Verification Continuum™

VC Verification IP
PCIe
UVM Getting Started 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 Getting Started Guide presents information about integrating the VC VIP for PCIe (referred to as VIP) into testbenches that are compliant with the SystemVerilog Universal Verification Methodology (UVM). You are assumed to be familiar with the PCIe protocol and UVM.

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.
  - For simulation issues, provide a UVM_FULL verbosity log file of the VIP instance, a VPD or FSDB dump file of the VIP interface, and the transaction and symbol log.
- 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.

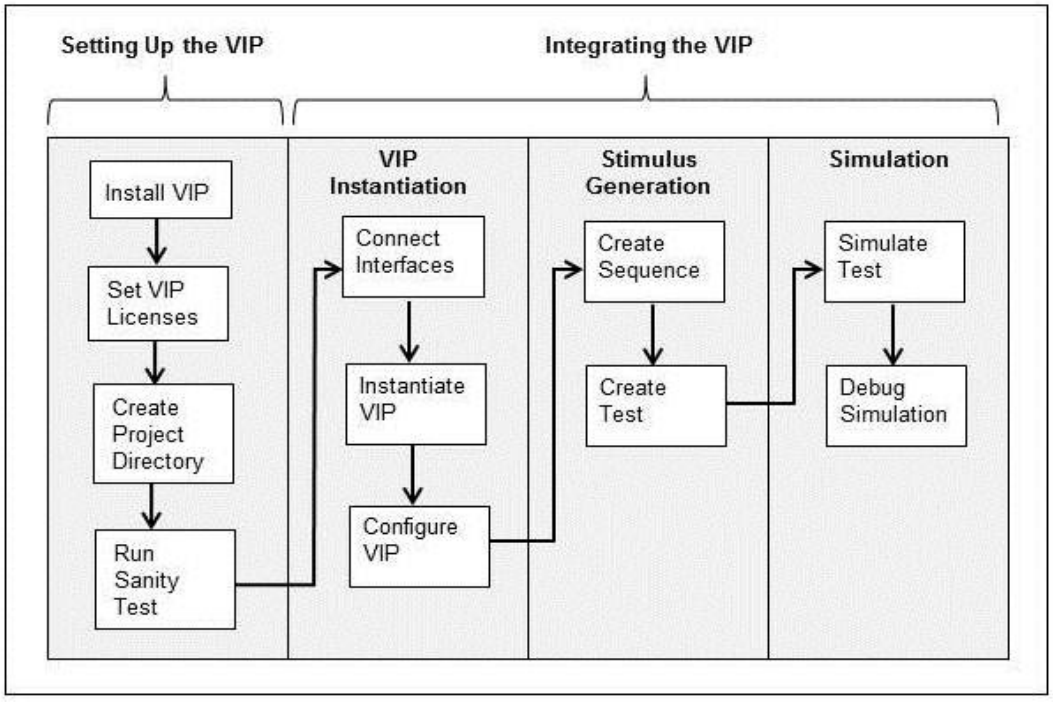
1 Overview of the Getting Started Guide

This Getting Started Guide presents information about integrating the VC VIP for PCIe (referred to as VIP) into testbenches that are compliant with the SystemVerilog Universal Verification Methodology (UVM). Figure 1-1 is the VIP integration and test work flow presented in this document. The steps for setting up the VIP are documented in the *VC Verification IP Installation and Setup Guide*. This guide is available on the SolvNetPlus Download page and in the VIP installation at the following location:

$DESIGNWARE_HOME/vip/svt/common/latest/doc/uvm_install.pdf

The VIP setup should be completed before executing the steps in this document.

Figure 1-1 VIP Integration and Test Work Flow



You are assumed to be familiar with the PCIe protocol and UVM. For more information on the VIP, refer to the *VC Verification IP PCIe UVM User Guide* on SolvNetPlus (click here) or in the VIP installation at the following location:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_user_guide.pdf

2 Integrating the VIP into a User Testbench

The VC VIP for PCIe provides a suite of advanced SystemVerilog verification components and data objects that are compliant to UVM. Integrating these components and objects into any UVM compliant testbench is straightforward. For a complete list of VIP components and data objects, refer to the main page of the *VC VIP PCIe Class Reference* (only in HTML format) at the following location:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_class_reference/html/index.html

2.1 VIP Testbench Integration Flow

A PCIe environment (pcie_device_unified_vip_env in Figure 2-1) is a user defined UVM environment that encapsulates all of the VIP components such as a Root Complex and an Endpoint Device in Figure 2-1. You can instantiate and construct the PCIe environment in the top-level environment of your UVM testbench.

