

ACE Assertion IP

User Guide

Version 2023.03, March 2023





Copyright Notice and Proprietary Information

© 2023 Synopsys, Inc. All rights reserved. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Third-Party Software Notices

VCS® and configurations of VCS includes or is bundled with software licensed to Synopsys under free or open-source licenses. For additional information regarding Synopsys's use of free and open-source software, refer to the `third_party_notices.txt` file included within the `<install_path>/doc` directory of the installed VCS software.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <http://www.synopsys.com/Company/Pages/Trademarks.aspx>.

All other product or company names may be trademarks of their respective owners.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

www.synopsys.com

Contents

Figures	5
Tables	7
Chapter ACE	
Preface	9
Chapter 1	
Introduction	11
1.1 Prerequisites	11
1.2 References	11
1.3 Product Overview	12
1.4 Language and Methodology Support	12
1.5 Feature Support	12
1.5.1 Protocol Features	12
1.5.2 Verification Features	12
1.6 Features Not Supported	12
Chapter 2	
Installation and Setup	15
2.1 Verifying Hardware Requirements	15
2.2 Verifying Software Requirements	15
2.2.1 Platform/OS and Simulator Software	15
2.2.2 Synopsys Common Licensing (SCL) Software	15
2.2.3 Other Third Party Software	16
2.3 Preparing for Installation	17
Chapter 3	
The ACE AIP in a Formal Verification Environment	19
3.1 Introduction to the VC Formal Tool	19
3.2 The ACE AIP in a Formal Environment	20
3.2.1 Instantiating the ACE AIP Using the bind Statement	20
3.2.2 Creating a Tcl File	20
3.2.3 Reading and Running a Tcl File	21
3.2.4 Commonly Used ACE AIP Configurations	24
3.2.5 The ACE AIP As a Master	24
3.2.6 The ACE AIP As a Slave	25
3.2.7 The ACE AIP As a Monitor	26
3.2.8 The ACE AIP As a Constraint Model	26
3.3 Clock and Reset Functionality	26

Chapter 4	27
The ACE AIP Configuration	27
4.1 The ACE AIP Configuration Parameters	27
4.2 The ACE AIP Interface Ports	30
4.2.1 AXI4 Interface Ports	31
4.2.2 ACE Interface Ports	33
4.3 The ACE AIP Properties	34
4.3.1 Write Address Channel Properties	34
4.3.2 Write Data Channel Properties	36
4.3.3 Write Response Channel Properties	37
4.3.4 Read Address Channel Properties	37
4.3.5 Read Data Channel Properties	40
4.3.6 Barrier Properties	41
4.3.7 ACE Specific Properties (Not applicable for ACE-Lite)	44
4.4 Behavior of Properties	58
4.4.1 Properties as Assert Directives	58
4.4.2 Properties as Assume Directives	58
4.4.3 Properties In Cover Directives	58
4.5 The CONFIG_MAXOUTS Parameter Setting Change	59
4.5.1 Background of this Change (when a potential bug can be missed in a slave DUT):	61
4.5.2 Backwards Compatibility	61
Chapter 5	63
The ACE AIP Use Cases	63
5.1 The ACE AIP Examples	63
5.1.1 The ACE-Lite Master AIP With Slave DUT	63
5.1.2 The ACE-Lite Slave AIP With a Master DUT	65
5.1.3 The ACE Master AIP With Slave DUT	67
5.1.4 The ACE Slave AIP With Master DUT	69

Figures

Figure 3-1:	Falsified and Proven Properties	22
Figure 3-2:	Options to Debug Properties	23
Figure 3-3:	Signal Behavior in Waves	23
Figure 3-4:	ACE AIP As a Master	25
Figure 3-5:	ACE AIP As a Slave	26
Figure 5-1:	Slave DUT with ACE-Lite Master AIP	64
Figure 5-2:	ACE-Lite Master AIP – Slave DUT bind example	65
Figure 5-3:	Master DUT with ACE-Lite Slave AIP	66
Figure 5-4:	ACE-Lite Slave AIP – Master DUT bind example	67
Figure 5-5:	Slave DUT with ACE Master AIP	68
Figure 5-6:	ACE Master AIP – Slave DUT bind example	69
Figure 5-7:	Master DUT with ACE Slave AIP	70
Figure 5-8:	ACE Slave AIP – Master DUT bind example	71



Tables

Table 2-1:	AIP Licensing Key Features	16
Table 3-1:	Tcl File Example	21
Table 3-2:	Common Usage Models	24
Table 4-1:	AXI4 AIP Configuration Parameters	27
Table 4-2:	ACE Configuration Parameters	29
Table 4-3:	AXI4 AIP Interface Ports	31
Table 4-4:	ACE related Interface Ports	33
Table 4-5:	Write Address Channel Properties	34
Table 4-6:	Write Data Channel Properties	36
Table 4-7:	Write Response Channel Properties	37
Table 4-8:	Read Address Channel Properties	37
Table 4-9:	Read Data Channel Properties	40
Table 4-10:	Barrier Properties	41
Table 4-11:	ACE Specific Properties (Not applicable for ACE-Lite)	44
Table 4-12:	CONFIG_MAXOUTS Parameter Setting Behavior Change	59
Table 4-13:	Behavior Per Value	59





ACE Preface

This guide contains installation, setup, and usage instructions for the AMBA ACE SystemVerilog AIP. It is intended for use by the design or verification engineers who want to verify RTL designs with an AMBA ACE interface. Readers are assumed to be familiar with the AMBA ACE protocol, SystemVerilog Assertions and the Verilog language.

This chapter includes the following sections:

- [Guide Organization](#)
- [Web Resources](#)
- [Customer Support](#)

Guide Organization

The chapters of this guide are organized as follows:

Chapter 1, “[Introduction](#)”, introduces the Synopsys DesignWare (DW) ACE AIP and its features.

Chapter 2, “[Installation and Setup](#)”, describes system requirements and provides instructions on how to install, configure, and begin using the Synopsys DesignWare (DW) ACE AIP.

Chapter 3, “[The ACE AIP in a Formal Verification Environment](#)”, introduces the VC Formal tool and the ACE AIP usage in the formal environment.

Chapter 4, “[The ACE AIP Configuration](#)”, presents an insight into the functionality of the ACE AIP.

Chapter 5, “[The ACE AIP Use Cases](#)”, presents an example of how to install and use the Synopsys ACE AIP.

Web Resources

AMBA ACE AIP is compliant with the following specifications:

- AMBA specification ARM IHI 0022E (ID022613)
- AMBA FAQ document

<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.amba/index.html>

Customer Support

To obtain support for your product, choose one of the following:

- Open a Case Through SolvNet:

Go to <http://onlinecase.synopsys.com/Support/OpenCase.aspx>

- Provide the requested information, including:
 - **Product L1:** VC Formal
 - **Product L2:** AIP
 - **Product L3:** ACE AIP
 - **Product Version:** 2019.06-SP1-1
 - Fill in the remaining fields according to your environment and issue.
- Send an e-mail message to support_center@synopsys.com
 - Include product name, sub-product name, and product version (as noted above) in your e-mail, so it can be routed correctly.
- Telephone your local support center:
 - North America:
Call 1-800-245-8005 from 7 AM to 5:30 PM Pacific time, Monday through Friday
 - All other countries:
<http://www.synopsys.com/Support/GlobalSupportCenters>

Introduction

This document describes the ACE protocol checkers available in the ACE Assertion IP (AIP). It also describes how to configure the AIP, including all parameters related to the configuration, how to instantiate the AIP, and so on.

Assertion IP is significant in reducing the verification effort and improving the design quality. Its value comes from the fact that the assertions can passively monitor the design behavior by simply attaching them to the target design, without modifying the RTL. Pre-built assertions from assertion IP provide a powerful quality criteria for sign-off.

This chapter consists of the following sections:

- [Prerequisites](#)
- [References](#)
- [Product Overview](#)
- [Language and Methodology Support](#)
- [Feature Support](#)
- [Features Not Supported](#)

**Note**

Based on the AMBA Progressive Terminology updates, you must interpret the term Master as Manager and Slave as Subordinate in the AIP documentation and messages.

1.1 Prerequisites

Familiarity with the ACE protocol, SystemVerilog Assertions and the SystemVerilog language.

1.2 References

ACE AIP is compliant with the following specifications:

- AMBA specification, ARM IHI 0022E (ID022613)
- AMBA FAQ document

<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.amba/index.html>

1.3 Product Overview

The ACE AIP is a suite of SystemVerilog Assertions and related SystemVerilog code that are meant to be used with RTL designs. This AIP is used to verify the RTL with the VC Formal tool.

The ACE AIP consists of the following:

- Assertions properties
- Assume properties
- Cover properties
- Synthesizable modeling SystemVerilog for properties

1.4 Language and Methodology Support

In the current release, the ACE AIP suite supports the following languages and methodology:

- SystemVerilog Assertion
- Verilog

1.5 Feature Support

1.5.1 Protocol Features

The ACE AIP currently supports the following protocol functions:

- All data widths
- All address widths
- All transfer types
- All burst types and burst lengths (up to 256 in ACE as opposed to 16 in AXI3)
- All protection types
- All slave response types
- Exclusive transactions

1.5.2 Verification Features

The ACE AIP currently supports the following verification features:

- Ability to configure as Master
- Ability to configure as Slave
- Ability to configure as Monitor
- Ability to configure as Constraint Provider
- Protocol Checks
- Selective inclusion or exclusion of multiple features like CONFIG_X_CHECK, WR_OUT_OF_ORDER, RD_INTERLEAVE, ENABLE_ASSERT, ENABLE_ASSUME and ENABLE_COVER. Refer [Tables 4-1](#) and [Tables 4-2](#) for more details.

1.6 Features Not Supported

- None





Installation and Setup

This chapter guides you through installing and setting up the ACE AIP. When you complete the checklist mentioned below, the provided example gets operational and you can use the ACE AIP.

The checklist consists of the following major steps:

- [“Verifying Hardware Requirements”](#) on page 15
- [“Verifying Software Requirements”](#) on page 15
- [“Preparing for Installation”](#) on page 17

2.1 Verifying Hardware Requirements

The ACE AIP suite requires a Linux workstation configured as follows:

- 400 MB available disk space for installation
- 1 GB available swap space
- 1 GB RAM (physical memory) recommended
- FTP anonymous access to ftp.synopsys.com (optional)

2.2 Verifying Software Requirements

This section lists software that the ACE AIP requires.

- VCS version L-2016.06-SP1 (Simulator)
- Verdi version L-2016.06-SP1 (Debugger)
- VCF version L-2016.06-SP1 (Formal)
- VC version L-2016.06-SP1 (Verification Compiler Platform)

2.2.1 Platform/OS and Simulator Software

- VC Formal tool is required

2.2.2 Synopsys Common Licensing (SCL) Software

The ACE AIP requires the following license feature:

- AIP-ACE-SVA

The following topics describe the required environment variables and path settings for the ACE AIP:

2.2.2.1 Running the ACE AIP on the VC Formal Tool

To run the ACE AIP on the VC Formal tool, set the following environment variable:

SNPSLMD_LICENSE_FILE: The absolute path to file(s) that contains the license keys for Synopsys software (AIP and/or other Synopsys Software tools) or the port@host reference to this file.

Example:

```
setenv SNPSLMD_LICENSE_FILE <port>@<server>:${SNPSLMD_LICENSE_FILE}
```

or

```
setenv SNPSLMD_LICENSE_FILE <full path to the license file>:${SNPSLMD_LICENSE_FILE}
```

2.2.2.2 Running the ACE AIP on VCS

To run the ACE AIP on VCS, set the following two environment variables:

- **SNPSLMD_LICENSE_FILE**
- **DW_LICENSE_FILE:** The absolute path to file that contains the license keys for the AIP product software or the port@host reference to this file.

Example:

```
setenv SNPSLMD_LICENSE_FILE <port>@<server>:${SNPSLMD_LICENSE_FILE}
```

```
setenv DW_LICENSE_FILE <port>@<server>:${DW_LICENSE_FILE}
```

or

```
setenv SNPSLMD_LICENSE_FILE <full path to the license file>:${SNPSLMD_LICENSE_FILE}
```

```
setenv DW_LICENSE_FILE <full path to the license file>:${DW_LICENSE_FILE}
```

Table 2-1 AIP Licensing Key Features


Package Name	Feature Keys	Included Titles
VC Formal AIP AMBA APB	AIP-APB-SVA	APB4, APB3, APB2 and APB
VC Formal AIP AMBA AHB	AIP-AHB-SVA	AHB5, AHB and AHB-Lite
VC Formal AIP AMBA AXI	AIP-AXI-SVA	AXI4, AXI4-Lite and AXI3
VC Formal AIP AMBA ACE	AIP-ACE-SVA	ACE, ACE, AXI4, AXI4-Lite and AXI3
VC Formal AIP AMBA5 AXI	AIP-AXI5-SVA	AXI5 and AXI5-Lite
VC Formal AIP AMBA5 CHI	AIP-CHI-SVA	CHI B, C, D, and E

2.2.3 Other Third Party Software

Adobe Acrobat: The ACE AIP documentation is available in the Acrobat PDF files. You can get the Adobe Acrobat Reader for free from <http://www.adobe.com>.

HTML browser: The ACE AIP coverage reports can be viewed in HTML. The following browser/platform combinations are supported:

- Microsoft Internet Explorer 6.0 or later (Windows)

- 
- Firefox 1.0 or later (Windows and Linux)
 - Netscape 7.x (Windows and Linux)

2.3 Preparing for Installation

Ensure that your environment and PATH variables are set correctly. For information on the environment variables and path settings required for the ACE AIP, see [“Synopsys Common Licensing \(SCL\) Software”](#) on page 15.



The ACE AIP in a Formal Verification Environment

This chapter describes the simulation environment for the ACE AIP, usage of the VC Formal tool, and the ACE AIP usage in a formal verification environment. This chapter discusses the following topics:

- [“Introduction to the VC Formal Tool”](#) on page 19
- [“The ACE AIP in a Formal Environment”](#) on page 20
- [“Clock and Reset Functionality”](#) on page 26

3.1 Introduction to the VC Formal Tool

The VC Formal tool is used to verify assertion properties by examining all sequences of possible value combinations. These valid inputs are constrained by the assume properties. The VC Formal tool provides several pieces of information as follows:

- Number of proven properties
- Number of falsified properties
- Number of vacuous properties
- Number of covered properties
- Number of witness properties

The VC Formal tool is useful for debugging failing properties by means of its GUI interface. For running the properties in the VC Formal tool, the following information must be provided through the tool's command line interface or through a Tcl script:

- The path of the ACE AIP source code
- The path of the RTL source code (if validating the RTL)
- Clock information
- Reset information
- Timeout details

3.2 The ACE AIP in a Formal Environment

The ACE AIP has ACE Master properties and ACE Slave properties. These properties are connected to either ACE Master or Slave module (for example, ACE Slave DUT to Master Properties) interface signals, then the formal verification tool, VC Formal, is used to verify the functional correctness of the module.

