Verification Continuum™

VC Verification IP
PCIe Test Suite
PHY DUT Integration Guide

Version S-2021.06, June 2021

SYNOPSYS®

Copyright Notice and Proprietary Information

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 https://www.synopsys.com/company/legal/trademarks-brands.html.
All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

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

Preface

About This Document

This guide provides the information on how to integrate user PHY DUT with Synopsys VIP in the test suite testbench or user testbench and verify the integration using the recommended tests.

Web Resources

- Documentation through SolvNetPlus: https://solvnetplus.synopsys.com (Synopsys password required)
- Synopsys Common Licensing (SCL): http://www.synopsys.com/keys

Customer Support

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

- Go to https://solvnetplus.synopsys.com and open a case.

  Enter the information according to your environment and your issue.
- Send an e-mail message to support_center@synopsys.com
  - Include the Product name, Sub Product name, and Product version for which you want to register the problem.
- 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:

    https://www.synopsys.com/support/global-support-centers.html

Synopsys Statement on Inclusivity and Diversity

Synopsys is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

1 PHY DUT Integration

The chapter describes the steps to integrate user PHY DUT with Synopsys VIP in the test suite testbench or user testbench and verify the integration using the recommended tests.

This chapter discusses the following topics:

- Integrating PHY DUT
- Validating the Integration

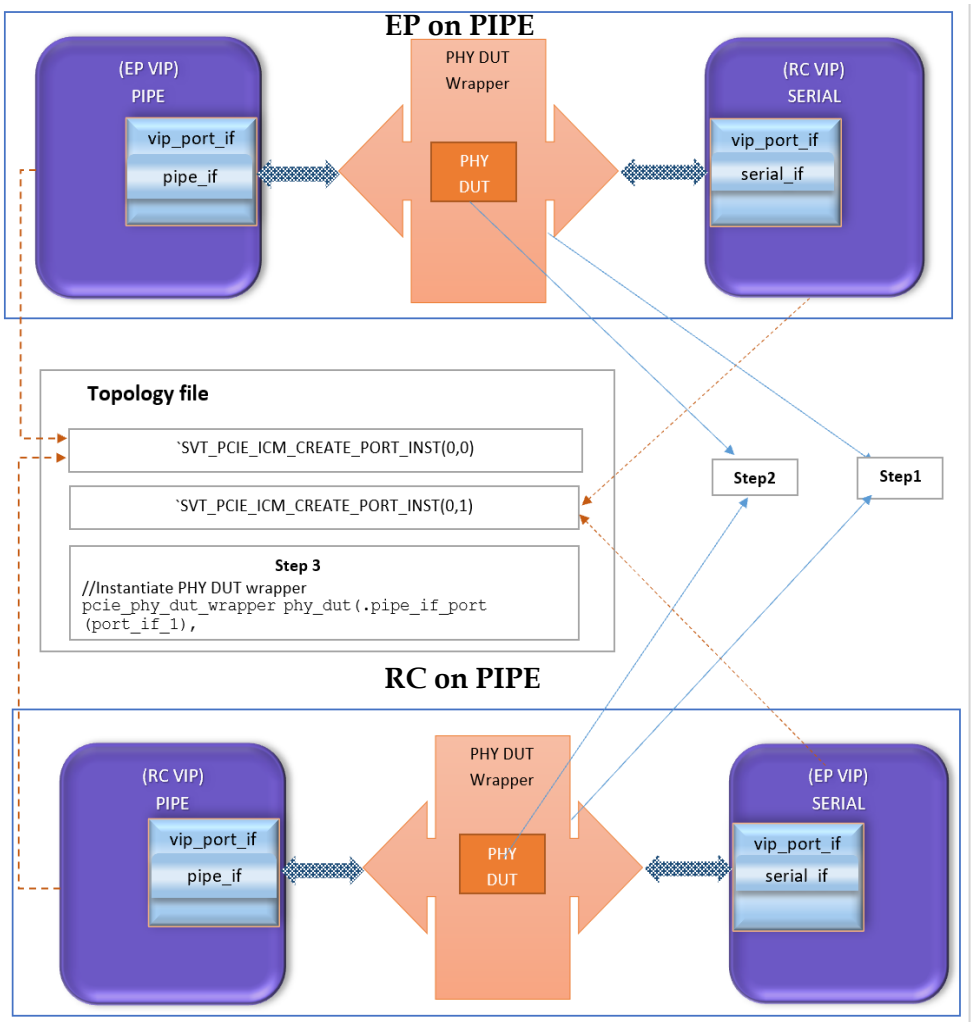
1.1 Integrating PHY DUT

Perform the following steps to integrate a PHY DUT:

- Step 1: Create a DUT Wrapper File
- Step 2: Instantiate the DUT
- Step 3: Create a Topology File
- Step 4: Base Test Changes
- Step 5: Create a Compile File
- Step 6: Compile and Run

The PCIe test suite supports two modes of PHY DUT operation as shown in Figure 1-1 (EP on PIPE and RC on PIPE).

Figure 1-1 PHY DUT Modes - EP and RC



Step 1: Create a DUT Wrapper File

Create a DUT wrapper module wherein the inputs are two ports of `svt_pcie_if` interface for Serial and PIPE connection.

```
module pcie_phy_dut_wrapper (svt_pcie_if pipe_if_port,
                             svt_pcie_if serdes_if_port
                            );
endmodule
```

Step 2: Instantiate the DUT