Figure 2-1 Top-level Architecture of a PCIe VIP Testbench

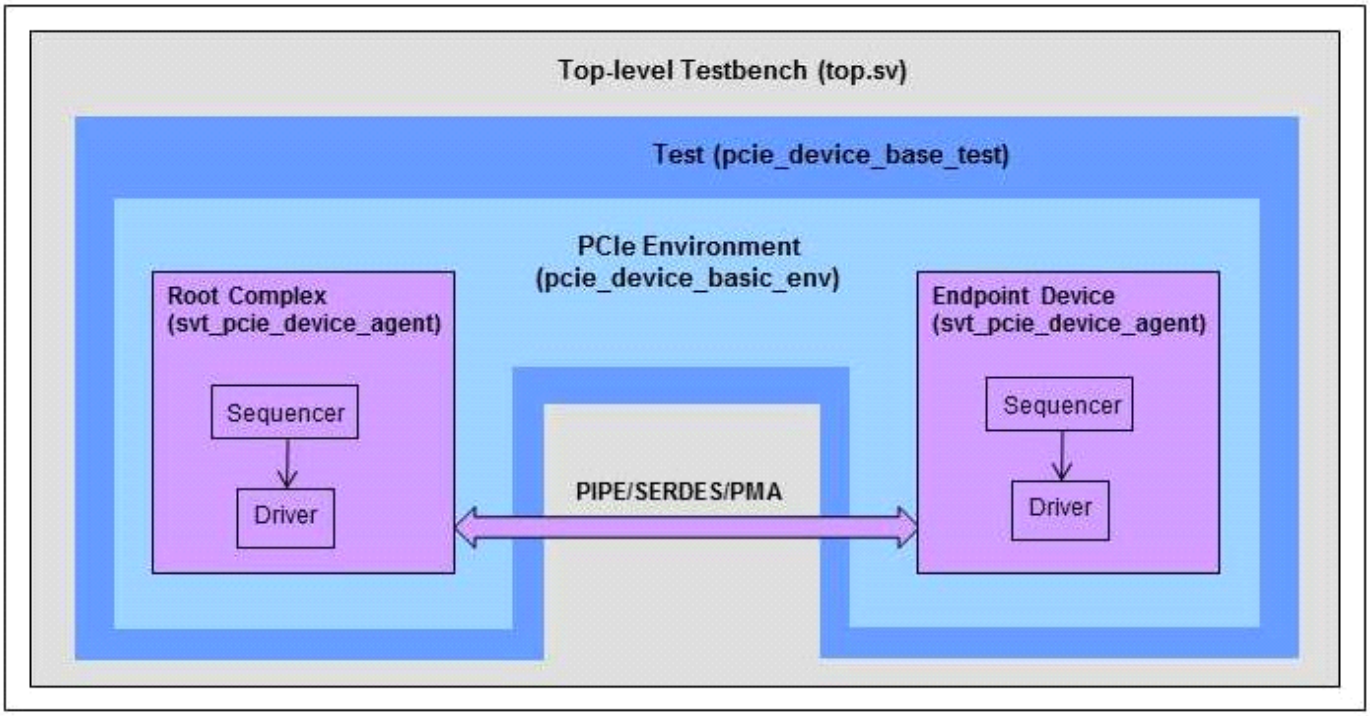


Figure 2-1 is a top-level architecture of a simple VC VIP for PCIe testbench. In a typical user environment, either the Root Complex or Endpoint component is replaced with the DUT. The steps for integrating the VIP into a UVM testbench are described in the following sections:

- Connecting the VIP to the DUT
- Instantiating and Configuring the VIP
- Creating a Test Sequence
- Creating a Test

The code snippets presented in this chapter are generic and can be applied to any UVM compliant testbench. For more information on the code usage, refer to the following example:

```
$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/examples/sverilog/
tb_pcie_svt_uvm_unified_vip_sys
```

You can use the PCIe environment from the “tb_pcie_svt_uvm_unified_vip_sys” example as a template for the environment that includes the integrated DUT.