To use the ACE AIP in a formal environment, perform the following steps in a sequence:

- Instantiate the ACE AIP and connect it to the DUT using the `bind` statement
- Creating a Tcl file
- Reading and running a Tcl file
- Analyzing results

3.2.1 Instantiating the ACE AIP Using the `bind` Statement

Create the `bind` file to instantiate the ACE AIP and bind the ACE AIP to the design. Map the module and port names in the design to those of the AIP in the `bind` statement. Pass valid values to the configuration parameters of the AIP. The next step in the process is compiling the files.



Note

If a signal corresponding to the ACE AIP port does not exist in the DUT, set inactive value or the value expected by the DUT for the port. For example, if the DUT does not have the `awlock` signal, set with `1'b0`.

3.2.2 Creating a Tcl File

To compile the DUT and the attached ACE AIP (including RTL files, AIP files, and the `bind` file), create a Tcl file, where path to RTL directory (ACE AIP directory) and the `bind` file are set. The DUT clock and reset are initialized with the `create_clock` and `create_reset` commands, as shown in [Table 3-1](#). VC Formal can report assertion status (proven, falsified, vacuous or inclusive using the `report_fv` command). For more command options, see the VC Formal User Guide. In case RTL is ACE-Lite protocol, analyze '`snps_ace_lite_aip.sv`' file. In case RTL is ACE protocol, analyze '`snps_ace_aip.sv`' file.

Table 3-1 Tcl File Example

<pre># variables setting set AIP_HOME \$::env(VC_STATIC_HOME)/packages/aip set AIP_SRC \$AIP_HOME/ACE_AIP/src set TBH_DIR ../tb set TCL_DIR ../tcl set design ace_dut # vcs option set vcs " \ +incdir+\${AIP_SRC} \ \${AIP_SRC}/snps_amba4_aip_pkg.sv \ \${AIP_SRC}/snps_ace_aip.sv \ \${TBH_DIR}/ace_dut.v \ \${TBH_DIR}/bind_ace_aip.sv \ -assert svaext "</pre>	<pre># Enable all Formal Debug Modes set_fml_appmode FPV # analyze and elaborate design and AIP read_file -sva -top \$design -format sverilog - vcs "\$vcs" # clock setting create_clock aclk -period 100 # reset setting create_reset aresetn -low sim_run -stable sim_save_reset # execute proof check_fv -run_finish { report_fv }</pre>
---	--

3.2.3 Reading and Running a Tcl File

To read and run a Tcl file, use either of the following two modes:

- “GUI Mode” on page 21
- “Reading and Running Tcl File in the Batch Mode” on page 23

3.2.3.1 GUI Mode

To read a VC formal Tcl file, invoke the VC Formal tool in the GUI mode and perform the following steps:

1. Navigate to the folder containing the Tcl file.
2. Open VC Formal in the GUI mode using the `vcf -gui -f <tcl file>` command.

After all the properties are executed, Verdi tool displays the list of properties as shown in [Figure 3-1](#), where the red cross indicates a falsified property and the green tick marks indicate proven properties.

Figure 3-1 Falsified and Proven Properties

File View Source OneTrace Tools Window Help

Appmode: FPV By: 400 x 1ns

Instance

Hierarchy

Module

ace_tb

ace_mst_cst

ace_slv_mon

Packages

_snovas_units

ace_tb

snps_ace_aip

snps_ace_aip

VCFGoalList

Time 8H Max Cycle -1

<Enter name Match Value>

☒ All ☒

3.2.3.1.1 Analyzing Results for the GUI Mode Run

After running a session, its results are dumped into the `vcf.log` file. This log file gives the number of proven, falsified, vacuous, inconclusive, and covered properties. When there are no falsifications, the design is verified with regard to the parameters configured in a Tcl file.

If properties are falsified, you should debug them to find the root cause. Perform the following steps to debug the falsification:

1. Right-click on any of the failures which you need to be debug to view the options, such as View Trace, Explore the Property, Report, and so on.
2. When you select the View Trace option, waveforms are opened, providing signal details and falsification depth. You can dump other signals required for debugging into the waveform as well.

You can also explore the options in the VC Formal tool and debug the failure. See the VC Formal User Guide for more information on options. [Figure 3-2](#) and [Figure 3-3](#) shows options to debug properties and signal behavior in the waveforms respectively.

Figure 3-2 Options to Debug Properties

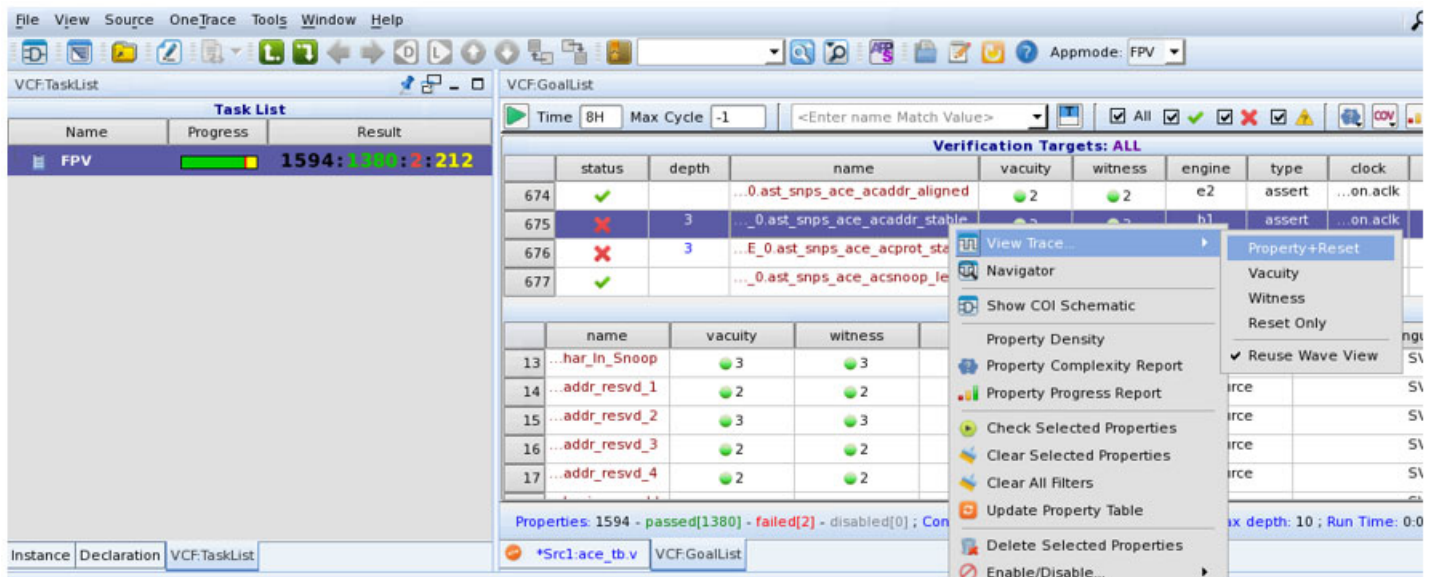
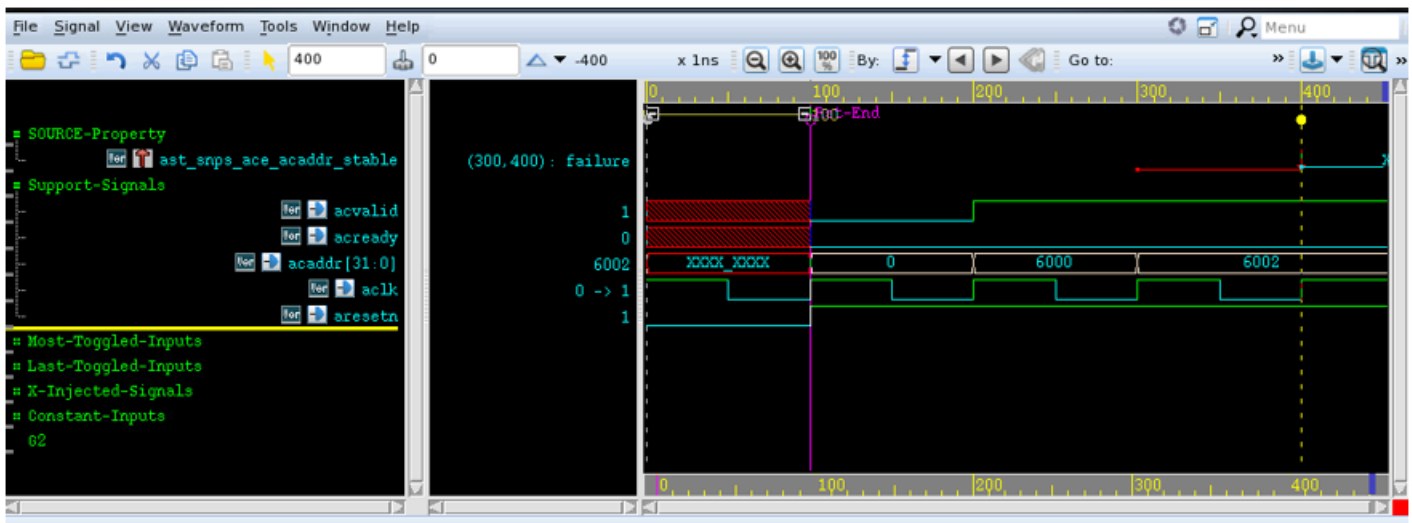


Figure 3-3 Signal Behavior in Waves



3.2.3.2 Reading and Running Tcl File in the Batch Mode

To read a VC Formal Tcl file using the command line, perform the following steps:

1. Check whether VC Static is installed and exists in PATH. For this, use the following command:

```
% which vcf
```

If the command gives the 'Command not found' error, install the VC Static tool.

2. Run a Tcl file using the following command:

```
% vcf -f <tcl file name> -full64
```

For more information on the VC formal command line options, see the VC Formal User Guide.

Once the Tcl file is read, the tool runs a formal session and give results, such as proven or falsified for various properties.

3.2.3.2.1 Analyzing Results for the Batch Mode Run

After running a session, results are dumped into a log file. This log file gives the number of proven, falsified, vacuous, inconclusive, and covered properties. If there are no falsifications, a design is qualified with regard to the parameters configured in a Tcl file.

If properties are falsified, you should debug them to find the root cause. Perform the following steps to debug the falsification

On the VC Formal window, execute the following commands:

1. `get_props -status falsified`

To display the number of falsified properties.

2. `view_trace -property <property name with path of property shown in get_props command>`

To open the Verdi tool and to display the waves of a falsified property which you want to debug.

3.2.4 Commonly Used ACE AIP Configurations

Table 3-2 Common Usage Models

1	Master	Instantiates the ACE AIP as a master to check the output behavior of a DUT slave and constraint a slave input.
		AGENT_TYPE=MASTER
		By default, ENABLE_ASSERT=1 and ENABLE_ASSUME=1
2	Slave	Instantiates the ACE AIP as a slave to check the output behavior of a DUT master and constraint a master input.
		AGENT_TYPE=SLAVE
		By default, ENABLE_ASSERT=1 and ENABLE_ASSUME=1
3	Monitor	Instantiates the ACE AIP as a monitor to check the behavior of a DUT master and slave.
		AGENT_TYPE=MONITOR
		By default, ENABLE_ASSERT=1 and ENABLE_ASSUME=0
4	Constraint	Instantiates the ACE AIP to constraint the inputs of a DUT master and slave.
		AGENT_TYPE=CONSTRAINT
		By default, ENABLE_ASSERT=0 and ENABLE_ASSUME=1

3.2.5 The ACE AIP As a Master

To verify an ACE SLAVE DUT, set the AGENT_TYPE parameter to MASTER during an AIP instantiation.

When the ACE AIP parameter is set as MASTER, all the ACE AIP properties that are related to the master inputs are declared as assertions, and all the ACE AIP properties that are related to the master outputs are declared as assumptions. This is required to make the ACE AIP AIP behave as a master.

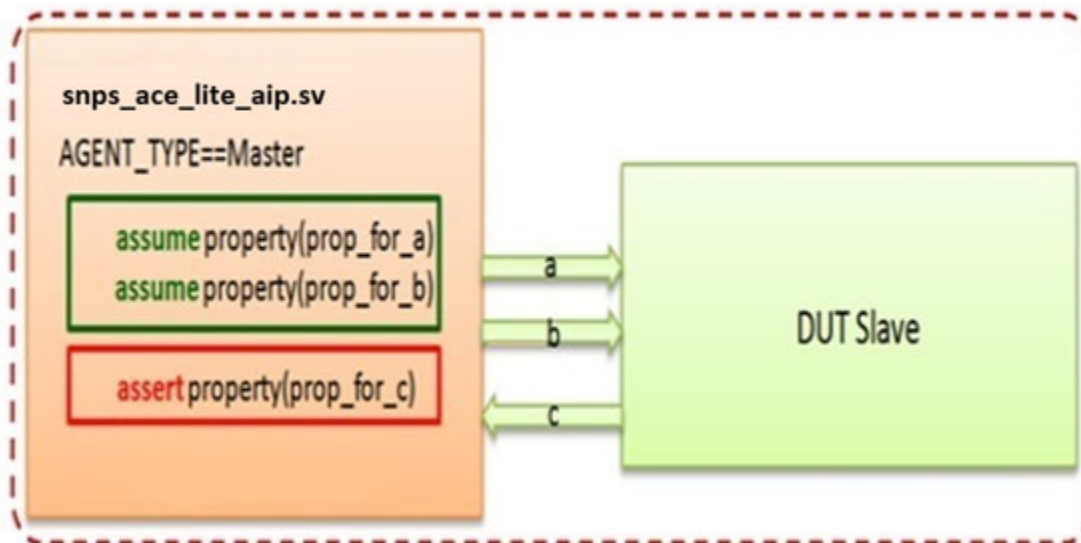
To disable all assert, assume, or cover properties, explicitly set the following parameters to value 0:

- ENABLE_ASSERT
- ENABLE_ASSUME
- ENABLE_COVER

For example, if you want to enable ACE slave checks (assert) only and do not want to apply constraints on ACE slave inputs (assume), set `ENABLE_ASSERT=1` and `ENABLE_ASSUME=0`. This enables all assert properties related to an ACE slave DUT, but disables all assume properties.