- Instantiate the PHY DUT in this module and connect the PIPE and Serial VIP through the ports.

  For more details on connections, see *pciesvc_phy_model.sv* in the *env* directory. You can also make a copy of this file and replace the demo PHY `pciesvc_port_model_x8_pipe_phy_8g` instance with your DUT instance and update the connections.

- Include the PHY DUT wrapper file and other DUT include files in *cust_pre_tb_top.svi.*

  For example,

```
`include "pcie_phy_dut_wrapper.sv"
```

Step 3: Create a Topology File

Topology file contains VIP instantiation and connection to DUT. Make a copy of the topology file *topology_dut_snps_pcie_phy_vip.svi* with a desired name and make the following changes in your topology file to integrate the PHY DUT. The topology file contains VIP instantiations and connections using helper macros which are defined in the file *env/hdl_interconnect_macros.sv*. The PHY used here is a behavioral PHY model wrapped in a top-level wrapper and instantiated in the topology file. You must replace the wrapper of PHY behavioral model with your PHY DUT as discussed in the following steps:

1. Update the number of physical lanes to a desired number. Currently, it is set to 8.

```
defparam SVT_PCIE_TEST_SUITE_NUM_PHYSICAL_LANES_P0= 4;
defparam SVT_PCIE_TEST_SUITE_NUM_PHYSICAL_LANES_P1= 4;
```

2. Replace the demo DUT wrapper instance with your DUT wrapper module instance created in Step 1.

```
defparam phy_dut.DISPLAY_NAME = "phy_dut.";
// original code
defparam phy_dut.DISPLAY_NAME = "phy_dut.";
pciesvc_phy_model phy_dut(.pipe_if_port (port_if_1),
.serdes_if_port (port_if_0));
// modified code
defparam phy_dut.DISPLAY_NAME = "phy_dut.";
pcie_phy_dut_wrapper phy_dut(.pipe_if_port (port_if_1),
.serdes_if_port (port_if_0));
```

3. Remove the following code from topology file as this check is applicable only in demo mode.

```
initial begin
  `ifdef SVT_PCIE_TEST_SUITE_MAX_LINK_WIDTH
    if (`SVT_PCIE_TEST_SUITE_MAX_LINK_WIDTH != 8)
     begin
       `uvm_fatal("topology_dut_snps_pcie_phy_vip",
       $sformatf("Expected value of
       macro SVT_PCIE_MAX_LINK_WIDTH is 8 observed value =
       %0d",`SVT_PCIE_TEST_SUITE_MAX_LINK_WIDTH));
     end
  `endif
    if($test$plusargs("svt_pcie_target_link_width"))
    begin
      `uvm_fatal("topology_dut_snps_pcie_phy_vip",
      $sformatf("Do not use plusarg
      svt_pcie_target_link_width to change run time width,
      instead refer the
      tests/*.scr files for correct plusarg , keyword
      link_width"));
    end
end
```

