of SystemC. For this reason, confirm that the difference in timing between interfaces between modules does not cause any problem. It is also necessary to correct the SystemC test bench so that it considers the difference in timing.

  A simulation execution script can automatically be generated using ssgen.
- Inputs
  - SystemC description (Test bench)
  - RTL(Verilog-HDL)file
  - Simulation execution script
- Outputs
  - Simulation execution results and waveform file
- Tool Used
  - Simulator (VCS-MX and IES)
  - ssgen

Related documents (See Chapter5)
- VCS-MX User's Manual
- NC-SC Tool Manual
- ssgen user's manual, ssgen Tutorial

3. Flow for Creating SystemC Descriptions for HLD (1)

- Flow for Creating SystemC Descriptions for HLD
  The following shows a flow for creating SystemC descriptions synthesizable with CtoS and outlines what to do.



3. Flow for Creating SystemC Descriptions for HLD (2)

Structure definition

Creation and definition of a structure

(1) I/O port definitions
Determine the input port for reading data in a module and the output port for writing data from the module. A port is a communication resource and can be shared. According to the circuit requirements, define the number of ports and whether to set one port for each I/O or share a port among I/Os. Also define clock and reset ports.

(2) Block partitioning
Partition the HW structure into blocks based on the function unit. If the HW structure is refined, a block may be partitioned again into several blocks. Assign one SystemC module for each block.

(3) Specification of inter-process communication
Determine the inter-block communication method. Provide dedicated ports and insert a signal between them as a communication resource. Based on the blocks and communication between them, the architecture of the top level in the circuit is determined.

(4) Minimum unit block and hierarchical block
In the final architecture, the circuit consists of minimum unit blocks in which other blocks are not contained and hierarchical blocks which bundle minimum unit blocks.

3. Flow for Creating SystemC Descriptions for HLD (3)

Structure definition

Creation of processes

(1) Creation of processes

A SystemC module which is a minimum unit block can contain multiple SC_CTHREAD and SC_METHOD processes. A process is the minimum HW processing unit and operates in parallel.

(2) Specification of inter-process communication

Use a signal line for inter-process communication. Define the communication method so that only one process outputs data to the line at a time. (The multi-drive mode is prohibited.)

(3) Implementation of reset and functions

In an SC_CTHREAD process, implement a reset and a function. Write processing performed in the normal state in an endless loop. The processing written just before the endless loop is assumed to be a reset description. Insert cycles in each function section.

3. Flow for Creating SystemC Descriptions for HLD (4)

Data definition

- **Definition of the data types and bit widths**

  The behavioral synthesis tool supports automatic bit width optimization. When all descriptions are of the C/C++ native data types, however, large-scale optimization cannot be expected since there is few information on the bit widths. For this reason, it is recommended to explicitly specify the bit widths using SystemC data types (such as sc_uint) at least for ports and signals. If the bit widths are not optimized properly, it is necessary to specify bit widths for member variables and local variables.

- **Conversion to the C++ subset synthesizable**

  Since there are description constraints on the behavioral synthesis tool, the C++/SystemC syntax is not always available. It is necessary to impose a limitation on descriptions for a model so that only the description subsets synthesizable are used.

  - Examples of unsupported C/C++ descriptions
    Dynamic memory allocation (new, malloc), global variables, recursive functions, virtual functions, inheritance, and so on
  - Examples of unsupported SystemC descriptions
    SC_THREAD, sc_fifo, sc_inout, and so on

3. Flow for Creating SystemC Descriptions for HLD (5)

Control definition

- **Specification of cycles in behavioral descriptions**
  - When there is a loop description, it is necessary to specify the unrolling of the loop in the synthesis script or insert cycles with a wait statement in the loop description so that the loop is not executed in 0 cycle.
  - When allocating cycles to a behavioral description, consider cycle constraints to be satisfied by the behavior to be implemented and insert cycles with a wait statement so that each arithmetic unit is executed in different cycles whenever possible.
  - With the behavioral synthesis tool, arithmetic operations executed in different cycles are automatically shared even with different bit widths. Arithmetic operations executed in the same cycle, but according to different branches are also shared.
- **Specification of cycles I/O sections**

  In each I/O section for inter-module communication or memory access, it is necessary to insert cycles to define protocols.

  (1) Protocol for the start and end signals of a module subject to synthesis
  (2) Protocol for asynchronously input signals
  (3) Protocol for signals for inter-module communication
  (4) Protocol for accessing to system registers (exclusive control also considered)
  (5) Protocol for accessing to external memory

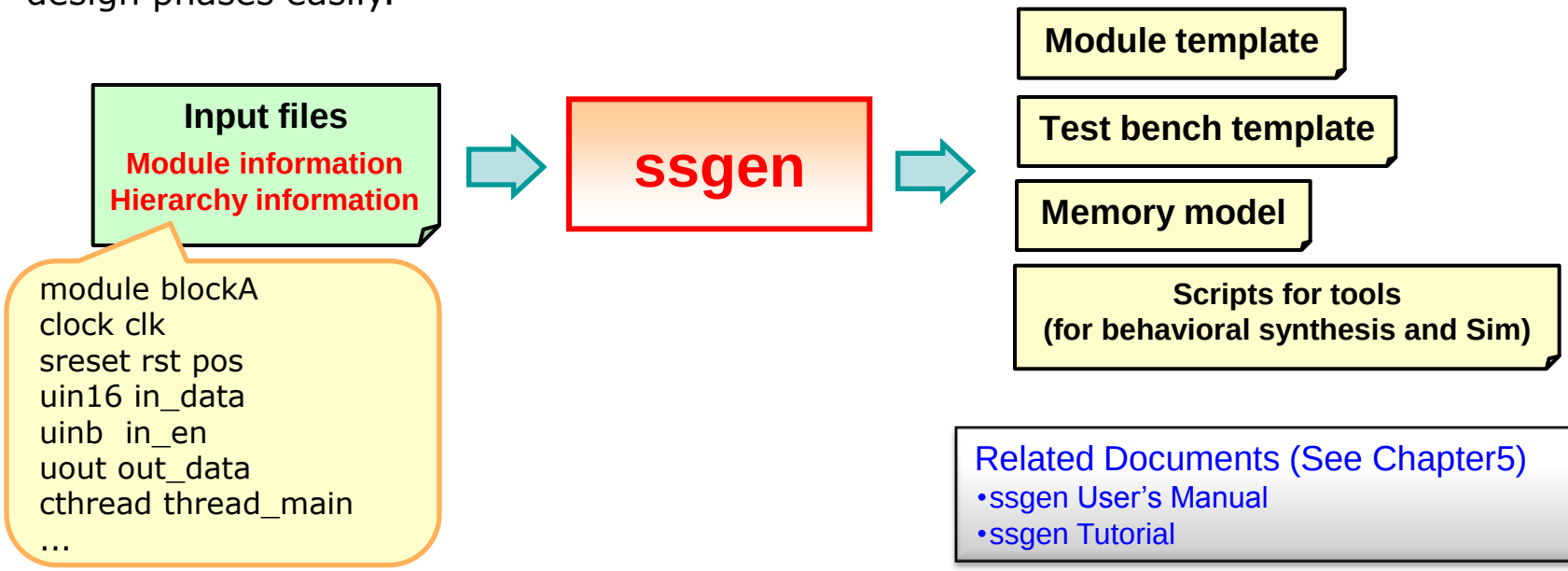
4. Tools Used in HLD (1)

- The following table lists the tools used in HLD.
  - The use of tools Nos. 8 to 12 is not mandatory in the HLD flow, but is recommended to enable the reduction of design and verification man-hours and improvement of design quality. (An overview is given in subsequent slides.)

No.	Tool name	Supplier	Function	Intended design phase
1	CtoS	Cadence	Behavioral synthesis tool	Behavioral synthesis
2	SLEC	Calypto	Equivalence checking tool	SystemC description check SystemC-RTL equivalence check
3	1Team:System	Atrenta	SystemC description style checker	SystemC description check
4	Spyglass	Atrenta	RTL description style checker	RTL style check
5	VCS-MX IES	Synopsys Cadence	Logic simulator	SystemC verification SystemC-RTL co-verification
6	gcov	Freeware	Code coverage tool	SystemC verification
7	SSChecker	REL tool	SystemC description style checker(Complementary tool for 1Team:System)	SystemC verification
8	ssgen	REL tool	HLD assistance tool	SystemC design SystemC verification Behavioral synthesis SystemC-RTL co-verification
9	dummyins	REL tool	Tool which automatically generates dummy descriptions for branch coverage	SystemC verification
10	mcpp	Freeware	Macro expansion tool	SystemC verification
11	Overflow Checker	Cadence	Bit overflow checker	SystemC description check
12	Common HLD environment	REL tool	Collection of scripts which enable executing some tools easily	Whole of design flow

4. Tools Used in HLD (2)

- **ssgen (Synthesizable SystemC code Generator)**
  ssgen is a HLD assistance tool which automatically generates fixed model descriptions for SystemC modules to be used as input to the behavioral synthesis tool and a tool execution script.
  - This tool automatically generates descriptions for module definitions, hierarchy definitions, port and signal definitions, port connections, member and member variable declarations for module classes, external memory accesses, port and signal VCD waveform dump operations, and others.
  - It can also generate test bench templates, memory models, simulation execution scripts (for OSCI-Sim, VCS-MX, and IES), and high-level synthesis scripts (for CtoS).
- You can use this tool to automatically generate 40% to 50% of SystemC descriptions (based on actual results) and prepare execution scripts for tools required in subsequent design phases easily.



Related Documents (See Chapter5)
- ssgen User's Manual
- ssgen Tutorial

4. Tools Used in HLD (3)

- **dummyins, mcpp**
  dummyins is a tool which automatically inserts dummy 'else' clauses and new lines to enable the branch coverage of SystemC descriptions to be measured by using the line coverage tool (gcov).
  - Since no 'else' clause cannot be inserted into any macro definition section, it is recommended to apply dummyins to the descriptions to which mcpp (free macro expansion tool) has been applied.