Figure 3-4 checks the behavior of an ACE slave DUT output and constraints the inputs of an ACE slave DUT to valid values.

Figure 3-4 ACE AIP As a Master



3.2.6 The ACE AIP As a Slave

To verify an ACE master DUT, set the `AGENT_TYPE` parameter to `SLAVE` during the ACE AIP instantiation. When this parameter is set as `SLAVE`, all the ACE AIP properties that are related to the slave inputs are declared as assertions, and all the ACE AIP properties that are related to the slave outputs are declared as assumptions. This is required to make the ACE AIP behave as a slave.

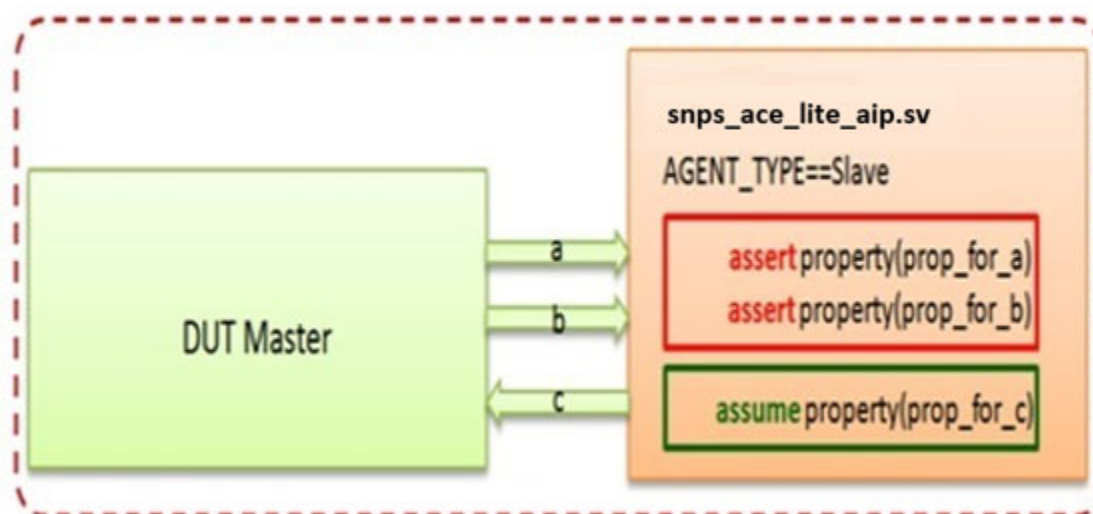
To disable all assert, assume, or cover properties, explicitly set the following parameters to value 0:

- ENABLE_ASSERT
- ENABLE_ASSUME
- ENABLE_COVER

For example, if you want to enable the ACE slave checks (assert) only and do not want to apply constraints on the ACE master inputs (assume), set `ENABLE_ASSERT=1` and `ENABLE_ASSUME=0`. This enables all assert properties related to an ACE master DUT, but disables all assume properties.

Figure 3-5 checks the behavior of an ACE master DUT output and constraints the inputs of the ACE Master DUT to valid values.

Figure 3-5 ACE AIP As a Slave



3.2.7 The ACE AIP As a Monitor

To verify the behavior of an ACE master and slave DUT, set the `AGENT_TYPE` parameter to `MONITOR` during the AIP instantiation. When this parameter is set to `MONITOR`, ACE AIP is instantiated as a monitor to check the behavior of both the inputs and outputs of the DUT slave and DUT master.

By default, `ENABLE_ASSERT=1` and `ENABLE_ASSUME=0`, this disables all the assume properties.

ACE AIP Use Case As A Monitor

In an RTL verification environment, the ACE AIP can be instantiated as a monitor on each ACE interface to check for protocol correctness.

3.2.8 The ACE AIP As a Constraint Model

If the `AGENT_TYPE` parameter is set to `CONSTRAINT`, the ACE AIP is instantiated to constraint the DUT slave input and DUT master input.

By default, `ENABLE_ASSERT=0` and `ENABLE_ASSUME=1`.

ACE AIP Use Case As A Constraint

To verify the RTL in a formal verification environment, the ACE AIP can be used in the constraint mode to generate stimulus to the RTL.

3.3 Clock and Reset Functionality

- To run the ACE AIP and a design in the VC Formal tool, create clock using the following command:

```
create_clock <design_clk> -period <time_period>
```

This command specifies clock period.

- To run the ACE AIP and a design in the VC Formal tool, create reset using the following command:

```
create_reset<design_reset> -low/high
```

The reset can be active low or active high depending on the type of reset in a design.

The formal analysis of properties starts after the reset state.

The ACE AIP Configuration

This chapter describes about configuration of the ACE AIP in the following sections:

- [“The ACE AIP Configuration Parameters”](#) on page 27
- [“The ACE AIP Interface Ports”](#) on page 30
- [“The ACE AIP Properties”](#) on page 34
- [“Behavior of Properties”](#) on page 58
- [“The CONFIG_MAXOUTS Parameter Setting Change”](#) on page 59

4.1 The ACE AIP Configuration Parameters

Table 4-1 AXI4 AIP Configuration Parameters

Parameter	Description	Default Value
CONFIG_LOWPOWER	Indicates whether low power-related properties are instantiated or not.	1
CONFIG_X_CHECK	Indicates whether X-related properties are instantiated or not.	1
CONFIG_WAITS	Indicates whether xREADY latency property is instantiated or not.	1
CONFIG_USER	Indicates whether USER signal related properties are instantiated or not.	1
CONFIG_RECOMMEND	Indicates whether ARM recommendation properties are instantiated or not.	1
CONFIG_WSTRB	Indicates whetherWSTRB-related properties are instantiated or not.	1

Table 4-1 AXI4 AIP Configuration Parameters

Parameter	Description	Default Value
CONFIG_MAXOUTS	Indicates whether maximum outstanding transaction related properties are instantiated or not. (0: Disable 1: Enable when need to check design implementation 2: Enable when need to constrain). For more information, refer to Section 4.5 .	1
EXCL_DEPTH	Indicates an exclusive-monitoring depth in a slave	4
RD_MAX_BURSTS	Maximum number of outstanding read bursts	4
WR_MAX_BURSTS	Maximum number of outstanding write bursts	4
MAXBURSTLENGTH	Maximum configured burst length	256
AW_MAX_WAITS	Maximum number of wait cycles from AWVALID to AWREADY	16
W_MAX_WAITS	Maximum number of wait cycles from WVALID to WREADY	16
B_MAX_WAITS	Maximum number of wait cycles from BVALID to BREADY	16
AR_MAX_WAITS	Maximum number of wait cycles from ARVALID to ARREADY	16
R_MAX_WAITS	Maximum number of wait cycles from RVALID to RREADY	16
NARROW_TRANSFER	Indicate if Narrow Transfer is supported or not	1
WR_OUT_OF_ORDER	Indicates whether write out-of-order response is supported or not	1
RD_OUT_OF_ORDER	Indicates whether read out-of-order data is supported or not	1
RD_INTERLEAVE	Indicates whether read interleave data is supported or not	1
WDATA_ADVANCE	Indicates whether write data can be issued earlier than write address or not	1
WDATA_WIDTH	Indicates the bit width of the write data bus	128
RDATA_WIDTH	Indicates the bit width of the read data	128
ADDR_WIDTH	Indicates address bus bit width	64
ID_WIDTH	Indicates the bit width of the ID	8
AWUSER_WIDTH	Indicates the width of the User AW Sideband field	32
WUSER_WIDTH	Indicates the width of the User W Sideband field	32
BUSER_WIDTH	Indicates the width of the User B Sideband field	32
ARUSER_WIDTH	Indicates the width of the User AR Sideband field	32

Table 4-1 AXI4 AIP Configuration Parameters

Parameter	Description	Default Value
RUSER_WIDTH	Indicates the width of the User R Sideband field	32
LEN_WIDTH	Indicates the width of the AWLEN and ARLEN signals	8
RD_ALLOW_SLVERR	1 indicates to allow SLVERR and 0 indicates not to allow SLVERR for RRESP signal.	1
RD_ALLOW_DECERR	1 indicates to allow DECERR and 0 indicates not to allow DECERR for RRESP signal.	1
WR_ALLOW_SLVERR	1 indicates to allow SLVERR and 0 indicates not to allow SLVERR for BRESP signal.	1
WR_ALLOW_DECERR	1 indicates to allow DECERR and 0 indicates not to allow DECERR for BRESP signal.	1
ADDR_RANGE	Indicates the number of Slave address ranges.	1
MIN_ADDR	Indicates lower address of each Slave address range.	All bits 0
MAX_ADDR	Indicates upper address of each Slave address range.	All bits 1

Table 4-2 ACE Configuration Parameters

Parameter	Description	Default Value	Available in Agent
AGENT_TYPE	<p>Specifies the agent type.</p> <p>MASTER:</p> <ul style="list-style-type: none"> Instantiates the AIP as a master to check the master input behavior and to constraint the master output. By default: ENABLE_ASSERT=1 and ENABLE_ASSUME=1 <p>SLAVE:</p> <ul style="list-style-type: none"> Instantiates the AIP as a slave to check the slave input behavior and to constraint the slave output. By default: ENABLE_ASSERT=1 and ENABLE_ASSUME=1 <p>MONITOR:</p>	MASTER	Y

Parameter	Description	Default Value	Available in Agent
	<ul style="list-style-type: none"> Instantiates the AIP as a monitor to check the DUT slave behavior and the DUT master behavior. By default: <code>ENABLE_ASSERT=1</code> and <code>ENABLE_ASSUME=0</code> <p>CONSTRAINT:</p>		
	<ul style="list-style-type: none"> Instantiates the AIP to constraint the DUT slave input and the DUT master input. By default: <code>ENABLE_ASSERT=0</code> and <code>ENABLE_ASSUME=1</code> 		
ENABLE_ASSERT	Indicates if assertions are instantiated or not	1	Y
ENABLE_ASSUME	Indicates if assumptions are instantiated or not	1	Y
ENABLE_COVER	Indicates if cover properties are instantiated or not	1	Y
CHECK_FORMAL	Set this to 1/0 for selecting instantiation of formal optimized/unoptimized properties	1	Y
CHECK_PARAMETERS	Indicates check if parameter setting or not	0	Y
BARR_DEPTH	Indicates number of maximum outstanding Barrier transactions	4	Y
CACHE_LINE_BYTES	Indicates size of cache line in bytes	64	Y
CDATA_WIDTH	Indicates Snoop Data bit width	128	N
AC_MAX_WAITS	Maximum number of wait cycles from ACVALID to ACREADY	16	N
CR_MAX_WAITS	Maximum number of wait cycles from CRVALID to CRREADY	16	N
CD_MAX_WAITS	Maximum number of wait cycles from CDVALID to CDREADY	16	N

4.2 The ACE AIP Interface Ports

This section describes all the interface signals of ACE AIP.

4.2.1 AXI4 Interface Ports

Table 4-3 AXI4 AIP Interface Ports

Signal Name	Signal Width	Description	
		Source	Destination
aclk	1	Input clock from the system	
aresetn	1	Reset input from the system	
awid	ID_WIDTH	Master	Slave
awaddr	ADDR_WIDTH	Master	Slave
awlen	LEN_WIDTH	Master	Slave
awsiz	3	Master	Slave
awburst	2	Master	Slave
awcache	4	Master	Slave
awprot	3	Master	Slave
awlock	1	Master	Slave
awqos	4	Master	Slave
awregion	4	Master	Slave
awuser	AWUSER_WIDTH	Master	Slave
awvalid	1	Master	Slave
awready	1	Master	Slave
wdata	WDATA_WIDTH	Master	Slave
wstrb	STRB_WIDTH	Master	Slave
wuser	WUSER_WIDTH	Master	Slave
wlast	1	Master	Slave
wvalid	1	Master	Slave
wready	1	Slave	Master
bid	ID_WIDTH	Slave	Master
bresp	2	Slave	Master
buser	BUSER_WIDTH	Slave	Master
bvalid	1	Slave	Master
bready	1	Master	Slave

Table 4-3 AXI4 AIP Interface Ports

Signal Name	Signal Width	Description	
arid	ID_WIDTH	Master	Slave
araddr	ADDR_WIDTH	Master	Slave
arlen	LEN_WIDTH	Master	Slave
arsize	3	Master	Slave
arburst	2	Master	Slave
arcache	4	Master	Slave
arprot	3	Master	Slave
arlock	1	Master	Slave
arqos	4	Master	Slave
arregion	4	Master	Slave
aruser	ARUSER_WIDTH	Master	Slave
arvalid	1	Master	Slave
arready	1	Slave	Master
rid	ID_WIDTH	Slave	Master
rdata	RDATA_WIDTH	Slave	Master
rresp	4	Slave	Master
ruser	RUSER_WIDTH	Slave	Master
rlast	1	Slave	Master
rvalid	1	Slave	Master
rready	1	Master	Slave
cactive	1	Master	Slave
csysreq	1	Peripheral device to system	
csysack	1	Peripheral device to system	

4.2.2 ACE Interface Ports

Table 4-4 ACE related Interface Ports

Signal Name	Signal Width	Description		
awbar	2	Master	Slave	Y
awdomain	2	Master	Slave	Y
awsnoop	3	Master	Slave	Y
arbar	2	Master	Slave	Y
ardomain	2	Master	Slave	Y
arsnoop	4	Master	Slave	Y
acaddr	ADDR_WIDTH	Slave	Master	N
acprot	3	Slave	Master	N
acsnoop	4	Slave	Master	N
acvalid	1	Interconnect	Master	N
acready	1	Master	Slave	N
crresp	5	Master	Slave	N
crvalid	1	Master	Slave	N
crready	1	Slave	Master	N
cdvalid	1	Master	Slave	N
cdready	1	Slave	Master	N
cdlast	1	Master	Slave	N
cddata	CDATA_WIDTH	Master	Slave	N

4.3 The ACE AIP Properties

4.3.1 Write Address Channel Properties

Table 4-5 Write Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/Slave	Parameter Required	Property Description
1	snps_ace_awcache_device	ERROR	Spec IHI 0022E Ref: table C3-3, page C3-163	MASTER	ENABLE_ASS ERT==1	When AWVALID is HIGH, if AWCACHE[1] is LOW, non-modifiable, device, AWDOMAIN must be 'b11, system shareable.
2	snps_ace_awcache_no_system	ERROR	Spec IHI 0022E Ref: table C3-3, page C3-163	MASTER	ENABLE_ASS ERT==1	When AWVALID is HIGH, if Allocate and Other Allocate bits of AWCACHE are not zero, then AWDOMAIN must not be System domain.
3	snps_ace_aw_incr_align_to_cache_line	ERROR	Spec IHI 0022E Ref: table C3-11 , page C3-169	MASTER	ENABLE_ASS ERT==1	WriteLineUnique and Evict transactions of type INCR must be aligned to the cache line size.
4	snps_ace_aw_wrap_align	ERROR	Spec IHI 0022E Ref: table C3-11 , page C3-169	SLAVE	ENABLE_ASS ERT==1	WriteLineUnique and Evict transactions of type WRAP must be aligned to the width of the bus.
5	snps_ace_aw_wlu_we_cache_line	ERROR	Spec IHI 0022E Ref: section C3.1.5, C6.7.2 , page C3-168, C6-249	MASTER	ENABLE_ASS ERT==1	WriteLine Unique and Evict transactions are required to be a full cache line size.
6	snps_ace_aw_shareable_lock	ERROR	Spec IHI 0022E Ref: table:C3- 11, C3-12,C3- 13, page C3- 169,170	MASTER	ENABLE_ASS ERT==1	All shareable accesses, as indicated by AxDOMAIN = Inner Shareable or Outer Shareable, must use AxLOCK = 0.
7	snps_ace_aw_prev_bar_dvm_id	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	A normal write transaction that shares its ID with an outstanding write barrier must not be issued.