4. Include the topology file in the *top* module.

   The top module in *top.sv* file has the topology file named *svt_pcie_test_suite_topology_file.svi* included in it.

```
`include "svt_pcie_test_suite_topology_file.svi"
```

   The user topology file automatically gets included in the top module in *top.sv*. This happens when you pass the user topology filename to the `gmake` command using the `SVT_PCIE_TEST_SUITE_TOPOLOGY_FILE` switch.

   For example,

```
SVT_PCIE_TEST_SUITE_TOPOLOGY_FILE=topology_dut_user_phy.svi
```

   A symbolic link with the name *svt_pcie_test_suite_topology_file.svi* pointing to user topology file gets created. This makes the user topology file automatically part of *top* module without user explicitly including it.

Note Next step can be ignored if you are directly using the test suite top module that is present in *top.sv* as `top_module`.

5. If you want to use your own *top* module, then use one of the following recommended flows:
   a. SNPS *top* module as child instance in user *top* module.
      i. Instantiate the test suite `test_top` module in your existing *top* module (for example, *tb_top.sv*).
      ii. Rename `SNPS test_top` module.

      You can override `SNPS test_top` module with any name by overriding macro `SVT_PCIE_TEST_SUITE_SNPS_TEST_TOP`. The default value of this macro is `test_top`.
      iii. Invoke `SNPS test_top` as child module in user `test_top` module.

      Here, you can invoke `run_phase` from `SNPS` *top* module using the `SVT_PCIE_TEST_SUITE_INVOKE_RUN_PHASE` parameter.

      The default value of `SVT_PCIE_TEST_SUITE_INVOKE_RUN_PHASE` parameter is 1, which implies that the `run_phase` is invoked from `SNPS` *top* module.
   b. Test suite *top.sv* file contents in user *top* file.

      Copy the contents of SNPS *top* module (*top.sv*) into user *top* file.

      The following snippet from the *top.sv* file lists the necessary major hooks. However, the entire `test_top` module is required.

```
`include "cust_pre_tb_top.svi"
`include "snps_pcie_test_suite.pkg"
// This file further include "cust_pcie_test_suite_pkg_files.svi"
module test_top();
import snps_pcie_test_suite_pkg::* ;
`include "svt_pcie_test_suite_topology_file.svi"
endmodule
```

i. *cust_pre_tb_top.svi*: This file is provided as a hook so that this can be populated to add DUT-specific defines or definitions of modules to be used (DUT itself/other VIPs and so on). Any file following the naming convention "*cust_*" will remain unchanged in the future releases. These files can be modified by the users.

   This file does not require any changes if you have done all the required inclusions in your top testbench. Existing test suite class packages will then be imported via the *snps_pcie_test_suite.pkg* hook. If the elements of user-defined packages are used in the top, then you must add import calls in the topology file.

Note Save a local copy of this file and make sure that your file is selected in the compile flow so that your changes are not overridden when the *tb* directory is upgraded with the new version of the test suite.

ii. *cust_pcie_test_suite_pkg_files.svi*: This file is provided as a hook so that you can populate this file to

   - Import the existing packages in user setup to this file (optional).

   - Add third-party DUT-specific classes as part of package `snps_pcie_test_suite`.

   - Include extended user env and bases test files.

   This file does not require any changes if you have done all the required inclusions in your top testbench.

iii. *svt_pcie_test_suite_topology_file*: This file includes VIP instantiation and connection to DUT. If you have already done the connection in your connection file, then you can create a symbolic link of your DUT connection file with the name *svt_pcie_test_suite_topology_file.svi*, otherwise create a topology (connection) file as described in Step 3.