Input file (sample.cpp)

```
int func(int arg, int arg2) {
    int rtn = 0;
    if (arg == 0) {
        rtn = arg2;
    }
    else if (arg < 10) {
        rtn = arg << 1;
        if (arg2 == 0) {
            rtn += 1;
        }
        else if (arg2 < 10) rtn -= 1;
    }

    if (rtn == arg2) rtn = 1; else rtn <<= 1;
    return rtn;
}
```

dummyins

Output file (sample_ins.cpp)

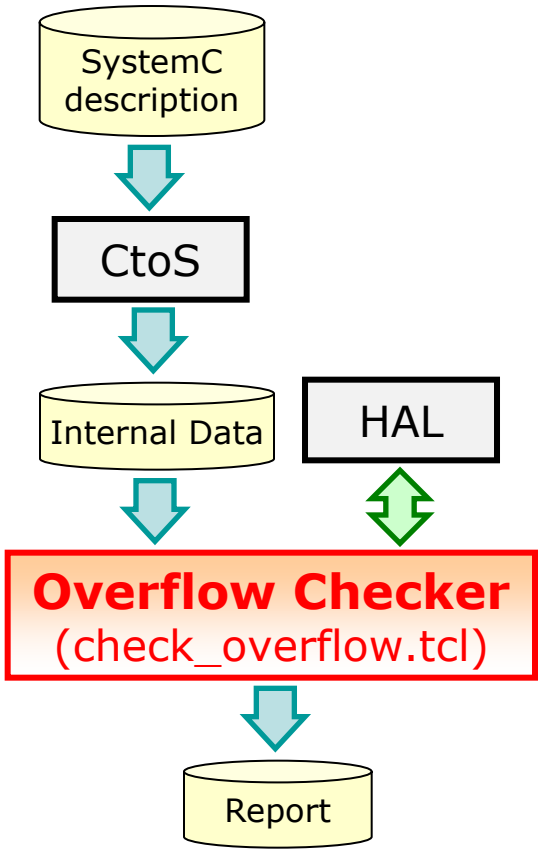
```
int func(int arg, int arg2) {
    int rtn = 0;
    if (arg == 0) {
        rtn = arg2;
    }
    else if (arg < 10) {
        rtn = arg << 1;
        if (arg2 == 0) {
            rtn += 1;
        }
        else if (arg2 < 10)
            rtn -= 1;
        else int dummy_for_gcov_implicit_branch = 0;
    }
    else int dummy_for_gcov_implicit_branch = 0;

    if (rtn == arg2) // [insert indent]
        rtn = 1;
    else rtn <<= 1;
    return rtn;
}
```

Related document (See Chapter5)
- Easily-Measurable SystemC Code Coverage Guide

4. Tools Used in HLD (4)

Overflow Checker

Overflow Checker is a tool which automatically detects bit overflow in calculation or assignment description that cannot be checked with static analysis using 1Team:System.
It performs in combination with the internal data (RTL simulation model) which CtoS generates, and RTL style checker HAL with built-in IES.



Example of report

```
### check_overflow
### Version: ...
hal  -check POIASG -check SHFTOF -check UEASTR -level 2 -NOCHECK
ALL_TOOL
-NOHALSYNTH -NOHALSTRUCT  ./model/post_build/test_post_build.v -logfile
hal_overflow.log
++++++++++++++++++++++++++++++++++++++++
WARNING: The result of addition operation may lead to a potential overflow.
Related Verilog Sim model:
./model/post_build/test_post_build.v:43
        add_ln7 = read_test_a_ln7 + read_test_b_ln7;
Related SystemC code and the operations:
test.cpp:7:26
      o.write(a.read() + b.read());
++++++++++++++++++++++++++++++++++++++++
```

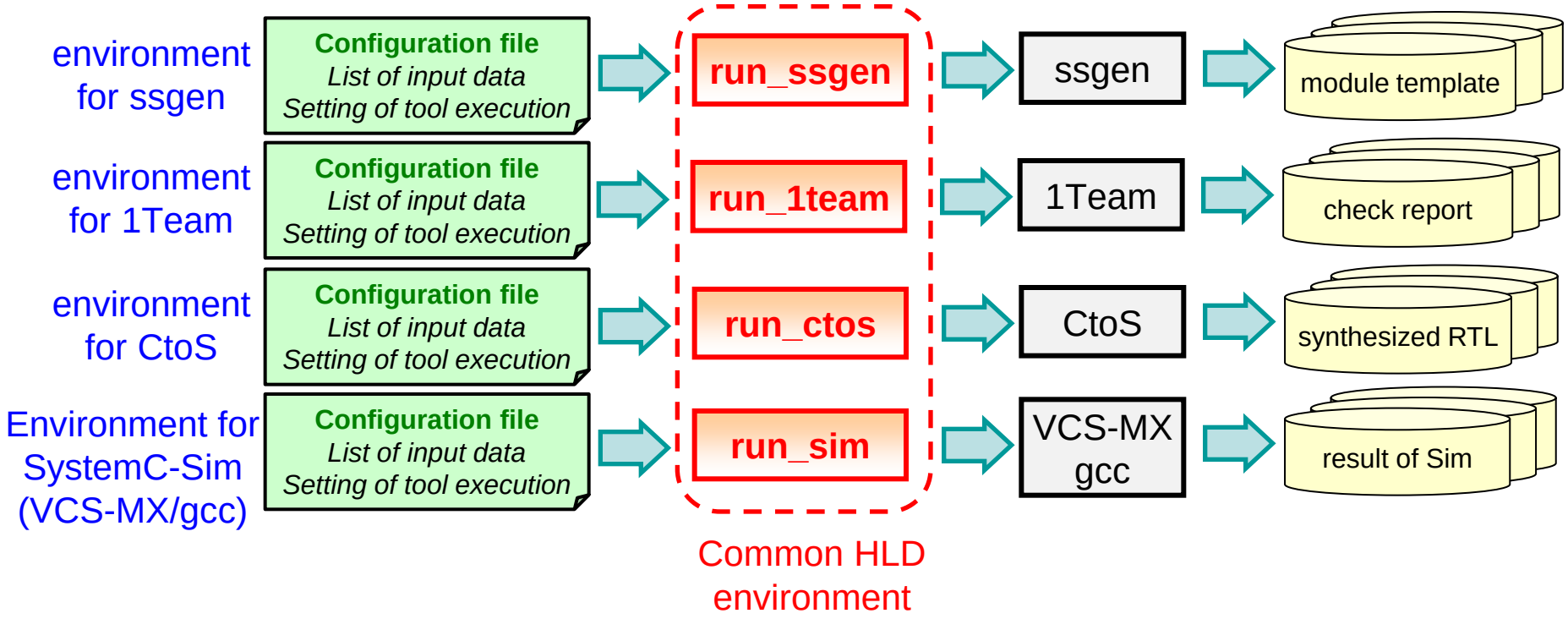
Output a WARNING to a description '8bit = 8bit + 8bit'

Related Documents (See Chapter5)
- Overflow Checker Utility User's Manual

4. Tools Used in HLD (5)

Common HLD environment

Common HLD environment is a collection of scripts for making tool execution easier in each design phase. Tools can be easily executed by an unified method and it can support an arbitrary design composition (directory structure of design data).
Now, the following four environments are supported.



Related documents (See Chapter5)
- HLD Environment user's manual

4. Tools Used in HLD (6)

- Vendor-supplied manual storage locations
  - All these manuals are stored in REL-design-environment/common/appl/.
  - For the paths listed in the table below, read "XXX" as the tool version

No.	Tool name	Supplier	Path
1	CtoS	Cadence	/common/appl/Cadence/cts/XXX/tools/ctos/doc/
2	SLEC	Calypto	/common/appl/Calypto/slec/XXX/slec/doc/
3	1Team:System	Atrenta	/common/appl/Atrenta/1teamsystem/XXX/SPYGLASS_HOME/doc
4	Spyglass	Atrenta	/common/appl/Atrenta/Spyglass/XXX/SPYGLASS_HOME/doc
5	VCS-MX	Synopsys	/common/appl/Synopsys/vcs_mx/XXX/doc/UserGuide/pdf
6	IES	Cadence	/common/appl/Cadence/incisive/XXX/IUSXXX/doc/

- REL tool and utility installation locations

No.	Tool name	Path
1	SSChecker	/common/appl/Renesas/SystemC/utility/SSChecker/SSChecker.pl
2	ssgen	/common/appl/Renesas/SystemC/utility/ssgen/ssgen.pl
3	dummyins	/common/appl/Renesas/SystemC/utility/coverage/dummyins.x
4	mcpp	/common/appl/Renesas/SystemC/utility/coverage/mcpp
5	Overflow Checker	/common/appl/Renesas/SystemC/utility/ctos/check_overflow.tcl

5. The documents related to HLD (1)

- The documents related to HLD are classified into the following six categories.

No.	Category	Description
1	Introduction	Documents for beginners in HLD Overview of HLD, tutorials, and others
2	Description style	Documents which describe SystemC description styles for HLD. Description style guides, reference manuals, and coding rules
3	Tool	Documents which describe how to use tools used for HLD CtoS, SLEC, 1Team:System, SSChecker, ECO flow, ssgen, and others
4	Know-how	Documents which describe various types of know-how for HLD Collection of TIPS, collection of Q&A, QoR improvement know-how, debug know-how, and others
5	Common	Documents available not only for HLD, but also for system level design and RTL design Methods for using VCS, IES, Spyglass, SVA, and others
6	Supplied by Cadence	CtoS-related documents provided by Cadence, CtoS supplier

  - All these documents are listed in "REL EDA Tools Information" in the REL intranet.
  - In subsequent slides, an overview of and the location in "REL EDA Tools Information" of each document are described for each category.

5. The documents related to HLD (2)

- 1. Introduction

Document name	Description	Location
High-Level Design Overview (Effects, Flows, and Application Examples) No English version	Document which introduces the advantages of HLD, design flow (overview), and application examples	【System Level Design Guide】 【Document】
C++/SystemC Syntax Rules No English version	Document which helps C++/SystemC beginners learn the basic syntax	【System Level Design Guide】 【Manual】
High-Level Design Tutorial SystemC Description for CtoS Edition Japanese version exists	Document which contains the basic SystemC description know-how for creating SystemC descriptions for CtoS	【System Level Design Guide】 【Document】
High-Level Design Tutorial Design Flow Edition Japanese version exists	Tutorial for learning the standard HLD flow by using a SystemC sample description to verify a SystemC description, automatically generate an RTL with high-level synthesis, and verify the generated RTL	【System Level Design Guide】 【Document】
High-Level Design Tutorial Memory Synthesis Flow Edition No English version	Tutorial for learning the HLD flow if performing memory synthesis, by using a SystemC sample description to synthesize memory from a SystemC array and verify an RTL containing memory	【System Level Design Guide】 【Document】
1Team:System Introduction and Tutorial English version only	Tutorial for learning the basic uses for 1Team:System	【1Team:System】 【Document】
ssgen Tutorial Japanese version exists	Tutorial for learning the uses for HLD assistance tool ssgen using a simple sample design	【System Level Design Guide】 【Document】

5. The documents related to HLD (3)

- 2. Description style

Document name	Description	Location
High-Level Design SystemC coding style guide Japanese version exists	Description style guide for creating a SystemC model for HLD	【System Level Design Guide】 【Manual】
Checker list of High-Level Design SystemC coding style guide Common to Japanese and English versions	List indicating how checkers (such as 1Team:System) support each item in "High-Level Design SystemC coding style guide"	【System Level Design Guide】 【Manual】
SystemC Coding Reference Manual for High Level Design using a Behavioral Synthesis Tool CtoS Japanese version exists	Guide which describes structural design techniques, how to create SystemC descriptions, and SystemC description subsets synthesizable for designing a SystemC model for HLD	【C2Silicon】 【Manual】
C++/SystemC Coding Rule Japanese version exists	Guide which describes basic coding rules for designing a SystemC model	【System Level Design Guide】 【Manual】
Collection of Sample SystemC Descriptions/Scripts No English version	Collection of sample SystemC descriptions and scripts for HLD	【System Level Design Guide】 【Document】