2.1.1 Connecting the VIP to the DUT

The following are the steps to establish a connection between the VIP to the DUT in your top-level testbench:

- Set the timescale directive.

```
`timescale 1 ns/1 fs
```

- Set the required defines.

```
`define PCIESVC_MEM_PATH                                  test.top.mem0
`define EXPERTIO_PCIESVC_GLOBAL_SHADOW_PATH  test_top.global_shadow0
`define SVC_RANDOM_SEED_SCOPE                          test_top.global_random_seed
```

- Include the standard UVM and VIP files and packages.

```
`include "svt_pcie.uvm.pkg"     //VIP package
import uvm_pkg::*;
`include "uvm_macros.svh"
import svt_uvm_pkg::*;
import svt_pcie_uvm_pkg::*;
```

- Include the utility package and example environment.

```
`include "svc_util_parms.v"
`include "pcie_device_unified_vip_env.sv"
```

- Instantiate and connect the module based interface as shown below for PIPE.

```
`include "hdl_interconnect_macros.sv"
`include "top.pcie_pipe_topology.sv"
```

- Instantiate the global memory and the global memory shadow that is used for self checking.

```
svc_mem #(.DISPLAY_NAME("mem0."))mem0();

pciesvc_global_shadow #(.DISPLAY_NAME("global_shadow0."))
    global_shadow0();
```

- Reset the VIP.

```
initial begin
    $timeformat(-9,5, "ns", 12);
    test_top.reset = 1;
    #200;
    test_top.reset = 0;
end
```

2.1.2 Instantiating and Configuring the VIP

The following are steps to instantiate and configure the PCIe system environment in your testbench environment.

- Instantiate the PCIe environment (`pcie_device_unified_vip_env`) in the build phase of your testbench environment. The "`pcie_device_unified_vip_env`" class is provided by the "`tb_pcie_svt_uvm_unified_vip_sys`" example.

```
pcie_device_unified_vip_env env;

env = pcie_device_unified_vip_env::type_id::create("env", this);
```

- Specify the default system settings by using the default configuration and modifying the configuration as needed.

For example:

```
class pcie_shared_cfg extends uvm_object;
        …
  function void setup_pcie_device_system_defaults();
    begin
      root_cfg.pcie_cfg_enable_tl_xml_gen = 1'b1;
      root_cfg.pcie_cfg_enable_dl_xml_gen = 1'b1;
      root_cfg.pcie_cfg_enable_pl_xml_gen = 1'b1;
      root_cfg.device_is_root = 1;
      root_cfg.pcie_spec_ver =
                  svt_pcie_device_configuration::PCIE_SPEC_VER_2_1
                  root_cfg.pcie_cfg.pl_cfg.set_link_width_values(4);
      root_cfg.pcie_cfg.pl_cfg.
                  set_link_speed_values(`SVT_PCIE_SPEED_2_5G);
    end
  endfunction
        …
endclass
```

For more information on the configuration class, refer to the *svt_pcie_configuration* and *svt_pcie_device_configuration Class References* at the following locations:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_class_reference/html/configuration/class_svt_pcie_configuration.html

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_class_reference/html/configuration/class_svt_pcie_device_configuration.html

- Construct the customized PCIe configuration and pass the configuration to the PCIe environment (instance of `pcie_device_unified_vip_env`) in the build phase of your testbench environment.

```
cust_cfg = pcie_shared_cfg::type_id::create("cust_cfg");

uvm_config_db#(pcie_shared_cfg)::set(this, "*", "cfg", cust_cfg);
```

2.1.3 Creating a Test Sequence

The VIP provides a base sequence class for the driver application to initiate bus transactions (`svt_pcie_driver_app_transaction_base_sequence`). You can extend this base sequence or use one of the pre-built sequences provided by the VIP to initiate transactions.

Note The VIP also provides service sequences for each of the layers to control the model.