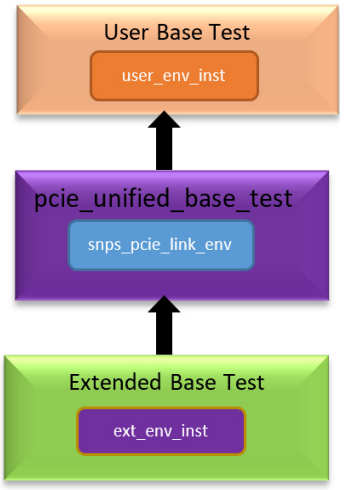
Step 4: Base Test Changes

The `pcie_phy_dut_base` test (extended from `pcie_unified_base_test`) which is part of the test suite has a default configuration settings of VIP. All VIP configuration settings customization as per the DUT must be done in extended the base test.

- Create a base test by extending from `pcie_unified_base_test`.

  If you have already configured the settings in your base test and also instantiating the env, then to get user base test along with env into the test suite testbench environment, use the recommended use model shown in the following figure.

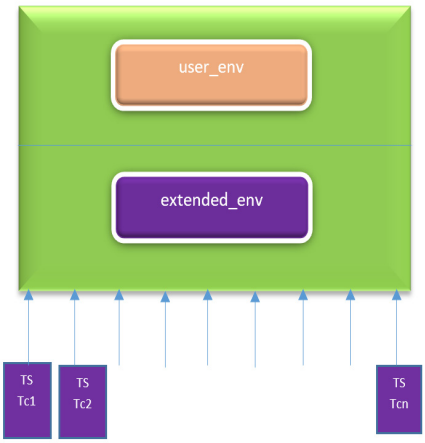
Figure 1-2 Base Test Hierarchy



- Extend the `pcie_unified_base_test` from `user_base_test`.

  This can be done by setting the macro `SVT_PCIE_TEST_SUITE_USER_BASE_TEST` to user base test name. By default, the `unified_base_test` is extended from `uvm_test`.
- As the extended base test is extended from the unified base test, it contains both user env instance and test suite env instance
- All the test suite tests are extended from this extended base test as shown in Figure 1-3.

Figure 1-3 Extended Base Test



You can make/add implementation-specific customizations in this base test class by referring to file *pcie_e/rcp_dut_cust_base_test.sv*. It is recommended that, in this file, you can add/modify only those

configurations that are generic, applicable to all test and are not dynamic. For all test-specific configurations, follow the *src/txt* file approach.

Note All the mandatory code is present only in `pcie_unified_base_test`, therefore you must extend your base test from `pcie_unified_base_test` depending on the requirement even for meeting multi-DUT requirement in a single test. The `pcie_phy_dut_cust_base_test` class must be used as a reference. It is recommended not to extend this class, use it only as a reference.

All test suite tests are extended from `pcie_unified_base_test` using the macro `` `SVT_PCIE_TEST_SUITE_RC_DUT_TEST `` or `` `SVT_PCIE_TEST_SUITE_EP_DUT_TEST ``. To replace the test suite base test with your extended base test, you must add the following macros in the compile file discussed in Step 5.

```
+define+SVT_PCIE_TEST_SUITE_RC_DUT_TEST=<extended_base_test_name>
+define+SVT_PCIE_TEST_SUITE_EP_DUT_TEST=<extended_base_test_name>
```

For example,

```
+define+SVT_PCIE_TEST_SUITE_RC_DUT_TEST=pcie_ep_dut_controller_base_test
```

Step 5: Create a Compile File

Following are the modes of operations supported by two different compile files.

- EP on PIPE (PHY DUT is Endpoint): *compile_dut_snps_pcie_phy_vip_ep_on_pipe.f*
- RC on PIPE (PHY DUT is Root Complex): *compile_dut_snps_pcie_phy_vip_rc_on_pipe.f*

Based on your configuration, copy one of these compile files as `dut_comple` file and add all the compilation options/flags/defines required for the DUT.