5. The documents related to HLD (4)

- 3. Tools(1/2)

Document name	Description	Location
Sequential Logic Equivalence Checker (SLEC) User Manual Japanese version exists	Manual which explains the standard use for applying high-level equivalence checking tool SLEC as well as limitations	【SLEC】 【Manual】
C/C++/SystemC Code Checker 1Team:System User's Manual Japanese version exists	Manual which explains the execution procedures for efficiently applying C/C++/SystemC description code checker 1Team:System to modeling design	【1Team:System】 【Manual】
Collection of 1Team:System Rules No English version	Collection of 1Team:System check rules (not cover all rules.)	【1Team:System】 【Document】
Checking known-bugs (Mis-detection)Script User's Manual No English version	Manual which describes how to use the script for extracting sections possibly corresponding to known false detection defects from the 1Team:System execution log file	【1Team:System】 【Manual】
SSChecker user's manual English version only	Guide which explains the uses for the complementary tool for checking problems with descriptions 1Team:System cannot check, SSChecker	【System Level Design Guide】 【Document】
RTL style check guide on High Level Design Japanese version exists	Guide which contains the notes on and procedures for performing the style check on RTL descriptions generated by CtoS, by using SpyGlass	【System Level Design Guide】 【Manual】

5. The documents related to HLD (5)

- 3. Tools(2/E)

Document name	Description	Location
ECO Flow Guide No English version	Guide which contains the general uses for the CtoS functions, incremental synthesis; and ECO design using source links, as well as notes and know-how	【System Level Design Guide】【Document】
High-Level design supporting tool ssgen user's manual Japanese version exists	Manual which contains the functions of and uses for the HLD assistance tool, ssgen	【System Level Design Guide】【Manual】
Guide to Simplifying SystemC Code Coverage Measurement Japanese version exists	Guide which contains the procedures for using the tool (dummyins) enabling the branch coverage to be measured when obtaining the code coverage of SystemC description by using gcov	【System Level Design Guide】【Document】
AMBA bus protocol verification methods for HLD Japanese version exists	Document which contains the procedure for performing bus protocol check and random generation of bus transaction by AMBA Verification IP of Synopsys on high-level design and verification	【System Level Design Guide】【Document】
Overflow Checker Utility User's Manual English version only	Manual which contains the procedures for using the supplement utility of CtoS enabling the detection of problems of bit overflow in calculation or assignment description that cannot be checked with static analysis using 1Team:System	【System Level Design Guide】【Document】
HLD Environment user's manual English version only	Guide which contains the procedures for using common HLD environment making tool execution (ssgen/1Team/CtoS/Sim) easily	【System Level Design Guide】【Document】

5. The documents related to HLD (6)

- 4. Know-how

Document name	Description	Location
Collection of High-Level Design TIPS No English version	Document which contains various types of know-how for HLD	【System Level Design Guide】 【Document】
Collection of High-Level Design Q&A No English version	Document which contains questions and answers collected during HLD application support	【System Level Design Guide】 【Document】
SLEC Know-How No English version	Document which contains know-how for equivalence checking tool SLEC	【SLEC】 【Document】
SystemC Code Change Examples for Area Reduction No English version	Document which contains know-how for area reduction (Issued in 2009. Some information may be outdated.)	【System Level Design Guide】 【Document】
SystemC Code Debug Know-How Japanese version exists	Document which contains the uses and know-how for the SystemC code debugger	【System Level Design Guide】 【Document】
CtoS FAQ No English version	Collection of CtoS FAQ for improving QoR of synthesized RTL	【C2Silicon】 【FAQ/Know-how】

5. The documents related to HLD (7)

- 5. Common

Document name	Description	Location
VCS-MX User's Manual No English version	Manual which explains how to use Synopsys logic simulator VCS-MX for verifying SystemC descriptions	【VCS [SystemC]】 【Manual】
NC-SC Tool Manual No English version	Manual which explains how to use Cadence logic simulator IES for verifying SystemC descriptions	【IES[SystemC]】 【Manual】
RTL description code checker SpyGlass Renesas Policy User Guide Japanese version exists	Manual which contains the uses for Atrenta RTL description code checker SpyGlass	【Spyglass】 【Manual】
SVA Application Guide No English version	Document which contains the uses for SVA (SystemVerilog Assertions)	【Function Verification Guide】 【Document】
SVA Training No English version	Training document for SVA (SystemVerilog Assertions)	【Function Verification Guide】 【Document】

5. The documents related to HLD (8)

- 6. provided by Cadence

Document name	Description	Location
How to improve area in CtoS English version only	Document which contains know-how (tool uses, analysis methods, and others) for reducing the area (circuit scale) in the RTL to be generated by using CtoS	【CtoS】 【Document】
How to improve timing in CtoS English version only	Document which contains know-how (tool uses, analysis methods, and others) for reducing the delay (path delay) in the RTL to be generated by using CtoS	【CtoS】 【Document】
CtoS Log file analysis English version only	Document which explains how to read the log report files output by CtoS	【CtoS】 【Document】
Resource Sharing in CtoS English version only	Document which explains resource sharing in CtoS	【CtoS】 【Document】
How to Save Execution Scripts Using GUI No English version	Document which explains how to save the command execution history using the CtoS-GUI	【CtoS】 【Document】
How to Work Around Memory Scheduling Errors No English version	Document which explains how to work around synchronous memory access scheduling problems (errors) in CtoS	【CtoS】 【Document】
How to Work Around Memory Scheduling Errors 2 No English version	Document which explains procedures for resolving problems when a scheduling error occurs according to array index dependency in the implementation of an array as memory by using CtoS	【CtoS】 【Document】
How to Allocate FPGA Block RAM No English version	Document which explains how to map arrays in memory in the CtoS FPGA mode	【CtoS】 【Document】

5. The documents related to HLD (9)

- 6. provided by Cadence

Document name	Description	Location
How to Set RC External Scripts No English version	Document which explains how to input RC options and commands with scripts to Encounter RTL Compiler (RC) executed for creating operations and other resources by using CtoS	【CtoS】 【Document】
How to Set IO External Delay No English version	Document which explains how to set external delay values for I/O ports during synthesis with CtoS	【CtoS】 【Document】
How to Output RTL for Hierarchical Synthesis No English version	Document which explains how to output the RTL for hierarchical design by using CtoS	【CtoS】 【Document】
How to Assign Desired Operation to Specified Pipeline Stage No English version	Document which explains how to assign desired operation to the specified pipeline stage by using CtoS	【CtoS】 【Document】
Loop Unroll Techniques No English version	Document which explains techniques for unrolling loops by using CtoS	【CtoS】 【Document】
How to Dump Module Members and I/Os to Waveforms with IES No English version	Document which explains how to dump module member variables and ports for which no name is defined, with correct names, on the IES waveform viewer (SimVision)	【CtoS】 【Document】
How to Execute FPGA Synthesis No English version	Document which explains how to execute RTL synthesis for FPGA by using CtoS	【CtoS】 【Document】
How to Set Automatic Pipeline No English version	Document which explains the procedure for setting automatic pipeline using the CtoS GUI	【CtoS】 【Document】
How to Avoid Function Name Collisions in Hierarchical Synthesis No English version	Document which explains how to avoid module name collisions caused with non-inline functions by using CtoS	【CtoS】 【Document】

5. The documents related to HLD (10)

- 6. provided by Cadence

Document name	Description	Location
How to Use SystemC Assertion No English version	Document which explains how to use SystemC PSL assertion by using IES	【CtoS】 【Document】
How to Measure Coverage for SystemC No English version	Document which explains how to measure the code coverage for SystemC by using gcov	【CtoS】 【Document】
How to Synthesize and Simulate External Memory No English version	Document which explains how to map an array into a RAM instance outside the module and perform simulation by using CtoS	【CtoS】 【Document】
ECO Flow with CtoS Incremental Synthesis and Conformal ECO No English version	Document which explains the ECO flow using the CtoS incremental synthesis function and Conformal ECO	【CtoS】 【Document】
How to Cancel Resource Sharing Manually No English version	Document which explains the method for canceling CtoS automatic resource sharing and the uses for the method	【CtoS】 【Document】
How to Debug SystemC by Using IES 9.2 No English version	Document which explains how to debug defects in SystemC descriptions by using the IES SystemC debug function, with actual examples	【CtoS】 【Document】
Array Coding Style and Notes No English version	Document which explains action to be taken when an array declared with Micro Architecture is not displayed on the CtoS GUI	【CtoS】 【Document】
Verilog-2001 Descriptions Used for Model Generation No English version	Document which explains the Verilog-2001 description style generated by CtoS	【CtoS】 【Document】
Multi-Drive for sc_signal No English version	Document which describes limitations and synthesis results in the multi-drive signal mode for CtoS	【CtoS】 【Document】

5. The documents related to HLD (11)

- 6. provided by Cadence

Document name	Description	Location
Insertion of User-Defined Comment in RTL Header No English version	Document which explains the script for adding a user-defined comment to the header of an RTL generated by CtoS	【CtoS】 【Document】
How to Measure Power in Netlist by RTL Logic Synthesis No English version	Document which explains how to measure power consumption of an RTL generated by CtoS by using RTL Compiler and TCF file generated by IES	【CtoS】 【Document】
How to Display I/O Ports of User-Defined Class in Waveform No English version	Document which explains how to recognize the class structure of a user-defined class used for I/O ports with an RTL generated by CtoS and display the I/O ports in waveform by IES	【CtoS】 【Document】
How to Improve the SystemC Compilation Speed in IES No English version	Document which explains the method for integrating multiple SystemC files into a single one, distributed compilation using LSF, and the use of the speed-up option as methods for improving the SystemC compilation speed in IES	【CtoS】 【Document】
How to Encrypt Library Files Used by RTL Compiler No English version	Document which explains how to encrypt Liberty library files to be used for synthesis with CtoS to conceal them and how to use them with RTL Compiler	【CtoS】 【Document】
Use of Resource Sharing in CtoS No English version	Document which explains the know-how related to resource sharing, which is the CtoS area reduction function	【CtoS】 【Document】
How to Use irc_run No English version	Document which explains how to use irc_run, which activates RTL Compiler and Conformal-LEC (equivalence checking) tools from CtoS	【CtoS】 【Document】
Function Inline Setting in CtoS No English version	Document which explains how to determine whether to perform function inline in CtoS, advantages and disadvantages of setting function inline, and others	【CtoS】 【Document】
How to Set PLE No English version	Document which explains how to use PLE (Physical Layout Estimator) instead of the library wire load model, for synthesis with CtoS	【CtoS】 【Document】