For more information on the PCIe driver base sequences, and the VIP sequence collection, refer to the Sequence Page of the *VC VIP PCIe Class Reference* at the following location:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_class_reference/html/sequencepages.html

In addition, a list of random and directed sequences is available in the VIP examples. For more information on the example sequences, refer to the example directories at the following location:

`$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/examples/sverilog`

2.1.4 Creating a Test

You can create a VIP test by extending the uvm_test class. In the build phase of the extended class, you construct the testbench environment and set the PCIe sequences.

```
class pcie_device_base_test extends uvm_test;

 // build_phase
  env = pcie_device_unified_vip_env::type_id::create("env", this);

  uvm_config_db#(uvm_object_wrapper)::set(this,
                 "env.sys_virt_seqr.main_phase", "default_sequence",
                 pcie_device_random_traffic_sequence::type_id::get());
```

2.2 Compiling and Simulating a Test with the VIP

The steps for compiling and simulating a test with the VIP are described in the following sections:

- Directory Paths for VIP Compilation
- VIP Compile-time Options
- VIP Runtime Option

2.2.1 Directory Paths for VIP Compilation

You need to specify the following directory paths in the compilation commands for the compiler to load the VIP files.

```
+incdir+project_directory_path/include/sverilog
+incdir+project_directory_path/src/sverilog/simulator
+incdir+project_directory_path/include/verilog
+incdir+project_directory_path/src/verilog/simulator
```

Where, *project_directory_path* is your project directory and *simulator* is vcs, ncv or mti.

For example:

```
+incdir+/home/project1/testbench/vip/include/verilog
+incdir+/home/project1/testbench/vip/include/sverilog
+incdir+/home/project1/testbench/vip/src/verilog/vcs
+incdir+/home/project1/testbench/vip/src/sverilog/vcs
```

2.2.2 VIP Compile-time Options

The following are the required compile-time options for compiling a testbench with the VC VIP for PCIe:

```
+define+SVT_UVM_TECHNOLOGY
+define+UVM_PACKER_MAX_BYTES=8192
+define+UVM_DISABLE_AUTO_ITEM_RECORDING
+define+SYNOPSYS_SV
```

Macro	Description
SVT_UVM_TECHNOLOGY	Specifies SystemVerilog based VIPs that are compliant with UVM
UVM_PACKER_MAX_BYTES	Sets to 8192 or greater
UVM_DISABLE_AUTO_ITEM_RECORDING	Disables the UVM automatic transaction begin and end event triggering and recording. This is not required with UVM 1.2 and newer versions due to a Mantix fix.
SYNOPSYS_SV	Specifies SystemVerilog based VIPs that are compliant with UVM

2.2.3 VIP Runtime Option

No VIP specific runtime option is required to run simulations with the VIP. Only relevant UVM runtime options are required.

For example:

```
+UVM_TESTNAME=base_pipe_test
```

A Summary of Commands, Documents, and Examples

A.1 Commands in This Document

Display VIP models and examples under the VIP installation directory specified by $DESIGNWARE_HOME:

```
% $DESIGNWARE_HOME/bin/dw_vip_setup -info home
```

Add VIP models to the project directory:

```
% $DESIGNWARE_HOME/bin/dw_vip_setup  -path project_directory -add VIP_model
-svlog
```

Add VIP examples to the directory where the command is executed:

```
% $DESIGNWARE_HOME/bin/dw_vip_setup -e VIP_example -svlog
```

A.2 Primary Documentation for VC VIP PCIe

VC Verification IP UVM Installation and Setup Guide:

$DESIGNWARE_HOME/vip/svt/common/latest/doc/uvm_install.pdf

VC VIP PCIe UVM User Guide:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_user_guide.pdf

VC VIP PCIe Getting Started Guide:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_getting_started.pdf

VC VIP PCIe Class Reference:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/pcie_svt_uvm_class_reference/html/index.html

VC VIP PCIe Verification Plans:

$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/doc/VerificationPlans

A.3 Example Home Directory

Directory that contains a list of VIP example directories:

```
$DESIGNWARE_HOME/vip/svt/pcie_svt/latest/examples/sverilog
```

View simulation options for each example:

```
gmake help
```