Also, remove the following define from the compile file as this is required only when demo PHY is used.

```
+define+SVT_PCIE_TEST_SUITE_PHY_IS_VIP
```

Step 6: Compile and Run

Compile and run the code using the *Makefile* or run script.

```
gmake USE_SIMULATOR=vcsvlog tl_directed_traffic-phy
SVT_PCIE_ENABLE_TRANSACTION_LOGGING=1 SVT_PCIE_PIPE_WIDTH=1
SVT_PCIE_TARGET_LINK_WIDTH=1
SVT_PCIE_TEST_SUITE_DUT_COMPILE_FILE=<compile_dut_user_phy_ep_on_pipe.f>
SVT_PCIE_TEST_SUITE_TOPOLOGY_FILE=<topology_dut_user_phy.svi>
SVT_PCIE_MAX_LINK_WIDTH=4 SVT_PCIE_LINK_SPEED=GEN1
```

- In the above example, the compile file created to compile the PHY DUT (non-demo) is set through `SVT_PCIE_TEST_SUITE_DUT_COMPILE_FILE` option.

  `SVT_PCIE_TEST_SUITE_DUT_COMPILE_FILE=compile_dut_user_phy_ep_on_pipe.f`

  - Topology file that was created for PHY DUT is set using the option `SVT_PCIE_TEST_SUITE_TOPOLOGY_FILE`.

1.2 Validating the Integration

Table 1-1 lists the tests required for validating the integration.

Table 1-1 Test Cases for Validating the Integration

Layer	Speed	Test Case	Description
PL	Gen-1	`demo_pl_gen1_linkup_test-phy`	Run this test to check and remove compilation errors. When the test is run successfully, then it shows the status as `Passed` which implies that PHY DUT integration is complete and basic flow at Gen1 speed is established. You can search for the following string in the transcript log file to confirm if the link is up at 2.5Gb/s LTSSM: `Link training completed. The link is READY! Speed is 2_5Gb/s.`
TL	Gen-1	`demo_tl_gen1_linkup_followed_by_tlps-phy`	This test case validates few TLPs (MRD) flow from VIP to DUT and expects the completion for those in Gen1 speed.
TL	Gen-2	`demo_tl_gen2_speed_change_followed_by_tlps-phy`	This test case validates few TLPs (MRD) flow from VIP to DUT and expects the completion for those in Gen2 speed. Run this test to check if the link up happens correctly at Gen2 speed. When the test is run successfully, then it shows the status as `Passed` which implies that PHY DUT is running fine and basic flow at Gen2 speed is established. You can search for the following string in the transcript log file to confirm if the link is up at 5Gb/s LTSSM: `Link training completed. The link is READY! Speed is 5Gb/s.`
PL	Gen-2	demo_pl_gen1_to_gen2_speed_change_test-phy	Run this test to verify if linkup happens correctly at Gen2 speed.
TL	Gen-3	`demo_tl_gen3_speed_change_followed_by_tlps-phy`	This test case validates few TLPs (MRD) flow from VIP to DUT and expects the completion for those.in Gen3 speed. Run this test to check if the link up happens correctly at Gen3 speed. When the test is run successfully, then it shows the status as `Passed` which implies that PHY DUT is running fine and basic flow at Gen3 speed is established. You can search for the following string in the transcript log file to confirm if the link is up at 8Gb/s LTSSM: `Link training completed. The link is READY! Speed is 8Gb/s.`
PL	Gen-3	demo_pl_gen1_to_gen3_speed_change_test-phy	Run this test to verify if linkup happens correctly at Gen3 speed.
PL	Gen-3	demo_gen3_pl_all_supported_speed_change_test-phy	Run this test to verify speed negotiation for all supported speeds.
PL	Gen-4	demo_gen4_pl_all_supported_speed_change_test-phy	Run this test to verify speed negotiation for all supported speeds.
PL	Gen-4	demo_pl_gen1_to_gen4_speed_change_test-phy	Run this test to verify if linkup happens correctly at Gen4 speed.

Table 1-1 Test Cases for Validating the Integration

Layer	Speed	Test Case	Description
PL	Gen-5	demo_pl_gen1_to_gen5_speed_change_test-phy	Run this test to verify if linkup happens correctly at Gen5 speed.