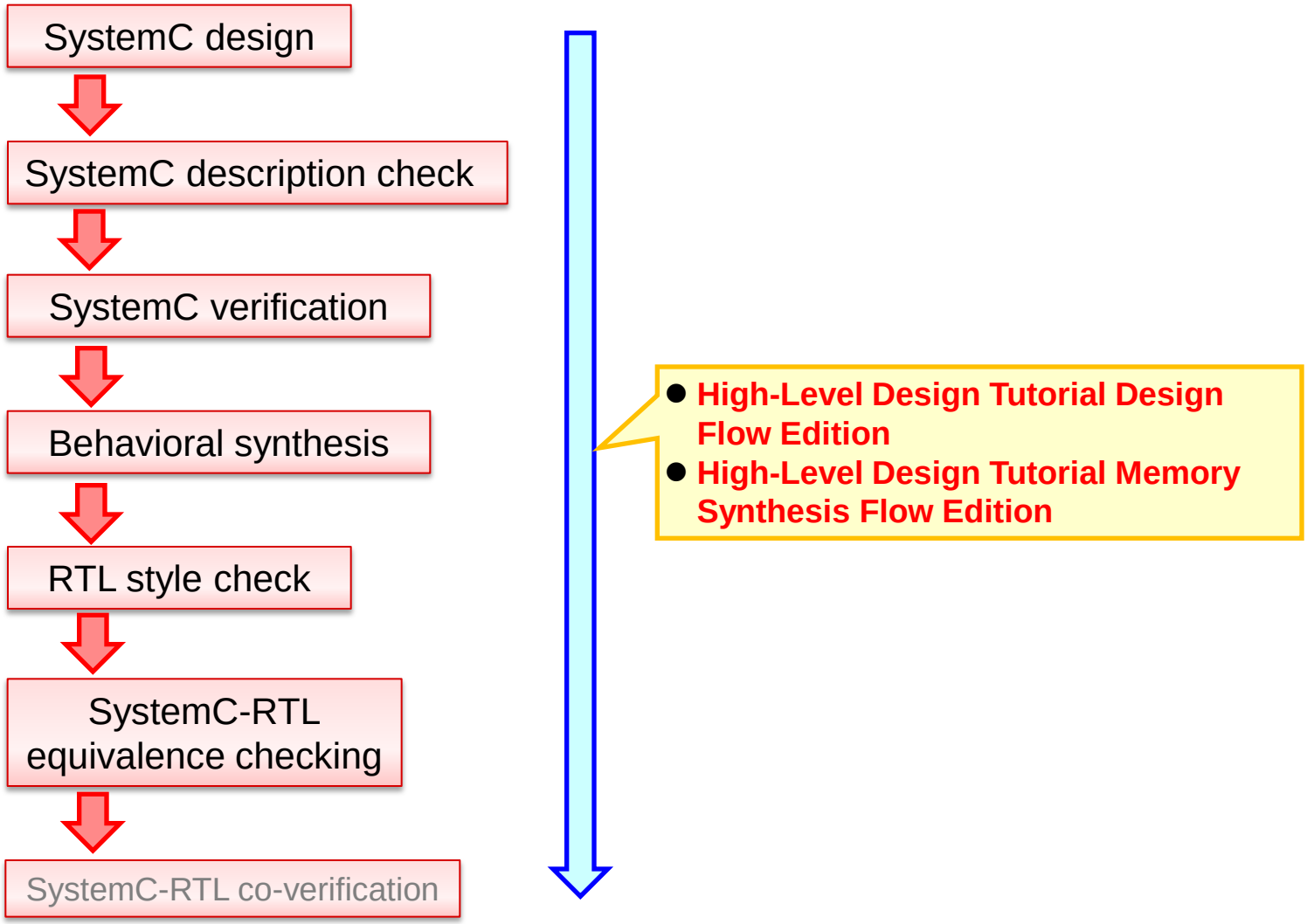
5. The documents related to HLD (12)

- 6. provided by Cadence

Document name	Description	Location
How to Allocate Multi-Bit Multiplier to FPGA DSP Macros No English version	Document which describes the procedure for dividing a multi-bit multiplier exceeding the bit width and mapping it in FPGA DSP macros by using CtoS	【CtoS】 【Document】
How to Use SystemC Wrapper for CtoS No English version	Document which explains how to generate and use a wrapper file for connecting an RTL generated by CtoS to a SystemC test bench	【CtoS】 【Document】
How to Work Around LEC Abort Due to Parallel Case and Case Default Change No English version	Document which describes how to handle problems (including aborts) with equivalence checking by Conformal LEC for an RTL generated by CtoS	【CtoS】 【Document】
Manual Scheduling No English version	Document which describes how to efficiently perform manual scheduling by using CtoS schedule -useBirthday and options	【CtoS】 【Document】
How to Change Resource Allocation No English version	Document which explains how to change the CtoS resource allocation results manually	【CtoS】 【Document】
Scripts for Generating XML and Verilog Wrapper for RTL IP No English version	Document which explains the procedure for binging SystemC functions to RTL IP (how to use scripts) by using CtoS	【CtoS】 【Document】
Scripts for Generating Vendor RAM Verilog Wrapper and Simulation Model from XML File No English version	Document which explains the procedure for generating Vendor RAM Verilog wrapper and simulation model based on an XML file containing RAM information (how to use scripts) by using CtoS	【CtoS】 【Document】
Script for Making XML File Compatible with CtoS 11.10 or Later No English version	Document which explains how to make an XML definition file (Vendor RAM and RTL IP definitions) in a former format compatible with the latest format by using CtoS	【CtoS】 【Document】

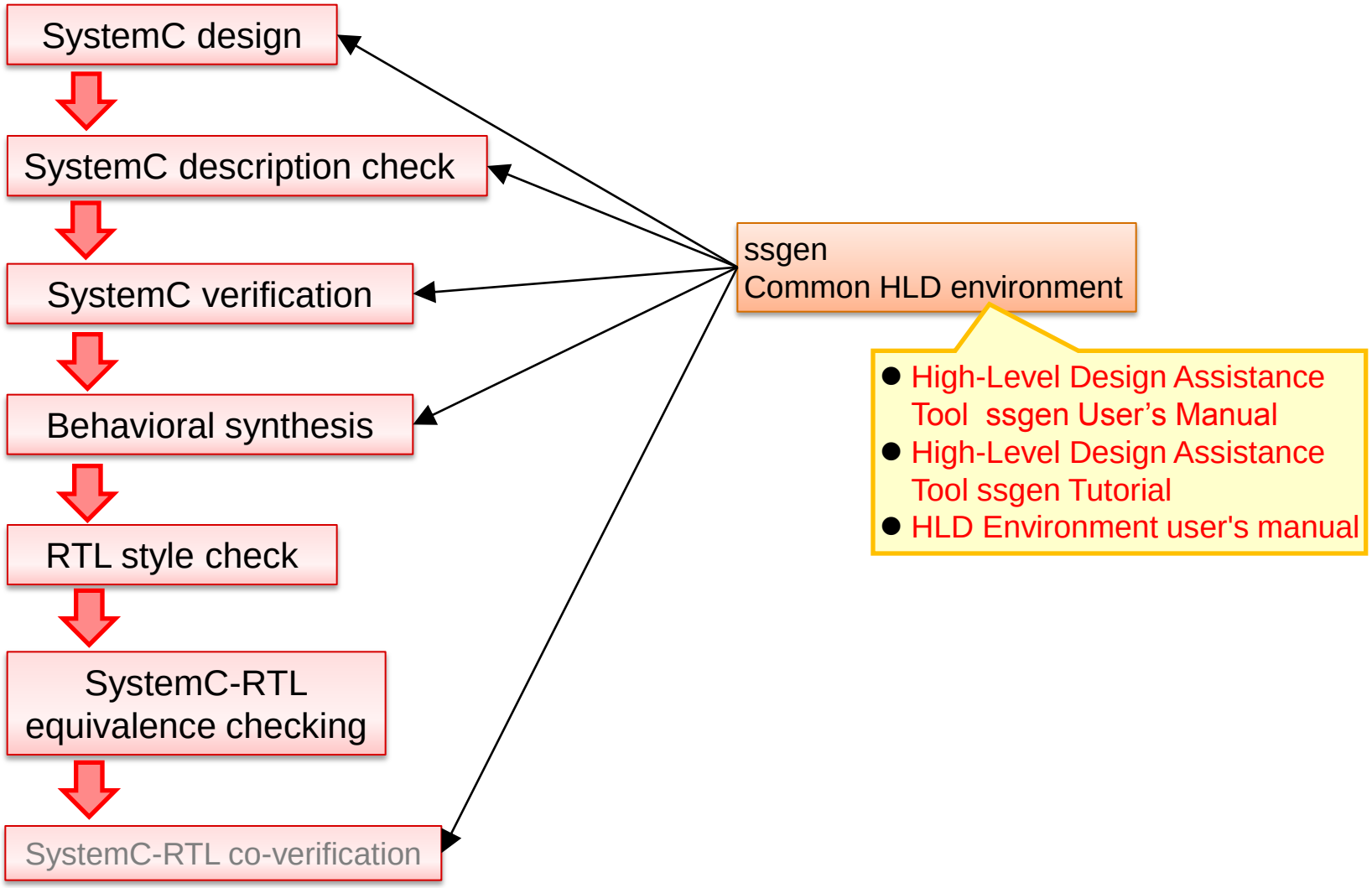
6. How to Refer to Documents Related to HLD (1)

- Assumed purpose of use: To experience the entire HLD flow and understand design techniques quickly