Table 4-5 Write Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/Slave	Parameter Required	Property Description
8	snps_ace_no_wr_chrst_when_upd	ERROR	Spec IHI 0022E Ref: section C4.8.7, page C4-212	MASTER	ENABLE_ASS ERT==1	Master must complete any outstanding WriteBack, WriteClean, or WriteEvict transactions before issuing a WriteUnique or WriteLineUnique transaction.
9	snps_ace_awdomain_stable	ERROR	Spec IHI 0022E Ref: section A3.2.1, page A3-39	MASTER	ENABLE_ASS ERT==1	AWDOMAIN must remain stable when AWVALID is asserted and AWREADY low.
10	snps_ace_awsnoop_stable	ERROR	Spec IHI 0022E Ref: section A3.2.1, page A3-39	MASTER	ENABLE_ASS ERT==1	AWSNOOP must remain stable when AWVALID is asserted and AWREADY low.
11	snps_ace_awbar_stable	ERROR	Spec IHI 0022E Ref: section A3.2.1, page A3-39	MASTER	ENABLE_ASS ERT==1	AWBAR must remain stable when AWVALID is asserted and AWREADY low.
12	snps_ace_awdomain_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When AWVALID is high, a value of X on AWDOMAIN is not permitted.
13	snps_ace_awsnoop_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When AWVALID is high, a value of X on AWSNOOP is not permitted.
14	snps_ace_awbar_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When AWVALID is high, a value of X on AWBAR is not permitted.

Table 4-5 Write Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/Slave	Parameter Required	Property Description
15	snps_ace_\$Maint_while_Read	ERROR	IHI 0022E Ref: section C7.2.1 on page C7-253	MASTER	ENABLE_ASSERT==1	Cache maintenance operations cannot be performed if the master has outstanding shareable read transactions to the same cache line.
16	snps_ace_AW_while_\$Maint	ERROR	IHI 0022E Ref: section C7.2.1 on page C7-253	MASTER	ENABLE_ASSERT==1	A master must not issue a shareable write transaction to a cache line that has outstanding cache maintenance operation.
17	snps_ace_W_after_R_hazard	ERROR	IHI 0022E Ref: section C7.2.1 on page C7-253	MASTER	ENABLE_ASSERT==1	A master should not issue a write transaction to a memory region it is currently reading from.
18	snps_ace_W_after_W_hazard	ERROR	IHI 0022E Ref: section C7.2.1 on page C7-253	MASTER	ENABLE_ASSERT==1	A master should not issue a write transaction to a memory region it is currently writing to.

4.3.2 Write Data Channel Properties

Table 4-6 Write Data Channel Properties

	Property Name	Error Kind	Spec Reference	Master/Slave	Parameter Required	Property Description
1	snps_ace_wlu_no_sprs_strb	ERROR	Spec IHI 0022E Ref: section C3.1.6, page C3-168	SLAVE	ENABLE_ASSERT==1	WriteLineUnique transactions are not permitted to have sparse strobes.

4.3.3 Write Response Channel Properties

Table 4-7 Write Response Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
1	snps_ace_bar_bresp_okay	ERROR	Spec: table C8-2 on page C8-260	SLAVE	ENABLE_ASS ERT==1	Barrier responses must be 'b00.
2	snps_ace_bresp_exok_is4_wns	ERROR	Spec: section C8-2 on page C8-260	SLAVE	ENABLE_ASS ERT==1	EXOKAY response is only permitted for WRITENOSNOOP transaction.

4.3.4 Read Address Channel Properties

Table 4-8 Read Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
1	snps_ace_lite_arsnoop	ERROR	Spec IHI 0022E Ref: table C3-8, page C3-166	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==1	When ARVALID is HIGH, ARSNOOP must be one of the non-reserved values.
2	snps_ace_arcache_device	ERROR	Spec IHI 0022E Ref: table C3-3, page C3-163	MASTER	ENABLE_ASS ERT==1	When ARVALID is HIGH, if ARCACHE[1] is LOW, non-modifiable, device, ARDOMAIN must be 'b11, system shareable.
3	snps_ace_arcache_no_system	ERROR	Spec IHI 0022E Ref: table C3-3, page C3-163	MASTER	ENABLE_ASS ERT==1	When ARVALID is HIGH, if Allocate and Other Allocate bits of ARCACHE are not zero, then ARDOMAIN must not be System domain.
4	snps_ace_ardomain_inner_outer	ERROR	Spec IHI 0022E Ref: table C3-11, page C3-169	MASTER	ENABLE_ASS ERT==1	ReadClean, ReadNotSharedDirty, ReadShared, ReadUnique, CleanUnique, MakeUnique transactions must be to the inner or outer domain.

Table 4-8 Read Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
5	snps_ace_ar_incr_align_to_cacheline	ERROR	Spec IHI 0022E Ref: table C3-11, page C3-169	MASTER	ENABLE_ASS ERT==1	ReadShared, ReadClean, ReadNotSharedDirty, Readunique, CleanInvalid, CleanShared, CleanUnique, MakeUnique, and MakeInvalid transactions of type INCR must be aligned to the cache line size.
6	snps_ace_ar_specific_ctrl	ERROR	Spec IHI 0022E Ref: table C3-11, C3-12, C3-13, page C3-169, C3-170	MASTER	ENABLE_ASS ERT==1	ReadShared, ReadClean, ReadNotSharedDirty, Readunique, CleanInvalid, CleanShared, CleanUnique, MakeUnique, and MakeInvalid transactions must have specific values for ARBURST, ARBAR & ARCACHE.
7	snps_ace_ar_shareable_lock	ERROR	Spec IHI 0022E Ref: table C3-11, C3-12, page C3-169, C3-170	MASTER	ENABLE_ASS ERT==1	ReadNotSharedDirty, ReadOnce, Readunique, CleanInvalid, CleanShared, MakeUnique, and MakeInvalid must use ARLOCK = 1'b0.
8	snps_ace_ar_prev_bar_dvm_id	ERROR	Spec IHI 0022E Ref: section C8.4.1, C12.3.5, page C8-264, C12-293	MASTER	ENABLE_ASS ERT==1	New RD transaction ID cannot coincide with pre-existing ID of Barrier or DVM transaction.
9	snps_ace_ardomain_stable	ERROR	IHI 0022E Ref: section A3.2.1 on page A3-39	MASTER	ENABLE_ASS ERT==1	ARDOMAIN must remain stable when ARVALID is asserted and ARREADY low.
10	snps_ace_arsnoop_stable	ERROR	IHI 0022E Ref: section A3.2.1 on page A3-39	MASTER	ENABLE_ASS ERT==1	ARSNOOP must remain stable when ARVALID is asserted and ARREADY low.

Table 4-8 Read Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
11	snps_ace_arbar_stable	ERROR	IHI 0022E Ref: section A3.2.1 on page A3-39	MASTER	ENABLE_ASS ERT==1	ARBAR must remain stable when ARVALID is asserted and ARREADY low.
12	snps_ace_arburst_wrap_align	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-11 on page C3-169	MASTER	ENABLE_ASS ERT==1	The address must be aligned to ARSIZE, which is equal to the data bus width.
13	snps_ace_rd_wrap_length	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-11 on page C3-169	MASTER	ENABLE_ASS ERT==1	For Cache line size transaction, the burst length must be 1, 2, 4, 8 or 16 transfers.
14	snps_ace_ardomain_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When ARVALID is high, a value of X on ARDOMAIN is not permitted.
15	snps_ace_arsnoop_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When ARVALID is high, a value of X on ARSNOOP is not permitted.
16	snps_ace_arbar_x	ERROR	Spec IHI 0022E Ref: section A3.2.2, page A3-40	MASTER	ENABLE_ASS ERT==1	When ARVALID is high, a value of X on ARBAR is not permitted.
17	snps_ace_\$Maint_while_Write	ERROR	Spec IHI 0022E Ref: section C7.2.1	MASTER	ENABLE_ASS ERT==1	Cache maintenance operations cannot be performed if the master has outstanding shareable write transactions to the same cache line.
18	snps_ace_AR_while_\$Maint	ERROR	Spec IHI 0022E Ref: section C7.2.1	MASTER	ENABLE_ASS ERT==1	A master must not issue a shareable read transaction to a cache line that has outstanding cache maintenance operation.

Table 4-8 Read Address Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
19	snps_ace_R_after_W_hazard	ERROR		MASTER	ENABLE_ASS ERT==1	A master should not issue a write transaction to a memory region it is currently reading from.
20	snps_ace_ar_shareable_cache_line	ERROR	Spec IHI 0022E Ref: table C3-11, pg. C3-169	MASTER	ENABLE_ASS ERT==1	ReadShared, ReadClean, ReadNotSharedDirty, Readunique, CleanInvalid, CleanShared, CleanUnique, MakeUnique, and MakeInvalid transactions must be to a full cache line size.

4.3.5 Read Data Channel Properties

Table 4-9 Read Data Channel Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
1	snps_ace_rresp_permitted_exokay	ERROR	Spec: section C3.2.1 on page C3-172	MASTER	ENABLE_ASS ERT==1	The EXOKAY response is only permitted for ReadNoSnoop, ReadClean, ReadShared, and CleanUnique transactions.
2	snps_ace_rresp_barrier_zero	ERROR	Spec: table C8-2 on page C8-260	MASTER	ENABLE_ASS ERT==1	A barrier response must use RRESP = 'b0000.
3	snps_ace_rd_dataless_rlast	ERROR	Spec: section C4.1.1 on page C4-190	MASTER	ENABLE_ASS ERT==1	A slave must return exactly one data beat for Make or Clean transactions.

4.3.6 Barrier Properties

Table 4-10 Barrier Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
1	snps_ace_barr_ctl_newr_olwr	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	The ID, DOMAIN, BAR and PROT values of read barrier must be same as corresponding write barrier.
2	snps_ace_barr_ctl_neww_olwr	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	The ID, DOMAIN, BAR and PROT values of write barrier must be same as corresponding read barrier.
3	snps_ace_barr_ctl_neww_newr	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	The ID, DOMAIN, BAR and PROT values of write barrier must be same as corresponding read barrier.
4	snps_ace_arbar_id_unseen	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	The ID of a read barrier must not be the same as any outstanding non-barrier transaction.
5	snps_ace_awbar_id_unseen	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	The ID of a write barrier must not be the same as any outstanding non-barrier transaction.
6	snps_ace_rd_barrier_allowed_araddr	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARADDR = All zeros.
7	snps_ace_rd_barrier_allowed_arlen	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARLEN = All zeros.
8	snps_ace_rd_barrier_allowed_arburst	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARBURST = 1.
9	snps_ace_rd_barrier_allowed_arsize	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARSIZE = Data bus width.

Table 4-10 Barrier Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
10	snps_ace_rd_barrier_allowed_arsnoop	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARSNOOP = All zeros.
11	snps_ace_rd_barrier_allowed_arcache	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARCACHE = 'b0010 or 'b0011
12	snps_ace_rd_barrier_allowed_arlock	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: ARLOCK= All zeros.
13	snps_ace_wr_barrier_allowed_awaddr	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWADDR = All zeros.
14	snps_ace_wr_barrier_allowed_awlen	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWLEN = 8'b0.
15	snps_ace_wr_barrier_allowed_awburst	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWBURST = 2'b01 - INCR.
16	snps_ace_wr_barrier_allowed_awsiz	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWSIZE = Data bus width.
17	snps_ace_wr_barrier_allowed_awsnoop	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWSNOOP = All zeros.
18	snps_ace_wr_barrier_allowed_awcache	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWCACHE = 'b0010 or 'b0011.

Table 4-10 Barrier Properties

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
19	snps_ace_wr_barrier_allowed_awlock	ERROR	Spec IHI 0022E Ref: table C3-14, page C3-171	MASTER	ENABLE_ASS ERT==1	Barrier transactions must have the following payload values: AWLOCK = 'b0.
20	snps_ace_max_rbar_num	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	A master interface must not issue more than 256 outstanding read barrier transactions.
21	snps_ace_max_wbar_num	ERROR	Spec IHI 0022E Ref: section C8.4.1, page C8-264	MASTER	ENABLE_ASS ERT==1	A master interface must not issue more than 256 outstanding write barrier transactions.
22	snps_ace_rd_barrier_zeros_aruser	ERROR	IHI 0022E Ref: section C8.2.3 on page C8-260	MASTER	ENABLE_ASS ERT==1	READ Barrier transactions recommended to have all zeros for ARUSER.
23	snps_ace_wr_barrier_zeros_awuser	ERROR	IHI 0022E Ref: section C8.2.3 on page C8-260	MASTER	ENABLE_ASS ERT==1	Write Barrier transactions recommended to have all zeros for AWUSER.
24	snps_ace_rd_barrier_rlast	ERROR	Spec: section table C8-2 on page C8-260	SLAVE	ENABLE_ASS ERT==1	A read barrier response must always have RLAST asserted.