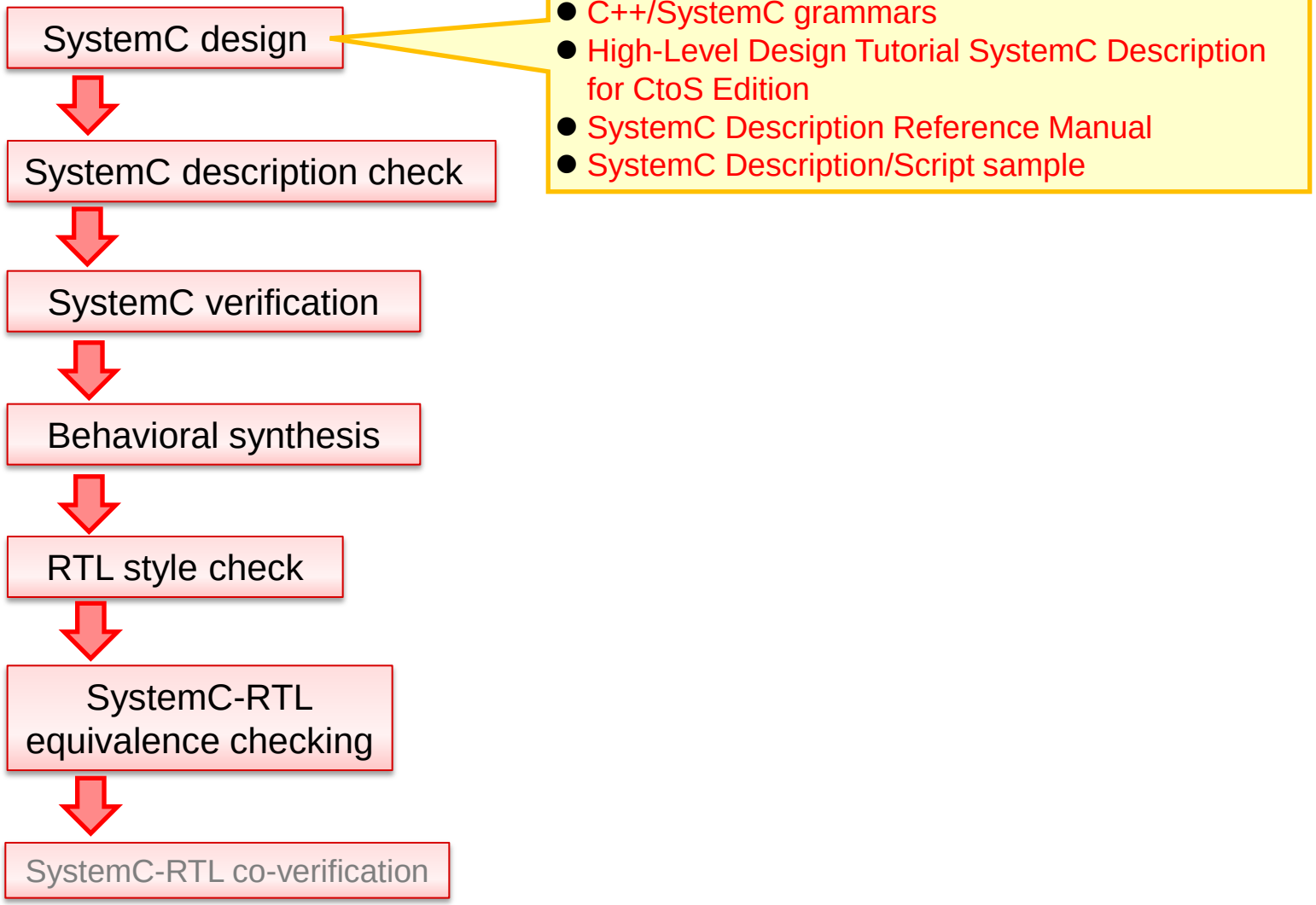
6. How to Refer to Documents Related to HLD (2)

- Assumed purpose of use : To speed up the HLD flow



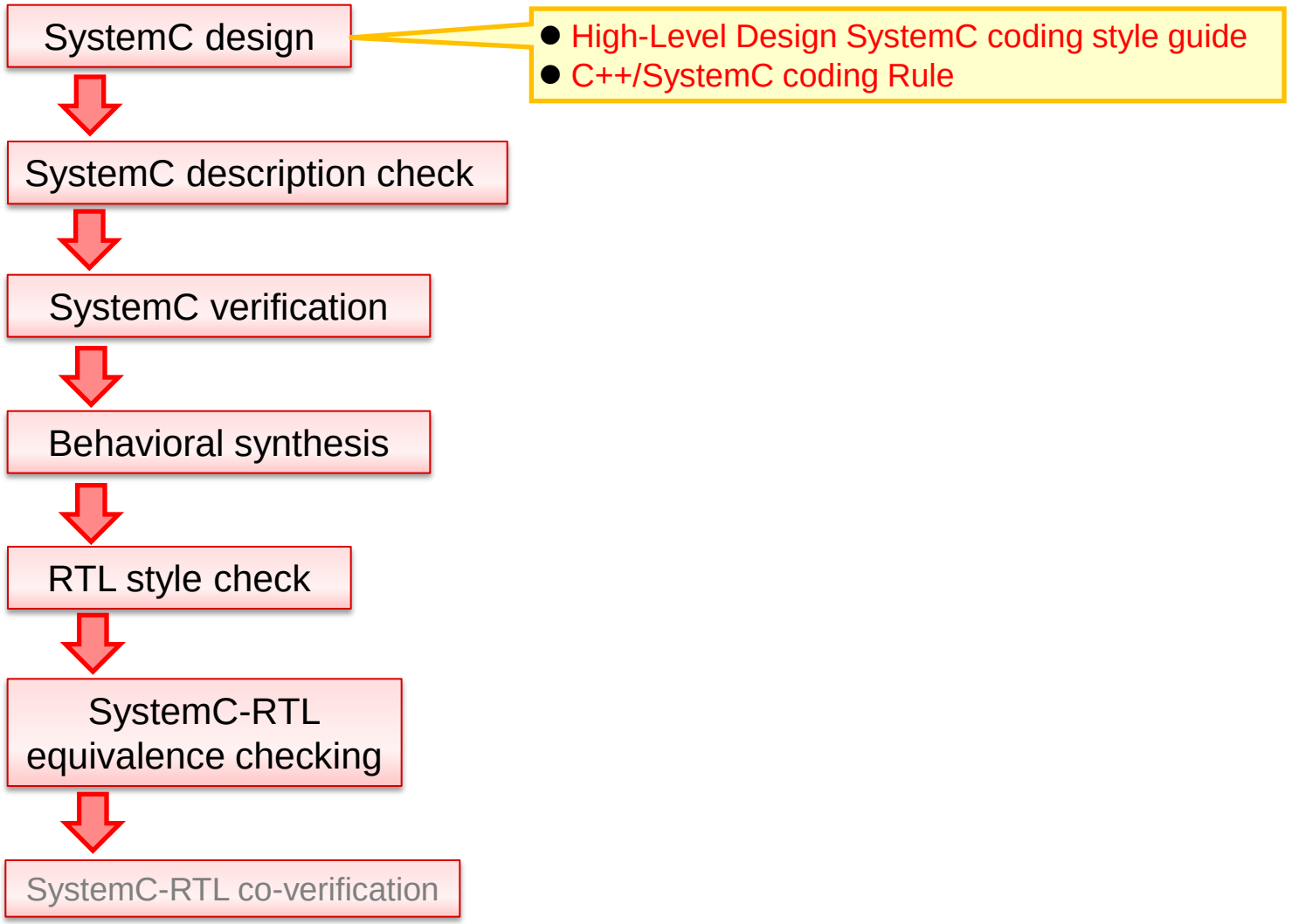
6. How to Refer to Documents Related to HLD (3)

- Assumed purpose of use : To learn how to create a model from basics (for SystemC beginners)



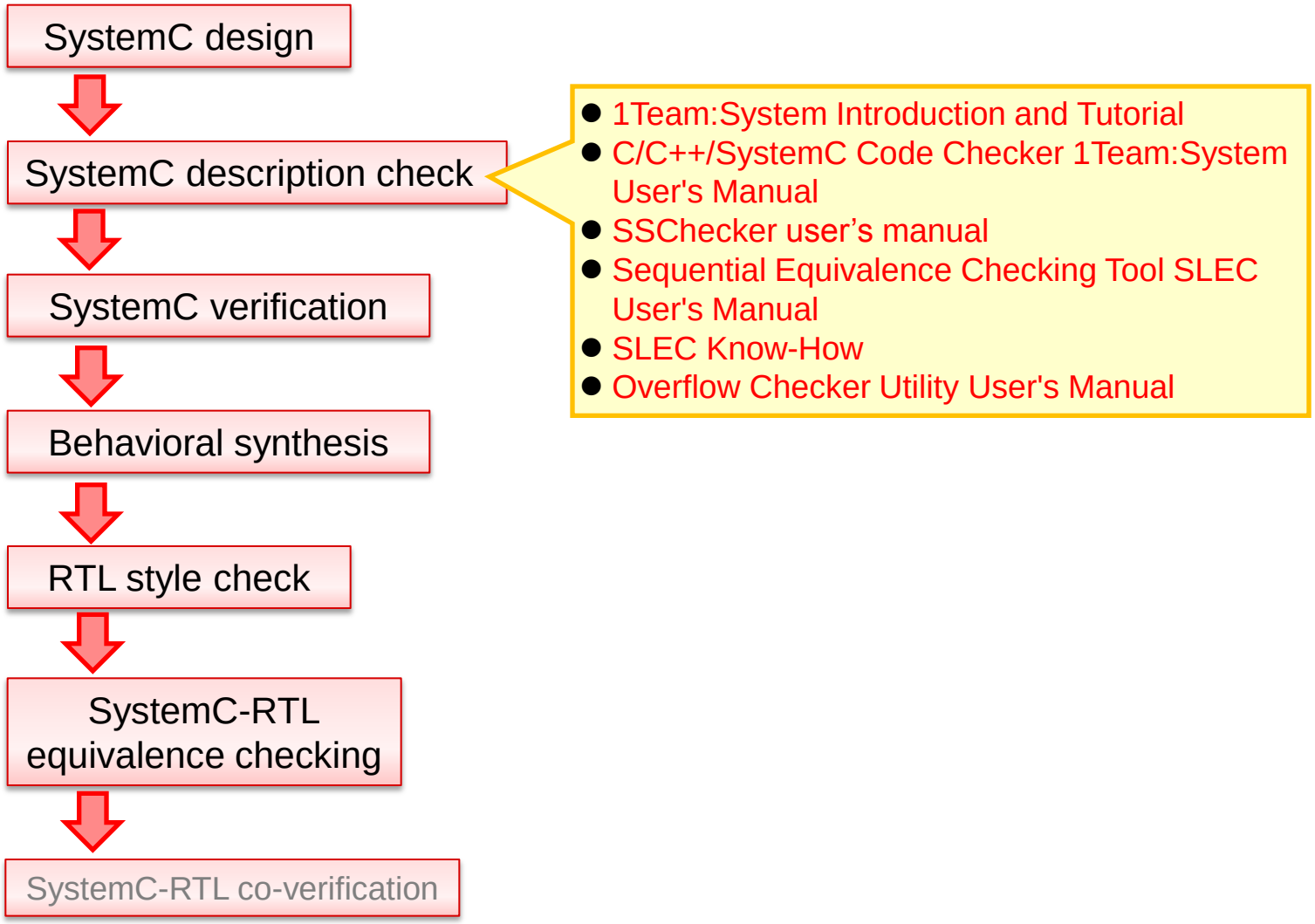
6. How to Refer to Documents Related to HLD (4)

- Assumed purpose of use : To learn and firmly understand description style of SystemC model for HLD (for users who have knowledge of C++/SystemC)



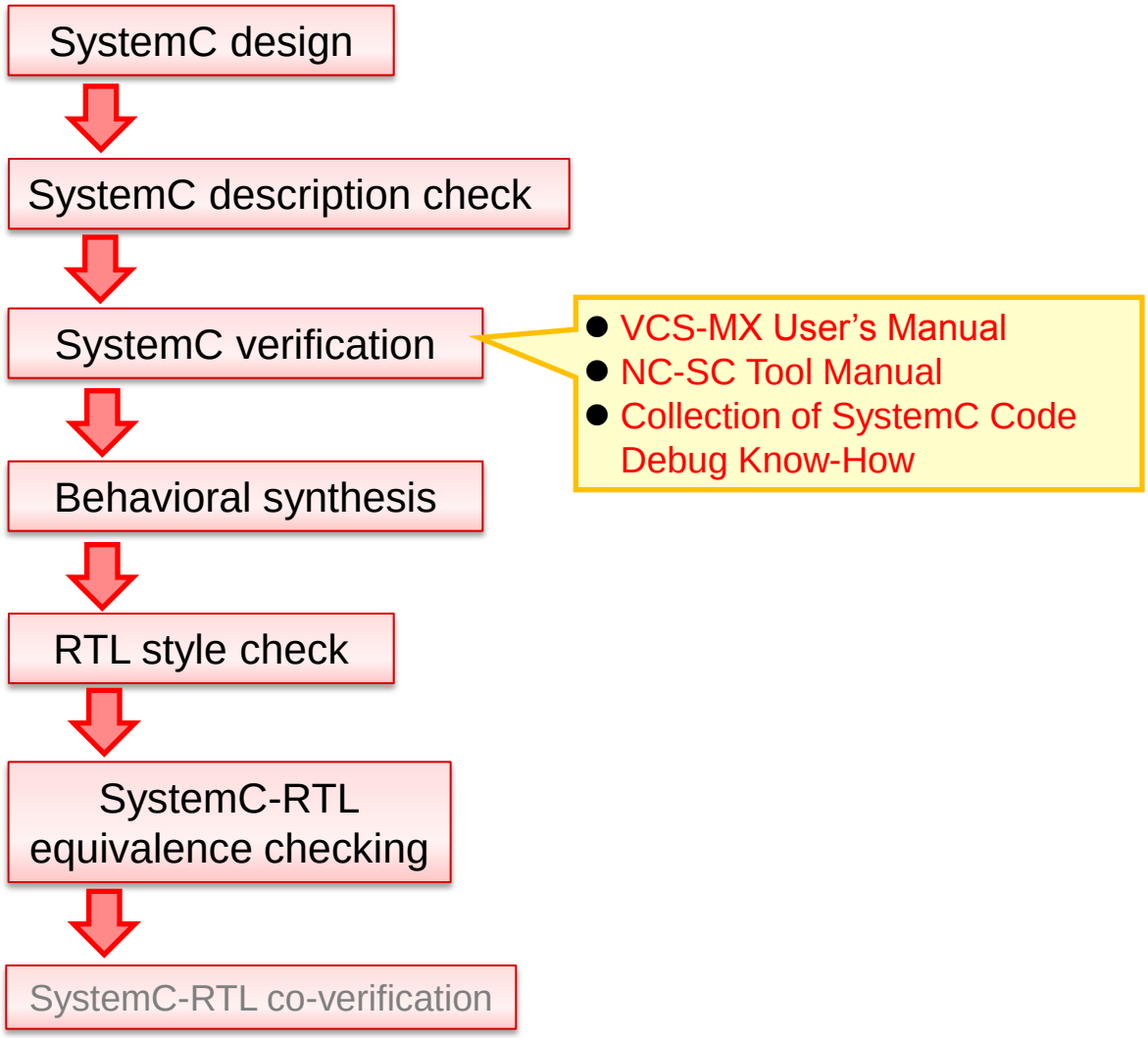
6. How to Refer to Documents Related to HLD (5)

- Assumed purpose of use : To check the style of a created SystemC model



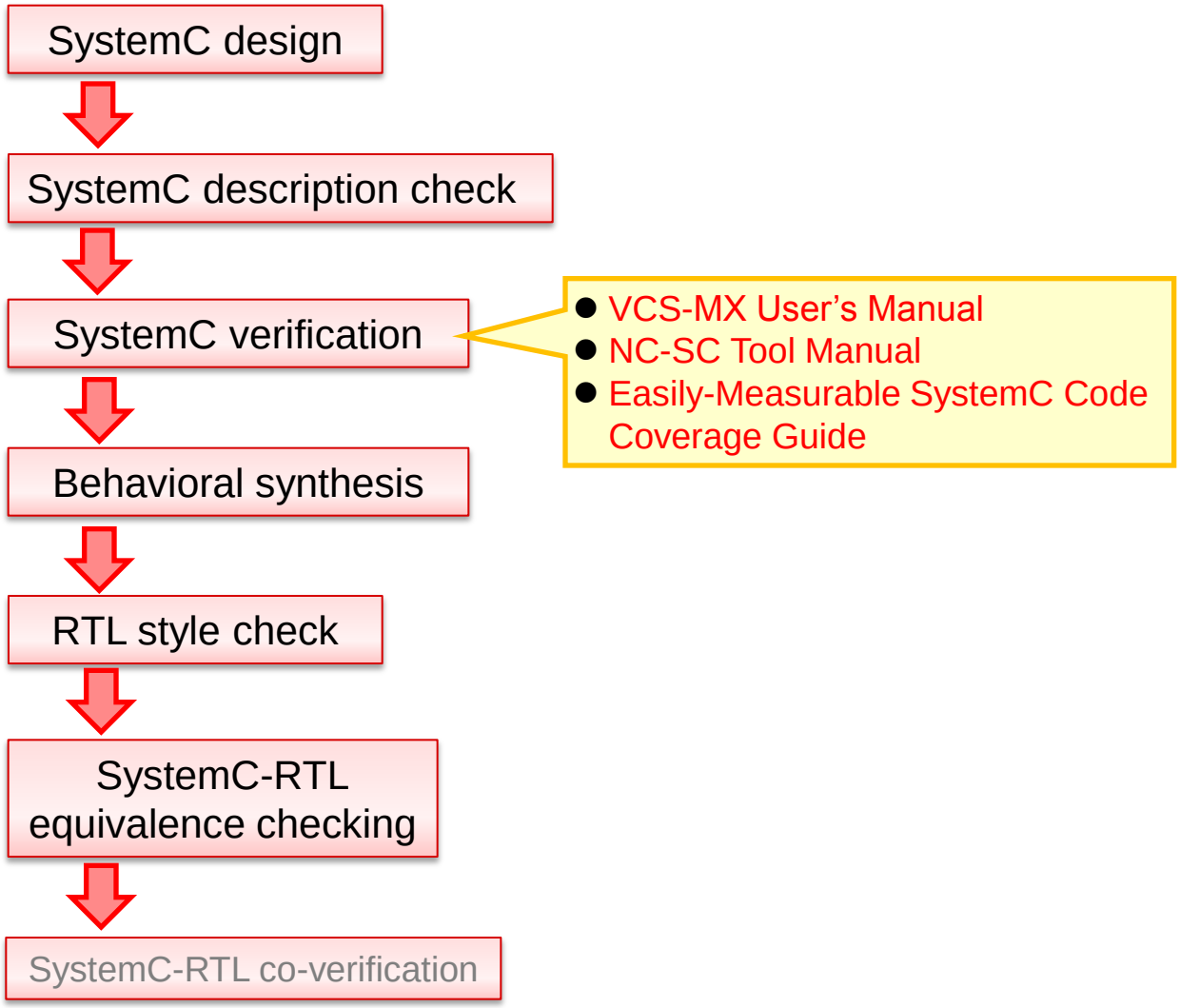
6. How to Refer to Documents Related to HLD (6)

- Assumed purpose of use : To perform code and waveform debug with SystemC simulation



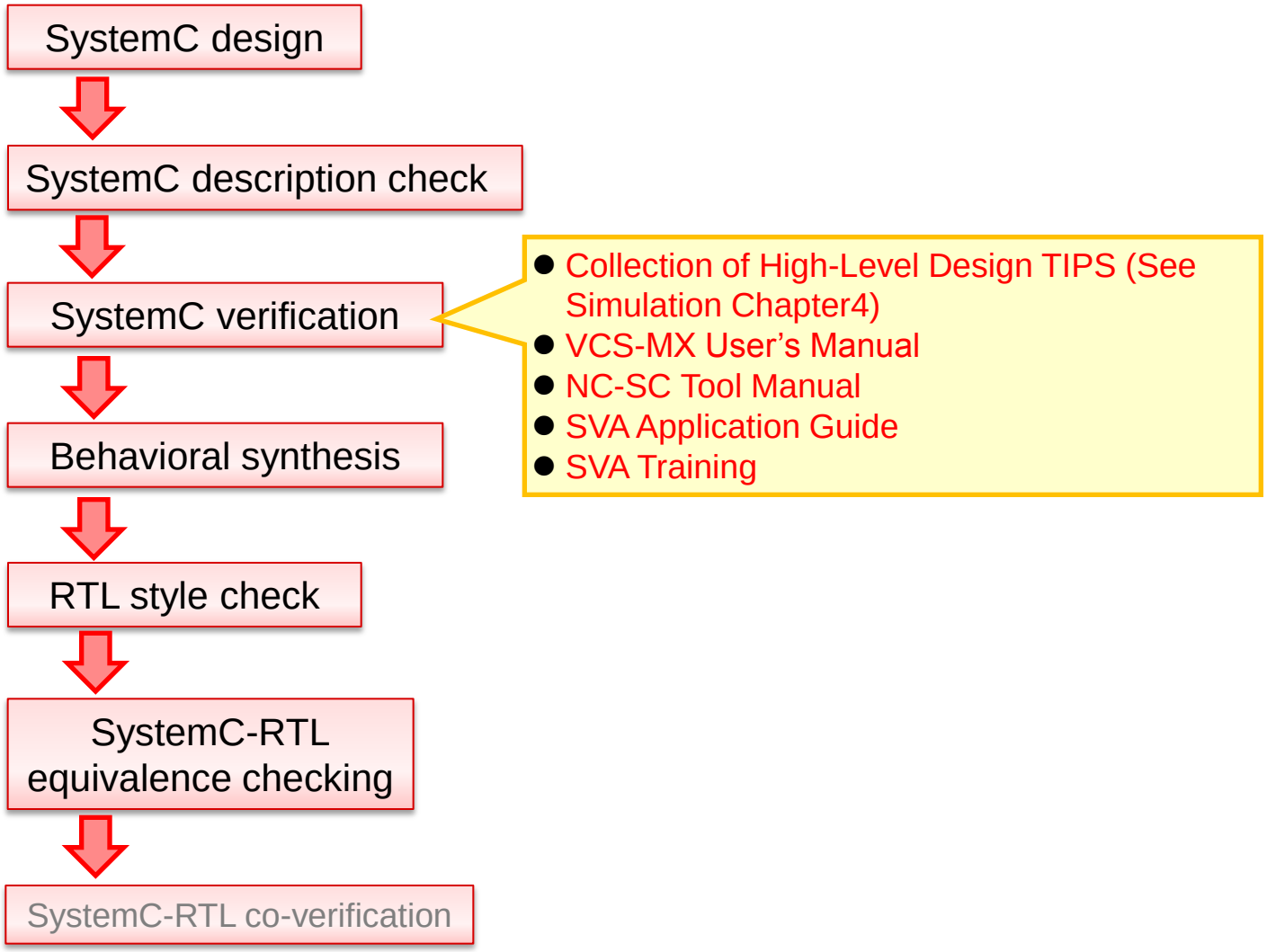
6. How to Refer to Documents Related to HLD (7)