4.3.7 ACE Specific Properties (Not applicable for ACE-Lite)

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
1	snps_ace_awsnoop	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-8 on page C3-166	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When AWVALID is HIGH, AWSNOOP must be one of the non-reserved values."
2	snps_ace_arsnoop	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-7 on page C3-165	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When ARVALID is HIGH, ARSNOOP must be one of the non-reserved values."
3	snps_ace_awdomain_inner_outer	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-11 on page C3-169	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"All transactions except CleanShared, CleanInvalid, MakeInvalid, or WriteEvict, the domain must be Inner Shareable or Outer Shareable."
4	snps_ace_ardomain_inner_outer	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-11 on page C3-169	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"All transactions except CleanShared, CleanInvalid, MakeInvalid, or WriteEvict, the domain must be Inner Shareable or Outer Shareable."
5	snps_ace_awdomain_not_system	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-13 on page C3-170	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"For a WriteBack or a WriteClean, AWDOMAIN must not be system shareable."
6	snps_ace_ardomain_not_system	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-11 on page C3-169	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"CleanShared, CleanInvalid, MakeInvalid, and WriteEvict transactions, the domain must be Non-shareable, Inner Shareable or Outer Shareable."
7	snps_ace_crresp_pass_dirty_data_xfer	ERROR	IHI 0022E Ref: section C3.7 on page C3-182	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When HIGH, it indicates that before the snoop process, the cache line was held in a Dirty state and the responsibility for writing the cache line back to main memory is being passed to the initiating master or the interconnect."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
8	snps_ace_rucimi_crr esp	ERROR	IHI 0022E Ref: section C3.7 on page C3-183	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"For ReadUnique, CleanInvalid and MakeInvalid transactions, the cache line in the snoop cache must be invalidated and the IsShared response must be LOW."
9	snps_ace_crvalid_stable	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Once CRVALID is asserted, it must remain asserted until CRREADY is high."
10	snps_ace_crresp_stable	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Once CRVALID is asserted, CRRESP must remain stable until CRREADY is high."
11	snps_ace_cr_has_ac	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	A snoop response is required on the snoop response channel for each snoop address that is presented to a cached master on the snoop address channel.
12	snps_ace_cdvalid_stable	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Once CDVALID is asserted, CDVALID must remain asserted until CDREADY is high."
13	snps_ace_cddata_stable	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Once CDVALID is asserted, CDDATA must remain stable until CDREADY is high."
14	snps_ace_cdlast_stable	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Once CDVALID is asserted, CDLAST must remain stable until CDREADY is high."
15	snps_ace_data_num_cdlast	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	The CDLAST signal must be asserted during the final data transfer associated with a snoop transaction.
16	snps_ace_cd_has_a c	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Snoop data is not required for every snoop transactions, it is only provided for a snoop transaction that has a snoop response with the Data Transfer bit asserted."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
17	snps_ace_wb_wc_in_cr_cacheline	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-13 on page C3-170	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"For a WriteBack or a WriteClean, the transaction must not cross a cacheline boundary. The final beat of a transaction must fall within the same cache line as the first. "
18	snps_ace_wb_wc_wr_ap_cacheline	ERROR	IHI 0022E Ref: section C3.1.6 Table C3-13 on page C3-170	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"For a WriteBack or a WriteClean, the transaction must not cross a cacheline boundary. The total number of bytes must not exceed the cache line size in bytes."
19	snps_ace_no_wr_up_d_when_chrnt	ERROR	IHI 0022E Ref: section C4.8.7 on page C4-212	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Master must not issue additional WriteBack, WriteClean, or WriteEvict transactions until all outstanding WriteUnique or WriteLineUnique transactions are completed."
20	snps_ace_crresp_while_wbwc	ERROR	IHI 0022E Ref: section C5.2.3 on page C5-225	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	The CRRESP of a snoop for a cacheline with an ongoing WriteBack or WriteClean must have IsShared=1 and PassDirty=0.
21	snps_ace_same_crresp_dvm_multi	ERROR	IHI 0022E Ref: section C12.3.4 on page C12-293	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	It is required that the response given to all parts of a multiple transaction DVM message are the same.
22	snps_ace_dvm_crresp	ERROR	IHI 0022E Ref: section C12.3.4 on page C12-293	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When a component receives a DVM transactions, it must respond as follows: If the component can perform the requested action, it must respond by setting CRRESP to 0b00000. If the component is unable to perform the requested action, it must respond by setting CRRESP to 0b00010."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
23	snps_ace_dvm_sch_crresp	ERROR	IHI 0022E Ref: section C12.3.4 on page C12-293	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	A component is not permitted to set CRRESP to 0b00010 in response to a DVM Sync or a DVM Complete.
24	snps_ace_dvm_type_s_araddr_14_12_valid	ERROR	IHI 0022E Ref: C12.7 Table C12-9 on page C12-299	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	DVM Message must have valid Message Type in araddr[14:12].
25	snps_ace_dvm_araddr_resvd_1	ERROR	IHI 0022E Ref: C12.7 Table C12-9 on page C12-300	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Message, ARADDR[1] is Reserved, SBZ."
26	snps_ace_dvm_araddr_resvd_2	ERROR	IHI 0022E Ref: C12.7 Table C12-10 on page C12-300	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	ARADDR[2:0] is SBZ if an additional transaction is used to convey address information.
27	snps_ace_dvm_araddr_resvd_3	ERROR	IHI 0022E Ref: C12.9.5 Table C12-20 on page C12-308	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Sync message, ARADDR[15] should be 1 and ARADDR[11:0] are 0."
28	snps_ace_dvm_araddr_resvd_4	ERROR	IHI 0022E Ref: C12.7 Table C12-9 on page C12-299	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Message except for DVM Sync, ARADDR[15] should be 0."
29	snps_ace_dvm_tlb_inval_araddr	ERROR	IHI 0022E Ref: C12.9.1 Table C12-12 on page C12-302	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When TLB Invalidate, ARADDR[7] and ARADDR[1] should be 0."
30	snps_ace_dvm_bp_inval_araddr	ERROR	IHI 0022E Ref: C12.9.2 Table C12-14 on page C12-305	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Branch Predictor Invalidate, ARADDR[11:1] should be 0."
31	snps_ace_dvm_phyl_inv_araddr	ERROR	IHI 0022E Ref: section C12.9.3 Table C12-16 on page C12-306	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Physical Instruction Cache Invalidate, ARADDR[15], ARADDR[11:10], ARADDR[7] and ARADDR[4:1] should be 0."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
32	snps_ace_dvm_phyl\$ _inv_araddr	ERROR	IHI 0022E Ref: section C12.9.3 Table C12-16 on page C12-306	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Physical Instruction Cache Invalidate, ARADDR[15], ARADDR[11:10], ARADDR[7] and ARADDR[4:1] should be 0."
33	snps_ace_dvm_allowed_ arlen	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"The burst length must be 1, that is ARLEN[7:0] must be 0b00000000 for DVM transaction."
34	snps_ace_dvm_allowed_ arburst	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	Burst type must be INCR for DVM transaction.
35	snps_ace_dvm_allowed_ arsize	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	The number of bytes in a transfer must be equal to the data bus width for DVM transaction.
36	snps_ace_dvm_allowed_ arcache	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Must be Modifiable and Non-cacheable, that is ARCACHE[3:0] must be 0b0010 for DVM transaction."
37	snps_ace_dvm_allowed_ arlock	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Must be a normal access, that is ARLOCK must be 0 for DVM transaction."
38	snps_ace_dvm_allowed_ ardomain	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	The domain must be Inner Shareable or Outer Shareable for DVM transaction.
39	snps_ace_dvm_allowed_ arbar	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	"Must be a normal access, that is ARBAR[0] must be 0 for DVM transaction."
40	snps_ace_dvm_comple te_araddr_zero	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	Must be zero for DVM Complete.

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
41	snps_ace_dvm_complete_ar	ERROR	IHI 0022E Ref: section C12.3.2 on page C12-292	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	A DVM Complete on the read address channel must only be issued after the handshake of the associated DVM Sync on the snoop address channel of the same master.
42	snps_ace_dvm_multi_consec_ar	ERROR	IHI 0022E Ref: section C12.3.3 on page C12-292	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	Multi-part of DVM messages are always sent as successive transactions and no other transaction can be interposed between them.
43	snps_ace_dvm_no_shared_ids	ERROR	IHI 0022E Ref: section C12.3.5 on page C12-293	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	DVM transactions must not use AXI ID values that are used by non-DVM transactions on the AR channel.
44	snps_ace_dvm_same_id_multi	ERROR	IHI 0022E Ref: section C12.3.3 on page C12-292	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	Each transaction of a multi-part DVM message must use the same AXID.
45	snps_ace_mst_max_outstanding_dvm_sync	ERROR	IHI 0022E Ref: section C12.3.2 on page C12-292	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	Only one outstanding DVM message that requires completion is allowed.
46	snps_ace_rd_dvm_rlast	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0	The burst length must be 1 for DVM transaction. Therefore RLAST is HIGH.
47	snps_ace_full_valid_excl	ERROR	IHI 0022E Ref: section C9.6 on page C9-278	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && EXCL_DEPTH>=1	Non Valid Exclusive Load or Store Command.
48	snps_ace_no_excl_in_xstore	ERROR	IHI 0022E Ref: section C9.2.3 on page C9-272	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && EXCL_DEPTH>=2	A master must not permit an Exclusive Store transaction to be in progress at the same time as any transaction that registers that it is performing an Exclusive sequence.

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
49	snps_ace_no_xstore_in_excl	ERROR	IHI 0022E Ref: section C9.2.3 on page C9-272	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && EXCL_DEPTH>= 1	A master must not permit an Exclusive Store transaction to be in progress at the same time as any transaction that registers that it is performing an Exclusive sequence.
50	snps_ace_acready_max_waits	WARNIN G	Recommendatio n	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_REC OMMEND==1 && CONFIG_WAITS ==1	ACREADY should be asserted within AC_MAX_WAITS cycles of ACVALID being asserted.
51	snps_ace_acready_maxouts	WARNIN G	Maximum number of outstanding transactions.	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_MAXO UTS==1	Should not accept Snoop requests more than SN_MAX_BURSTS.
52	snps_ace_acready_x	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on ACREADY is not permitted.
53	snps_ace_crvalid_x	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CRVALID is not permitted.
54	snps_ace_crresp_x	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CRRESP is not permitted.
55	snps_ace_cdvalid_x	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CDVALID is not permitted.
56	snps_ace_cdlast_x	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CDLAST is not permitted.