- Assumed purpose of use : To measure the code coverage of SystemC descriptions with SystemC simulation



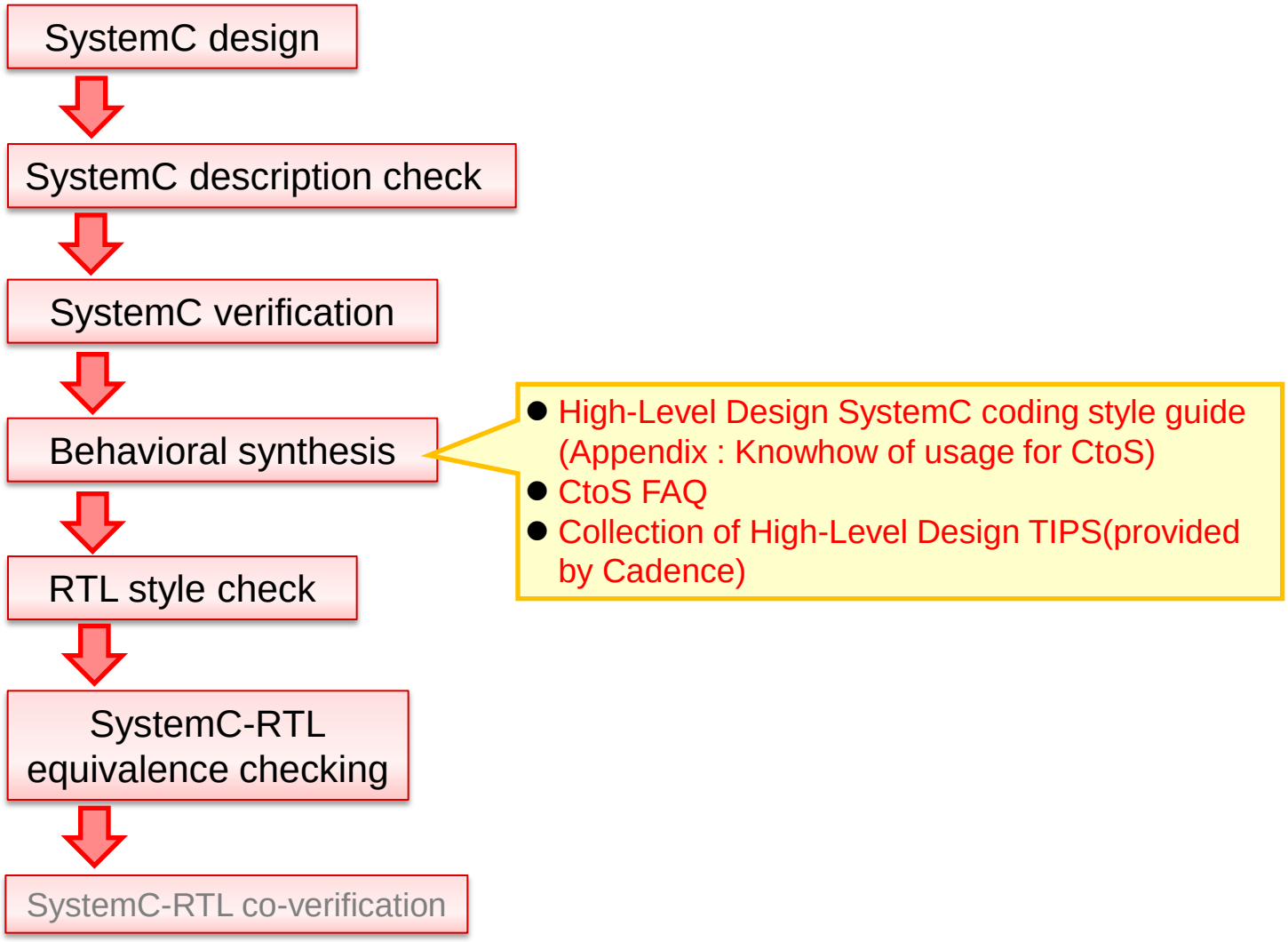
6. How to Refer to Documents Related to HLD (8)

- Assumed purpose of use : To apply assertion for SystemC simulation



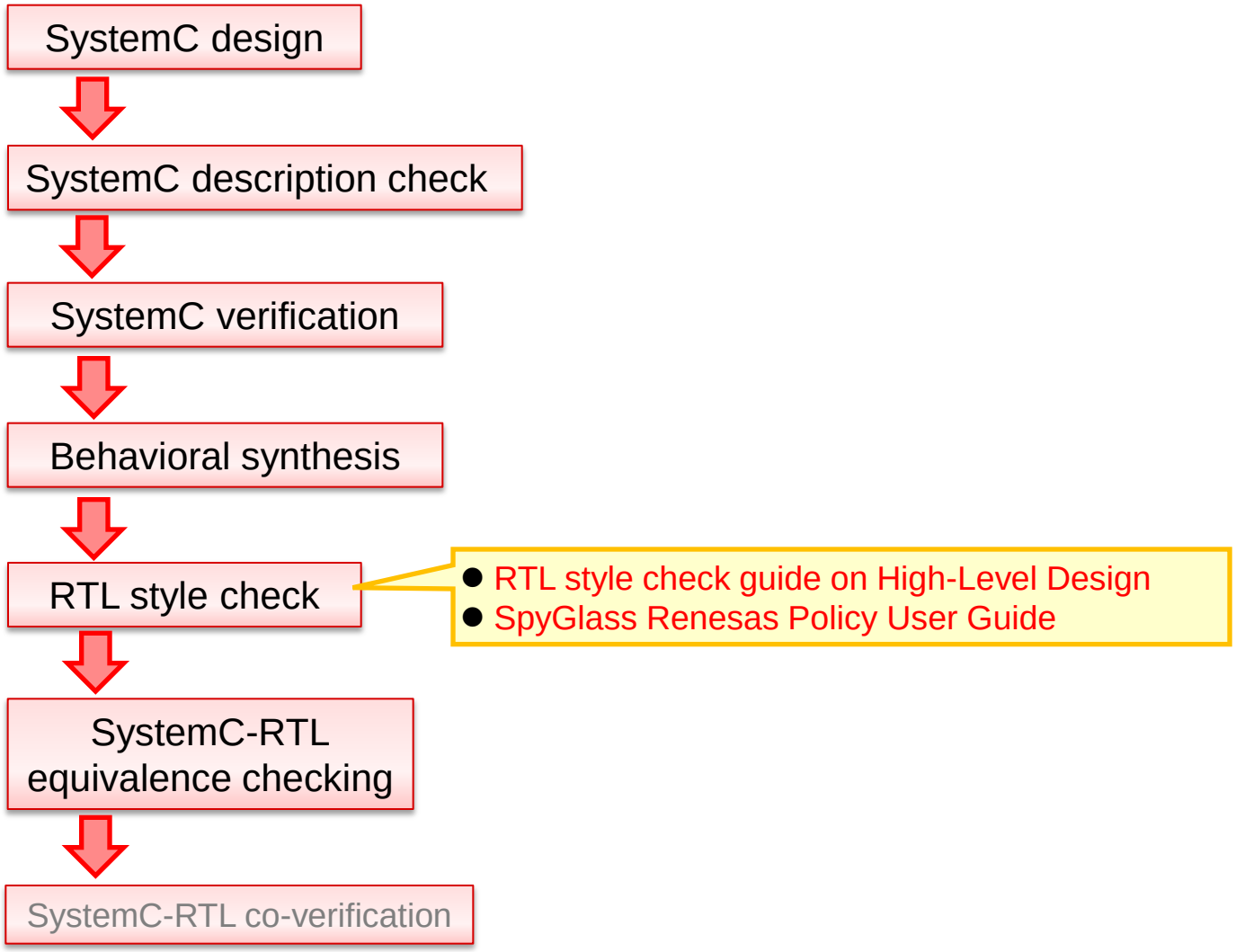
6. How to Refer to Documents Related to HLD (9)

- Assumed purpose of use : To perform behavioral synthesis



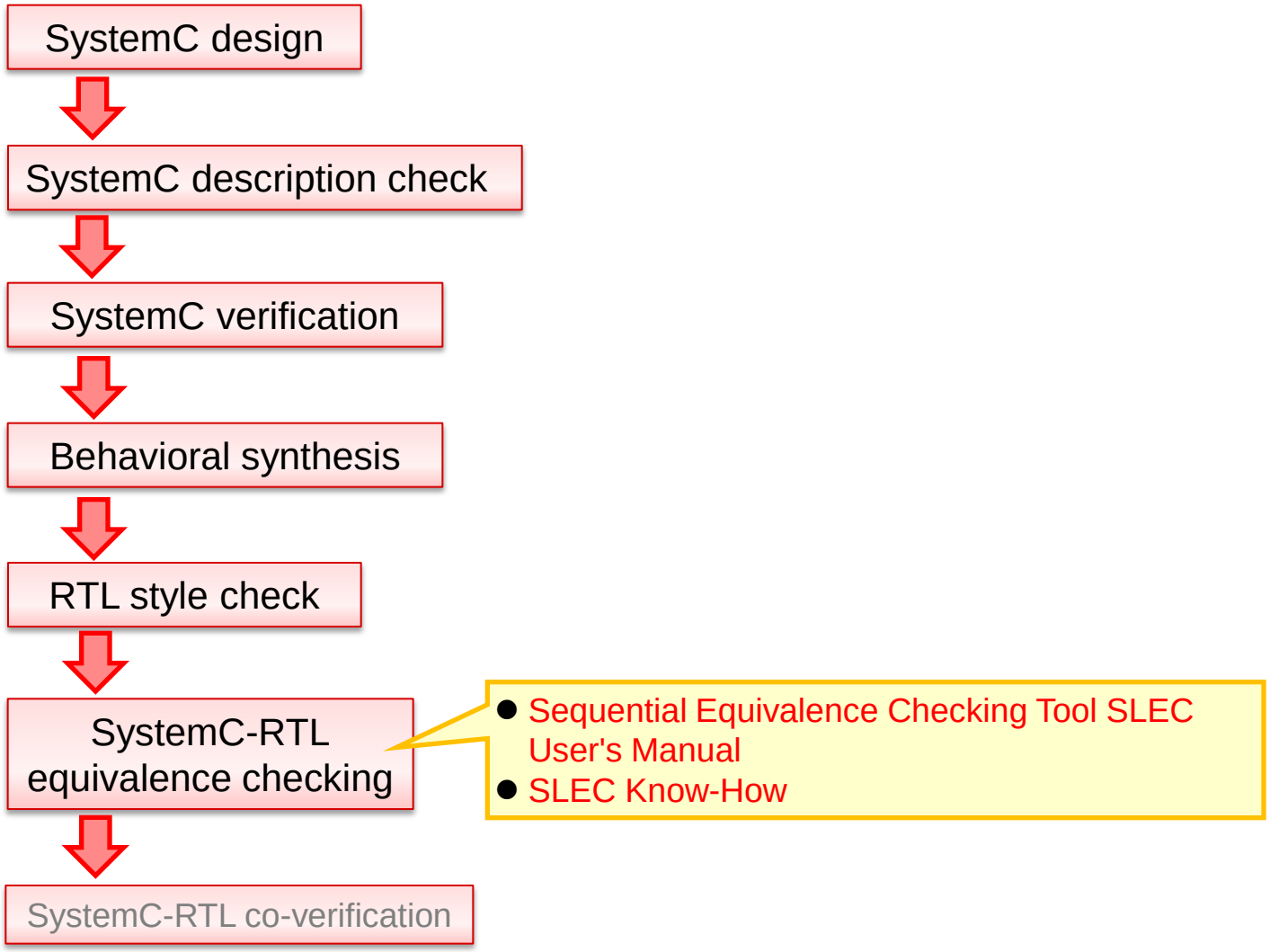
6. How to Refer to Documents Related to HLD (10)