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
57	snps_ace_cddata_x	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CDDATA is not permitted.
58	snps_ace_rack_x	ERROR	IHI 0022E Ref: section A3.2.2 on page A3-40	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on RACK is not permitted.
59	snps_ace_wack_x	ERROR	IHI 0022E Ref: section A3.2.2 on page A3-40	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on WACK is not permitted.
60	snps_ace_rack_x	ERROR	IHI 0022E Ref: section A3.2.2 on page A3-40	MASTER	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on RACK is not permitted.
61	snps_ace_same_rresp_dvm_multi	ERROR	IHI 0022E Ref: section C12.3.4 on page C12-293	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	It is required that the response given to all parts of a multiple transaction DVM message are the same.
62	snps_ace_rresp_const_each_beat	ERROR	IHI 0022E Ref: section C3.2.1 on page C3-172	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	RRESP[3:2] are required to be constant for all data transfers within a burst.
63	snps_ace_rresp_isshared_passdirty_const	ERROR	IHI 0022E Ref: section C3.2.1 on page C3-172	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	CRRESP[3:2] Isshared and PassDirty bit constant for all beats in a RD response burst.
64	snps_ace_rresp_not_dirty	ERROR	IHI 0022E Ref: section C3.2.1 on page C3-172	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"A slave must not give a PassDirty (RESP[2] = 1'b1) response to a ReadNoSnoop, ReadOnce, ReadClean, CleanUnique, CleanInvalid, MakeUnique, MakeInvalid, or CleanShared transaction."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
65	snps_ace_rresp_not_shared	ERROR	IHI 0022E Ref: section C3.2.1 on page C3-172	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"A slave must not give an IsShared (RESP[3] = 'b1) response to a ReadNoSnoop, ReadUnique, CleanUnique, CleanInvalid, MakeInvalid, or MakeUnique transaction."
66	snps_ace_no_ac_in_nonsh_read	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"At any point in time, different agents should have a consistent view of the shareability of a memory region. A snoop has been received to a region that has an outstanding non-shareable read."
67	snps_ace_no_ac_in_bresp_wack	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"At any point in time, different agents should have a consistent view of the shareability of a memory region. A snoop has been received to a region that has an outstanding non-shareable write."
68	snps_ace_no_ac_in_bresp_wack_wuwlu	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If the interconnect provides a master with a response to a WriteUnique or WriteLineUnique transaction, it must not send the same master a snoop transaction to the same cache line until it has received and acknowledgement of the transaction response on WACK."
69	snps_ace_no_ac_until_rack	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If the interconnect provides a master with a response to a Coherent or Cache Maintenance transaction, it must not send that master a snoop transaction to the same cache line before it has received the associated RACK or WACK response from that master."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
70	snps_ace_bresp_In_Snoop	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If the interconnect sends a snoop transaction to a master, it must not provide the same master with a response to a WriteUnique or WriteLineUnique transaction to the same cache line until it has received a snoop response on CRRESP to the snoop transaction."
71	snps_ace_bresp_No n_Shar_In_Snoop	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"At any point in time, different agents should have a consistent view of the shareability of a memory region. A write response to a non-shareable transaction has been sent to the same cache line as an active snoop."
72	snps_ace_rresp_In_Snoop	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If the interconnect sends a snoop transaction to a master, it must not provide the same master with a response to a transaction to the same cache line, until it has received a snoop response on CRRESP to the snoop transaction."
73	snps_ace_rresp_Non_Shar_In_Snoop	ERROR	IHI 0022E Ref: section C6.2 Figure C6-1 on page C6-235	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"At any point in time, different agents should have a consistent view of the shareability of a memory region. A read response to a non-shareable transaction has been sent to the same cache line as an active snoop."
74	snps_ace_acsnoop_I egal	ERROR	IHI 0022E Ref: section C3.6.2 Table C3-20 on page C3-179 and C3-180	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When ACVALID is asserted, ACSNOOP should have valid encodings."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
75	snps_ace_acaddr_aligned	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"ACADDR must be aligned to the data transfer size, which is determined by the width of the snoop data bus in bytes."
76	snps_ace_acvalid_reset	ERROR	IHI 0022E Ref: section C2.3.3 on page C2-159	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"During reset, an interconnect must drive ACVALID LOW."
77	snps_ace_acvalid_stable	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When ACVALID is asserted, it must remain asserted until ACREADY is asserted."
78	snps_ace_acsnoop_stable	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	When the ACVALID signal is asserted ACSNOOP must not change until ACREADY is asserted by the master.
79	snps_ace_acaddr_stable	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	When the ACVALID signal is asserted ACADDR must not change until ACREADY is asserted by the master.
80	snps_ace_acprot_stable	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	When the ACVALID signal is asserted ACPROT must not change until ACREADY is asserted by the master.
81	snps_ace_rack_per_rlast	ERROR	IHI 0022E Ref: section C3.3 on page C3-175	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"The RACK signal must not be asserted before the cycle after the completion of the associated RVALID/RREADY handshake of the last read data channel transfer, as indicated by RLAST."
82	snps_ace_wack_per_bresp	ERROR	IHI 0022E Ref: section C3.5 on page C3-177	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	The WACK signal must not be asserted before the cycle after the completion of the associated BVALID/BREADY handshake.
83	snps_ace_rresp_32No11_RNSD	ERROR	IHI 0022E Ref: section C5.3.2 Table C5-8 on page C5-228	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If master is initiating ReadNotSharedDirty, then RRESP[3:2] can't have 2'b11 value."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
84	snps_ace_dvm_type s_acaddr_14_12_vali d	ERROR	IHI 0022E Ref: C12.7 Table C12-9 on page C12-299	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	DVM Message must have valid Message Type in acaddr[14:12].
85	snps_ace_dvm_acad dr_resvd_1	ERROR	IHI 0022E Ref: section C12.7 Table C12-9 on page C12-300	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Message, ACADDR[1] is Reserved, SBZ."
86	snps_ace_dvm_acad dr_resvd_2	ERROR	IHI 0022E Ref: section C12.7 Table C12-10 on page C12-300	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	ACADDR[2:0] is SBZ if an additional transaction is used to convey address information.
87	snps_ace_dvm_acad dr_resvd_3	ERROR	IHI 0022E Ref: section C12.9.5 Table C12-20 on page C12-308	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Sync message, ACADDR[15] must be 1 and ACADDR[11:0] are 0."
88	snps_ace_dvm_acad dr_resvd_4	ERROR	IHI 0022E Ref: section C12.7 Table C12-9 on page C12-299	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When DVM Message except for DVM Sync, ACADDR[15] must be 0."
89	snps_ace_dvm_tlb_i nv_acaddr	ERROR	IHI 0022E Ref: section C12.9.1 Table C12-12 on page C12-302	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When TLB Invalidate, ACADDR[7] and ACADDR[1] must be 0."
90	snps_ace_dvm_bp_i nv_acaddr	ERROR	IHI 0022E Ref: section C12.9.2 Table C12-14 on page C12-305	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Branch Predictor Invalidate, ACADDR[11:1] must be 0."
91	snps_ace_dvm_phyl \$_inv_acaddr	ERROR	IHI 0022E Ref: section C12.9.3 Table C12-16 on page C12-306	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Physical Instruction Cache Invalidate, ACADDR[15], ACADDR[11:10], ACADDR[7] and ACADDR[4:1] should be 0."
92	snps_ace_dvm_virl\$ _inv_acaddr	ERROR	IHI 0022E Ref: section C12.9.4 Table C12-18 on page C12-307	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"When Virtual Instruction Cache Invalidate, ACADDR[15], ACADDR[7] and ACADDR[4:1] should be 0."

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
93	snps_ace_dvm_complete_acaddr_zero	ERROR	IHI 0022E Ref: section C12.8 Table C12-11 on page C12-301	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	Must be zero for DVM Complete.
94	snps_ace_dvm_complete_ac	ERROR	IHI 0022E Ref: section C12.3.2 on page C12-292	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"For a DVM Complete message to be sent on the snoop address channel, there must be an outstanding DVM message that requires completion."
95	snps_ace_dvm_multi_consec_ac	ERROR	IHI 0022E Ref: section C12.3.3 on page C12-292	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	Multi-part of DVM messages are always sent as successive transactions and no other transaction can be interposed between them.
96	snps_ace_rresp_dvm_3_2_0_zero	ERROR	IHI 0022E Ref: section C12.3.4 on page C12-293	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0	"If CRRESP is 0b00000 for all responses, the interconnect component sets RRESP to 0b0000, if CRRESP is 0b00010 for any response, the interconnect component sets RRESP to 0b0010."
97	snps_ace_cdready_max_waits	WARNING	Recommendation	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 &&CONFIG_RECOMMEND==1 && CONFIG_WAITS ==1	CDREADY should be asserted within CD_MAX_WAITS cycles of CDVALID being asserted.
98	snps_ace_crready_max_waits	WARNING	Recommendation	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 &&CONFIG_RECOMMEND==1 && CONFIG_WAITS ==1	CRREADY should be asserted within CR_MAX_WAITS cycles of CRVALID being asserted.
99	snps_ace_acvalid_maxouts	WARNING	Maximum number of outstanding transactions.	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_MAXOUTS==1	Should not issue Snoop requests more than SN_MAX_BURSTS.

Table 4-11 ACE Specific Properties (Not applicable for ACE-Lite)

	Property Name	Error Kind	Spec Reference	Master/ Slave	Parameter Required	Property Description
100	snps_ace_acvalid_x	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on ACVALID is not permitted.
101	snps_ace_acsnoop_x	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on ACSNOOP is not permitted.
102	snps_ace_acprot_x	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on ACPROT is not permitted.
103	snps_ace_acaddr_x	ERROR	IHI 0022E Ref: section C3.6.2 on page C3-179	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on ACADDR is not permitted.
104	snps_ace_crready_x	ERROR	IHI 0022E Ref: section C3.7 on page C3-181	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CRREADY is not permitted.
105	snps_ace_cdready_x	ERROR	IHI 0022E Ref: section C3.8 on page C3-185	SLAVE	ENABLE_ASS ERT==1 && ACE_LITE==0 && CONFIG_X_CHE CK==1	A value of X/Z on CDREADY is not permitted.

4.4 Behavior of Properties

Properties are grouped on the basis of categories, which depends on the configuration parameter values. Categories can be like x_check properties, configure command properties and max waits properties. Parameters have some default value, refer to [Table 4-2](#) to know default values.

- To instantiate x_check properties:
Set `CONFIG_X_CHECK=1`
- To instantiate write strobe check properties:
Set `CONFIG_WSTRB=1`
- To instantiate max wait check properties:
Set `CONFIG_WAITS=1`

Similarly, to enable all assert or assume properties, the `ENABLE_ASSERT` and `ENABLE_ASSUME` parameters must be set to 1. Refer to “[The ACE AIP Properties](#)” for information on “Required parameters” for each property.

4.4.1 Properties as Assert Directives

Assert properties have the following features:

- Assert properties check the functionality of a protocol by monitoring its output as per its input, and issue an error message when the protocol is violated.
- When `AGENT_TYPE` is MASTER, checkers mentioned as ‘Master’ in the Master/Slave checkers column are declared as assume, and checkers mentioned as ‘Slave’ in Master/Slave column are declared as assert.
- When `AGENT_TYPE` is SLAVE, checkers mentioned as ‘Slave’ in Master/Slave column are declared as assume, and checkers mentioned as ‘Master’ in Master/Slave column are declared as assert.

4.4.2 Properties as Assume Directives

Assume properties have the following features:

- Assume properties act as a constraint for generating controlled stimulus as per a protocol because the VC Formal tool treats the inputs as free variables.
- When `AGENT_TYPE` is SLAVE, checkers mentioned as ‘Slave’ in Master/Slave checkers column are declared as assume, and checkers mentioned as ‘Master’ in Master/Slave column are declared as assert.
- When `AGENT_TYPE` is SLAVE, checkers mentioned as ‘Slave’ in Master/Slave column are declared as assume, and checkers mentioned as ‘Master’ in Master/Slave column are declared as assert.

4.4.3 Properties In Cover Directives

Cover properties have the following features:

- Cover properties tell the number of assertions executed or covered when stimulus is generated. This number reflects the AIP coverage. Also, the cover properties indicate the type of transactions or scenarios exercised during proof. It can be helpful in checking number of unexecuted properties.
- Cover properties are useful in checking if there are no over-constraints in environment, and if the design issues all possible transactions.

- When ENABLE_COVER = 1, cover properties are generated. Note that the cover properties are independent from the properties used as assert/assume.

4.5 The CONFIG_MAXOUTS Parameter Setting Change

Starting with the 2020.03-SP2-1 patch release, the CONFIG_MAXOUTS parameter setting is changed in ACE AIP. This enhancement allows you to adjust constraints to avoid missing bugs in design implementation.

[Table 4-12](#) describes difference in behavior of the CONFIG_MAXOUTS parameter.

Table 4-12 CONFIG_MAXOUTS Parameter Setting Behavior Change

Release	Behavior
2020.03-SP2 and below versions	CONFIG_MAXOUTS can be either 0 or 1 Default value is 1
2020.03-SP2-1 and above versions	CONFIG_MAXOUTS==0: same with previous version CONFIG_MAXOUTS==1: new behavior CONFIG_MAXOUTS==2: same with CONFIG_MAXOUTS==1 in previous version Default value is 1

[Table 4-13](#) describes behavior per value:

Table 4-13 Behavior Per Value

CONFIG_MAXOUTS==0	No change with previous version
	Master AIP does not constrain AWVALID, WVALID nor ARVALID. These signals may be asserted even the number of outstanding transactions exceed WR_MAX_BURSTS or RD_MAX_BURSTS.
	Master AIP does not check AWREADY, WREADY nor ARREADY if they are de-asserted even when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.
	Slave AIP does not check AWVALID, WVALID nor ARVALID if they are de-asserted even when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.
	Slave AIP does not constrain AWREADY, WREADY nor ARREADY. These signals may be asserted even the number of outstanding transactions exceed WR_MAX_BURSTS or RD_MAX_BURSTS.
CONFIG_MAXOUTS==1	New behavior
	Master AIP constrains AWVALID, WVALID and ARVALID with Low when the number of outstanding transactions reaches (WR_MAX_BURSTS+1) or (RD_MAX_BURSTS+1). Use asm_snps_axi_awvalid_maxexcd, asm_snps_axi_wvalid_maxexcd and asm_snps_axi_arvalid_maxexcd.

Table 4-13 Behavior Per Value

	<p>Master AIP checks AWREADY, WREADY and ARREADY if they are de-asserted when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>
	<p>Slave AIP checks AWVALID, WVALID and ARVALID if they are de-asserted when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>
	<p>Slave AIP constrains AWREADY, WREADY and ARREADY with Low when the number of outstanding transactions reaches (WR_MAX_BURSTS+1) or (RD_MAX_BURSTS+1).</p> <p>Use asm_snps_axi_awvalid_maxexcd, asm_snps_axi_wvalid_maxexcd and asm_snps_axi_arvalid_maxexcd.</p>
CONFIG_MAXOUTS==2	Same with previous behavior with CONFIG_MAXOUTS==1
	<p>Master AIP constrains AWVALID, WVALID and ARVALID with Low when the number of outstanding transactions reaches WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>
	<p>Master AIP checks AWREADY, WREADY and ARREADY if they are de-asserted when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>
	<p>Slave AIP checks AWVALID, WVALID and ARVALID if they are de-asserted when the number of outstanding transactions reaches to WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>
	<p>Slave AIP constrains AWREADY, WREADY and ARREADY with Low when the number of outstanding transactions reaches WR_MAX_BURSTS or RD_MAX_BURSTS.</p> <p>Use asm_snps_axi_awvalid_maxouts, asm_snps_axi_wvalid_maxouts and asm_snps_axi_arvalid_maxouts.</p>

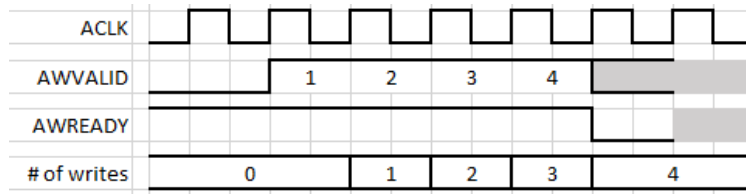
This change is applicable only for assume properties. This is not applicable for AMBA specification and depends on design implementation. Hence, assertions ast_snps_axi_awvalid_maxouts, ast_snps_axi_wvalid_maxouts, and ast_snps_axi_arvalid_maxouts do not check protocol violation and are categorized as WARNING.

Setting CONFIG_MAXOUTS to 0 is not recommended because it most likely causes overflow in AIP internal data structure.

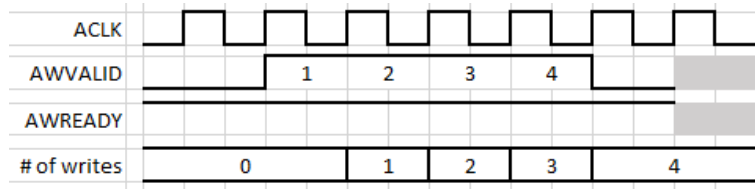
4.5.1 Background of this Change (when a potential bug can be missed in a slave DUT):

In previous releases, if you set `WR_MAX_BURSTS` with 4, MASTER AIP issues up to 4 outstanding write requests, and won't issue 5th write request until (`bvalid & bready`) is received.

Let us assume slave DUT has 4 depth FIFO and slave DUT stops driving `AWREADY` when FIFO is full. The expected slave DUT behavior is as follows:

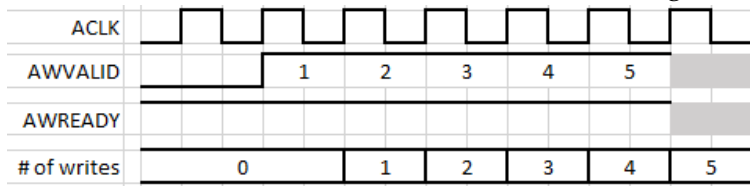


However, if MASTER AIP does not issue 5th write request, then it is not possible to verify if slave DUT stops `AWREADY` or not for the 5th request. As shown below, the number of outstanding transactions is 4 and slave DUT returns 5th `AWREADY`.

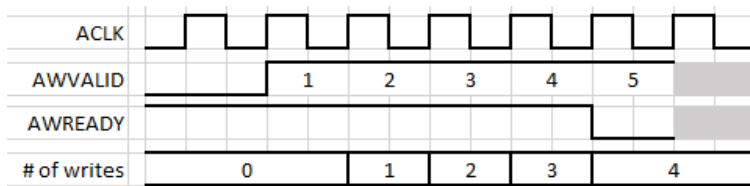


To verify if slave DUT stops to return `AWREADY`, you must set `WR_MAX_BURSTS` with 5 or bigger. However, user most likely sets `WR_MAX_BURSTS` with 4 in such cases. Therefore, it would be desirable to change AIP behavior that MASTER AIP issues write requests up to (`WR_MAX_BURSTS+1`). This is same with read requests.

As shown below, if slave DUT returns 5th `AWREADY`, the number of outstanding transactions reaches 5 and exceeds `WR_MAX_BURSTS`, and AIP can detect slave DUT bug.



If slave DUT stops to return 5th `AWREADY`, the number of outstanding transactions reaches 4 and MASTER AIP drives 5th `AWVALID` as follows:.




Default behavior is changed to issue (`WR_MAX_BURST+1`) outstanding write requests and (`RD_MAX_BURSTS+1`) read requests.

This behavior change is applied only to assume properties. There is no change for assert properties.

4.5.2 Backwards Compatibility

In some cases, it is required to keep previous behavior. For example, AIP back-to-back environment. Another example is if DUT is AXI bridge and upstream transactions are passed through to downstream,



DUT does not have any limitation regarding number of outstanding transactions. In such cases, CONFIG_MAXOUTS should be set with 2.

The ACE AIP Use Cases

This chapter discusses about the ACE-Lite AIP in different environments used for validation.

5.1 The ACE AIP Examples

This section describes the setup of the ACE-Lite AIP where it is connected with RTL through a bind file. The bind file example is shown in [Figure 5-2](#), [Figure 5-4](#), [Figure 5-6](#), and [Figure 5-8](#), where both master and slave DUT signals are connected to AIP master and slave signals.



Note

If a signal corresponding to the ACE AIP port does not exist in the DUT, set inactive value or the value expected by the DUT for the port. For example, if the DUT does not have the `awlock` signal, set with `1'b0`.

5.1.1 The ACE-Lite Master AIP With Slave DUT

The following steps describe the setup of the ACE AIP with a slave DUT:

1. In this setup, the ACE-Lite Master AIP is connected with the ACE-Lite Real Slave DUT.
2. For connecting the ports of the ACE-Lite DUT with the ACE-Lite AIP, create a bind file to include the instance of the ACE-Lite AIP into the top level module of the DUT.
3. Instantiate the master ACE-Lite AIP and connect with the slave DUT signal.
4. [Figure 5-1](#) shows a slave DUT connected with the master ACE-Lite AIP.

Figure 5-1 Slave DUT with ACE-Lite Master AIP

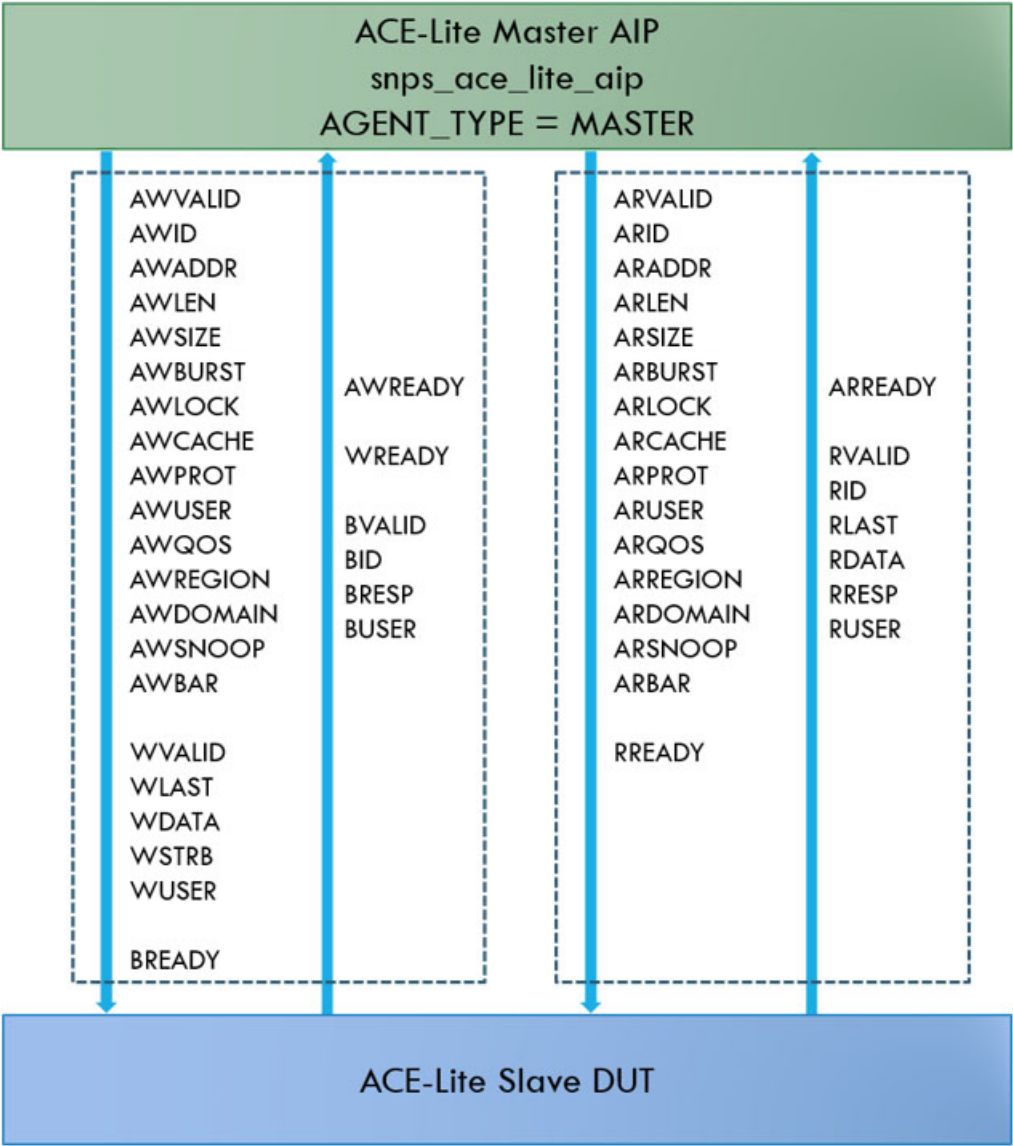
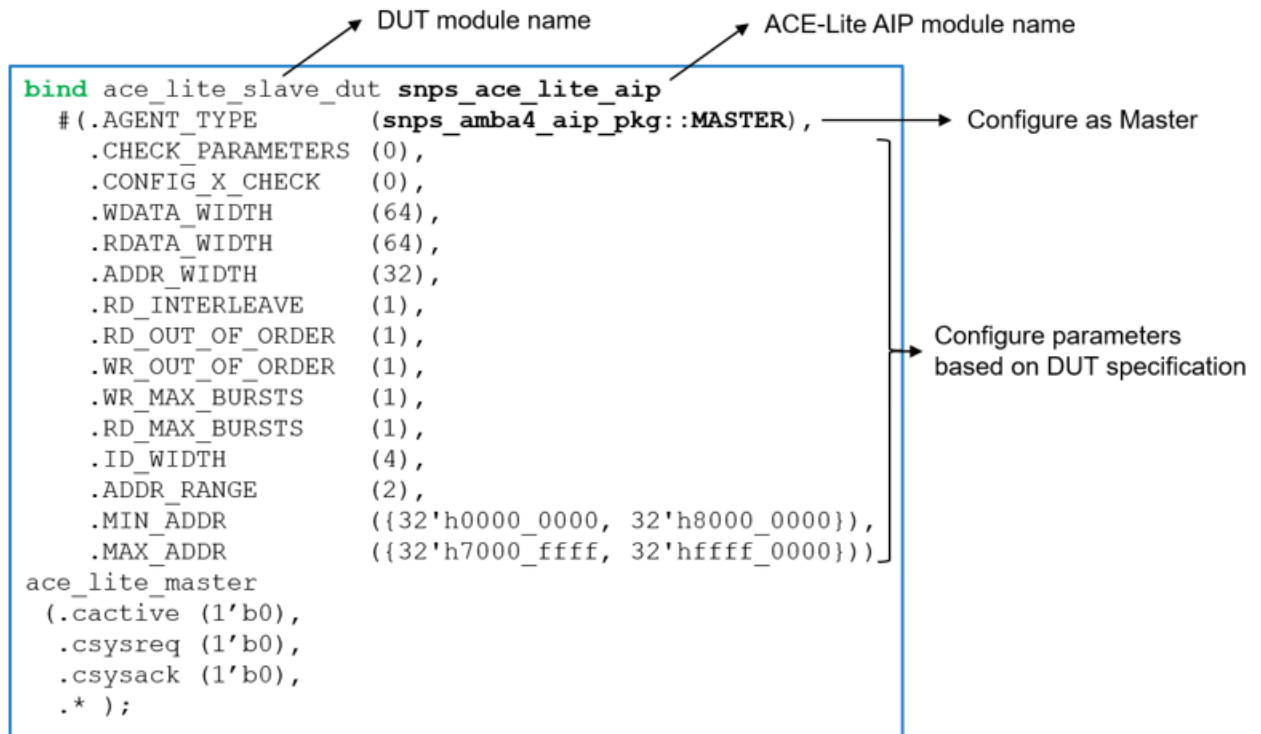


Figure 5-2 ACE-Lite Master AIP – Slave DUT bind example



5.1.2 The ACE-Lite Slave AIP With a Master DUT

The following steps describe the setup of the ACE-Lite Slave AIP with a master DUT:

- In this setup, the ACE-Lite Slave AIP is connected with the ACE-Lite Real Master DUT.
- For connecting the ports of the ACE-Lite DUT with the ACE-Lite AIP, create a bind file to include the instance of the ACE-Lite AIP into the top level module of the DUT.
- Instantiate the ACE-Lite Slave AIP and connect with the master DUT signal.
- [Figure 5-3](#) shows a Master DUT connected with the ACE-Lite Slave AIP.



Note

Use “snps_ace_lite_aip.sv” file for ACE-Lite AIP and configure `AGENT_TYPE` parameter with `SLAVE`.

Figure 5-3 Master DUT with ACE-Lite Slave AIP

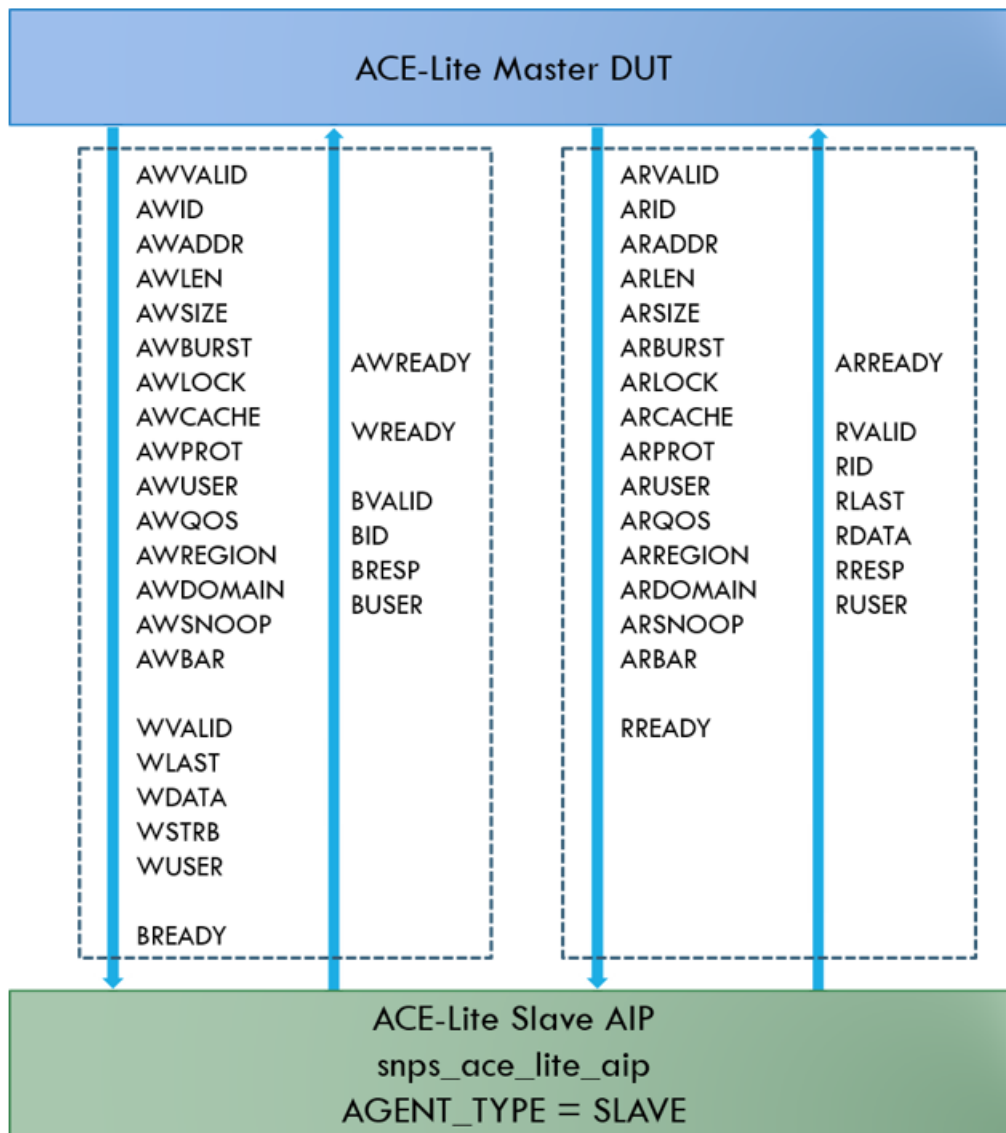


Figure 5-4 ACE-Lite Slave AIP – Master DUT bind example

```
bind ace_lite_master_dut snps_ace_lite_aip
# (.AGENT_TYPE      (snps_amba4_aip_pkg::SLAVE),
  .CHECK_PARAMETERS (0),
  .CONFIG_X_CHECK   (0),
  .WDATA_WIDTH      (64),
  .RDATA_WIDTH      (64),
  .ADDR_WIDTH       (32),
  .RD_INTERLEAVE     (1),
  .RD_OUT_OF_ORDER   (1),
  .WR_OUT_OF_ORDER   (1),
  .WR_MAX_BURSTS     (1),
  .RD_MAX_BURSTS     (1),
  .ID_WIDTH          (4),
  .ADDR_RANGE        (2),
  .MIN_ADDR          ({32'h0000_0000, 32'h8000_0000}),
  .MAX_ADDR          ({32'h7000_ffff, 32'hffff_0000}))
ace_lite_slave
(.cactive (1'b0),
 .csysreq (1'b0),
 .csysack (1'b0),
 .* );
```

5.1.3 The ACE Master AIP With Slave DUT

The following steps describe the setup of the ACE AIP with a slave DUT:

- In this setup, the ACE MasterAIP is connected with the ACE Slave DUT.
- For connecting the ports of the ACE DUT with the ACE AIP, create a bind file to include the instance of the ACE AIP into the top level module of the DUT.
- Instantiate the master ACE AIP (snps_ace_aip) and connect with the slave DUT signal.
- [Figure 5-5](#) shows a slave DUT connected with the master ACE AIP.