- Assumed purpose of use : To perform an RTL style check



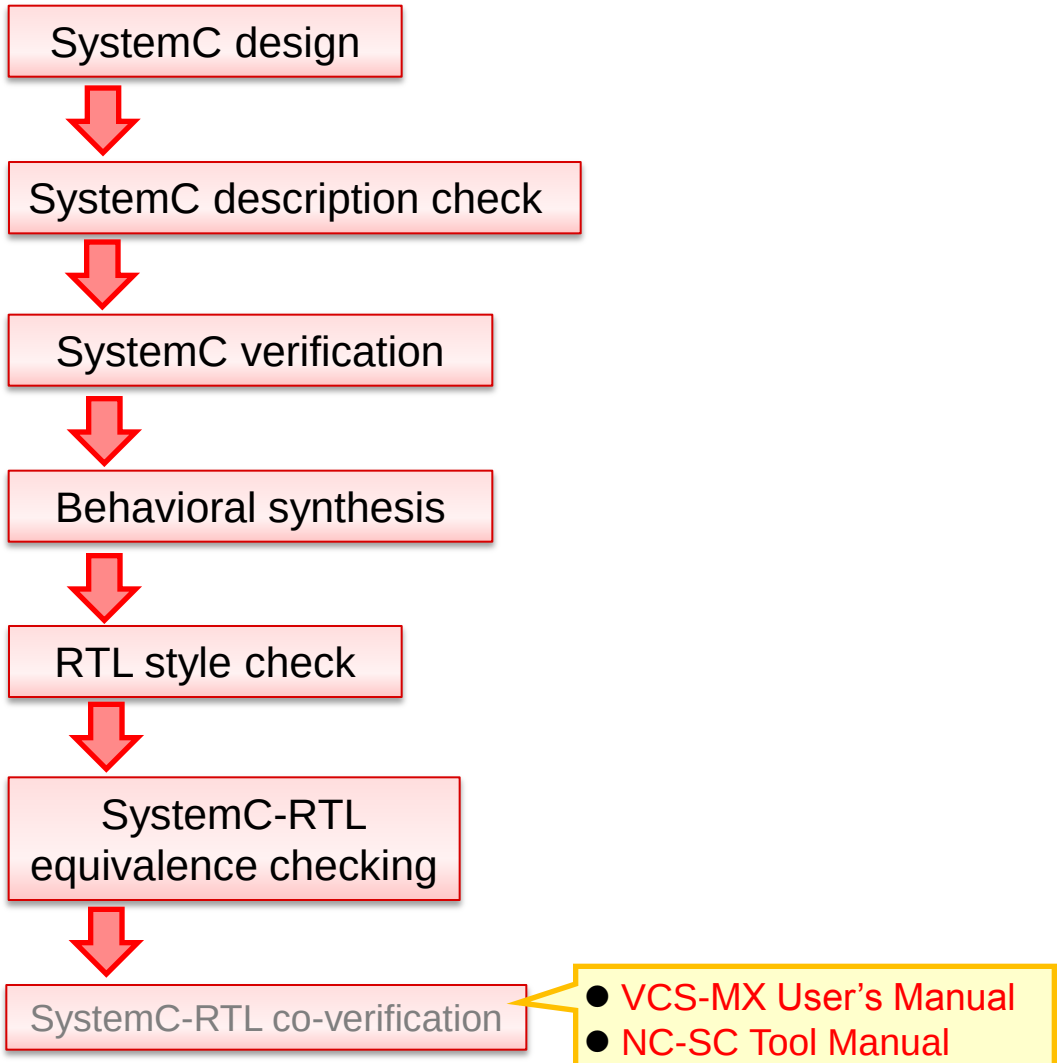
6. How to Refer to Documents Related to HLD (11)

- Assumed purpose of use : To perform SystemC-RTL equivalence checking



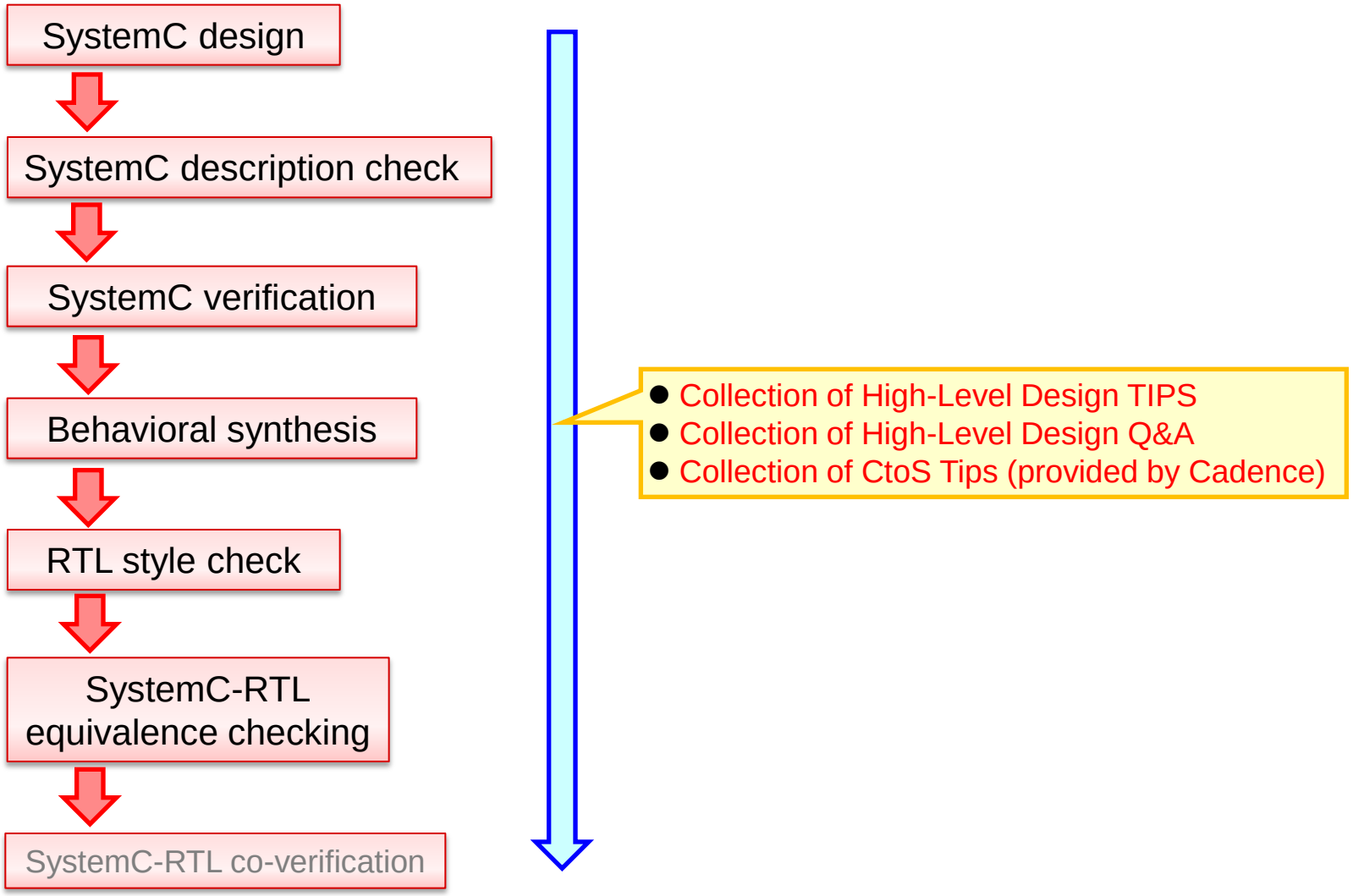
6. How to Refer to Documents Related to HLD (12)

- Assumed purpose of use : To perform SystemC-RTL co-simulation



6. How to Refer to Documents Related to HLD (13)

- Assumed purpose of use : To gain various types of know-how related to HLD



7. Contact Information

- About the contents of this guide

  Contact one of those in charge:
  - FEDT : Oshima (yoshiki.oshima.wm@renesas.com)
  - FEDT : Fujii (mototsugu.fujii.xw@renesas.com)

- Contents of the various documents mentioned in this guide

  Contact the people mentioned in the contact information in the individual documents.

Revision History (1)

Rev. No.	Contents	Approved by	Checked by	Issued by
1.0	• First Printing	FEDT Fujii 10/12/3	FEDT Fujii 10/12/3	FEDT Nonaka 10/12/3
2.0	• Wholly-reviewing the contents based on the latest design flow and document system. • Adding "Flow for Creating SystemC Descriptions for High-Level Design" (Chapter 3). • Adding "Tools Used in High-Level Design" (Chapter 4). • Adding "How to Refer to Documents Related to High-Level Design" (Chapter 6).	FEDT Fujii 13/1/23	-	FEDT Oshima 13/1/23

Revision History (2)

Rev. No.	Contents	Approved by	Checked by	Issued by
2.1	• Wholly-reviewing the contents based on the latest design flow and document system. • Tools newly added Overflow Checker Common HLD environment • Documents newly added CtoS FAQ AMBA bus protocol verification methods for HLD Overflow Checker Utility User's Manual HLD Environment user's manual	FEDT Fujii 13/3/27	-	FEDT Oshima 13/3/27

Renesas Electronics Corporation