Note

Use “snps_ace_aip.sv” file for ACE-Lite AIP and configure AGENT_TYPE parameter with MASTER.

Figure 5-5 Slave DUT with ACE Master AIP

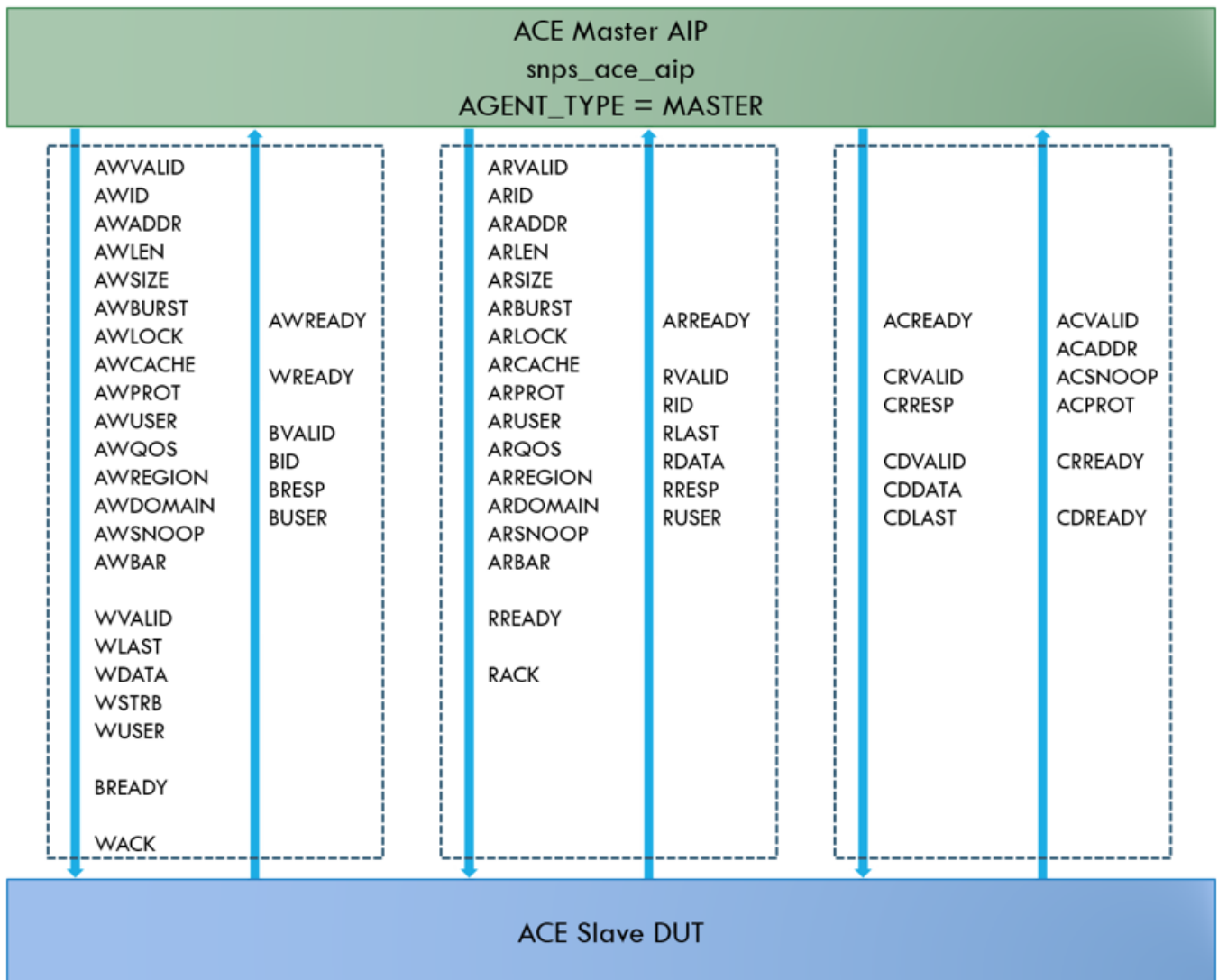
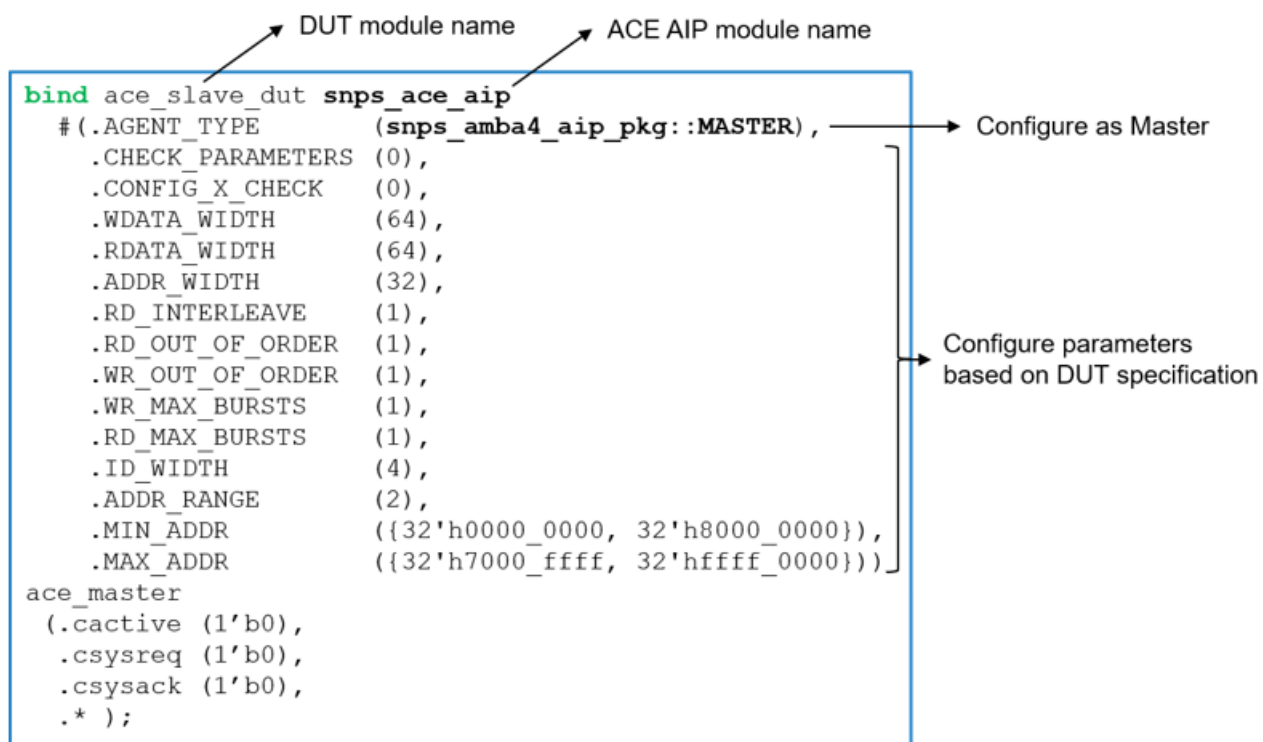


Figure 5-6 ACE Master AIP – Slave DUT bind example



5.1.4 The ACE Slave AIP With Master DUT

The following steps describe the setup of the ACE Slave AIP with a Master DUT:

- In this setup, the ACE Slave AIP is connected with the ACE Master DUT.
- For connecting the ports of the ACE DUT with the ACE AIP, create a bind file to include the instance of the ACE AIP into the top level module of the DUT.
- Instantiate the master ACE AIP (snps_ace_aip) and connect with the slave DUT signal.
- [Figure 5-7](#) shows a slave DUT connected with the ACE Slave AIP.



Note

Use “snps_ace_aip.sv” file for ACE AIP and configure AGENT_TYPE parameter with SLAVE.

Figure 5-7 Master DUT with ACE Slave AIP

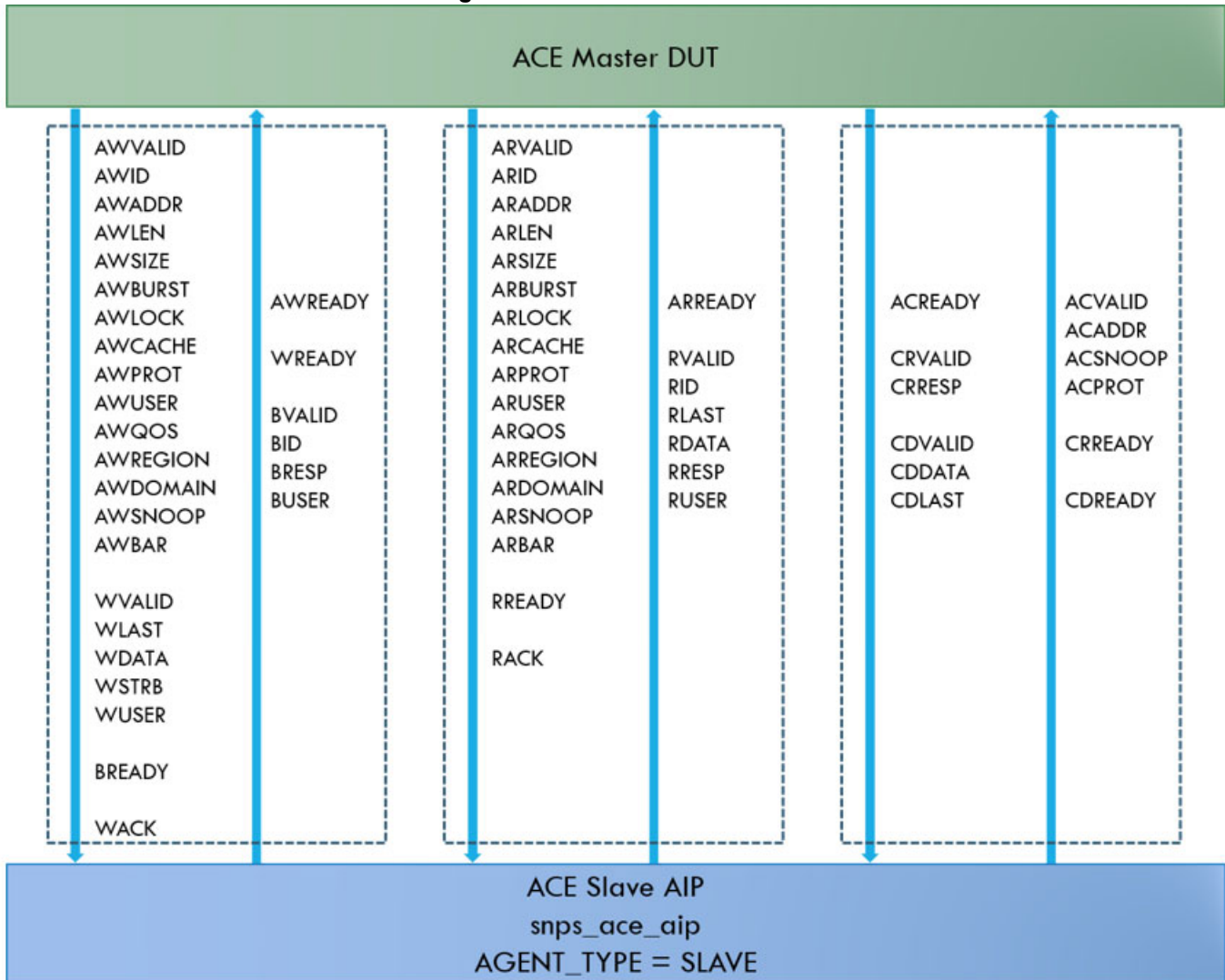


Figure 5-8 ACE Slave AIP – Master DUT bind example

```
bind ace_master_dut snps_ace_aip
#(.AGENT_TYPE      (snps_amba4_aip_pkg::SLAVE),
 .CHECK_PARAMETERS (0),
 .CONFIG_X_CHECK   (0),
 .WDATA_WIDTH      (64),
 .RDATA_WIDTH      (64),
 .ADDR_WIDTH       (32),
 .RD_INTERLEAVE    (1),
 .RD_OUT_OF_ORDER  (1),
 .WR_OUT_OF_ORDER  (1),
 .WR_MAX_BURSTS    (1),
 .RD_MAX_BURSTS    (1),
 .ID_WIDTH         (4),
 .ADDR_RANGE       (2),
 .MIN_ADDR         ({32'h0000_0000, 32'h8000_0000}),
 .MAX_ADDR         ({32'h7000_ffff, 32'hffff_0000}))
ace_slave
(.cactive (1'b0),
 .csysreq (1'b0),
 .csysack (1'b0),
 .* );
```

DUT module name

ACE AIP module name

Configure as Slave

Configure parameters based on DUT specification

