Verification Continuum™

VC Verification IP MIPI SPMI UVM Getting Started Guide

Version Q-2020.06, June 2020

SYNOPSYS®

Contents

Preface

About This Document

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

To register a problem, perform any of the following tasks:

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

   Enter the information according to your environment and your issue.
2. 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.
3. 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

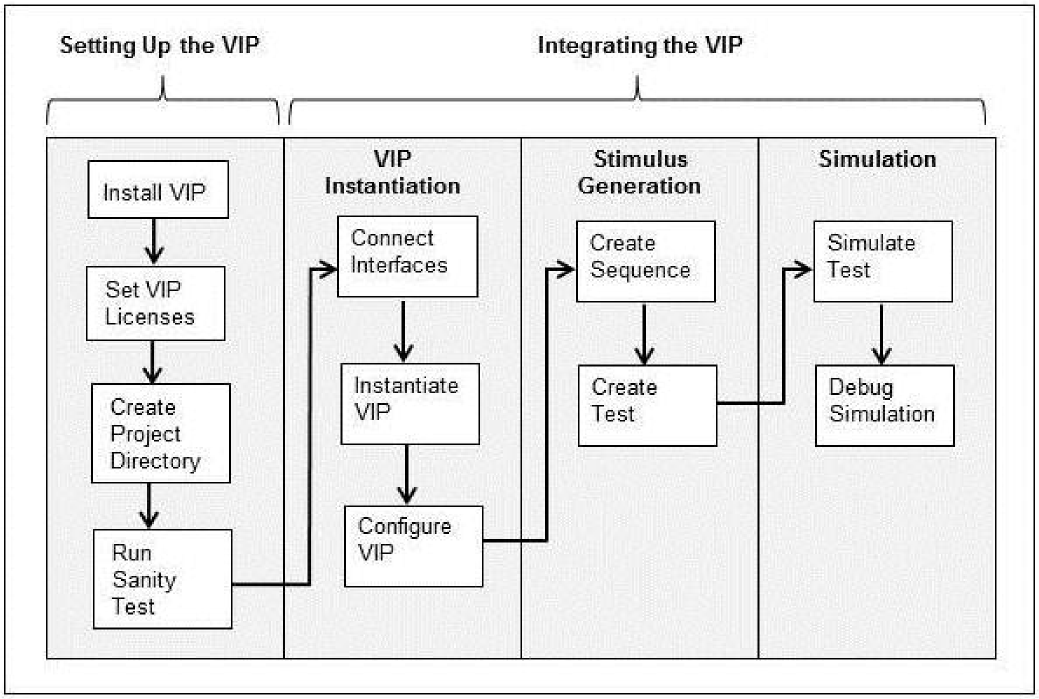
1 Overview of the Getting Started Guide

This Getting Started Guide presents information about integrating the VC VIP for MIPI SPMI (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 UVM Installation and Setup Guide*. This guide is available on the SolvNet Download Centre (click here -> VC VIP Library -> Q-2020.03-> Installation Guide) 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 MIPI SPMI protocol and UVM. For more information on the VIP, see the *VC Verification IP MIPI SPMI UVM User Guide* on SolvNet (click here) or in the VIP installation at the following location:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/mipi_spmi_svt_uvm_user_guide.pdf

2 Integrating the VIP into a User Testbench

The VC VIP for MIPI SPMI 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, see the main page of the *VC VIP MIPI SPMI Class Reference* (only in HTML format) at the following location:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/mipi_spmi_svt_uvm_class_reference/html/index.html

2.1 VIP Testbench Integration Flow

The MIPI SPMI agent (svt_mipi_spmi_agent) is the top-level component provided by the VIP for modeling SPMI Master and Slave devices. This generic agent encapsulates the following components:

- Sequencer
- Monitor
- Driver

You can instantiate and construct the MIPI SPMI agent in the top-level environment of your UVM testbench.

Figure 2-1 Top-level Architecture of a MIPI SPMI VIP Testbench

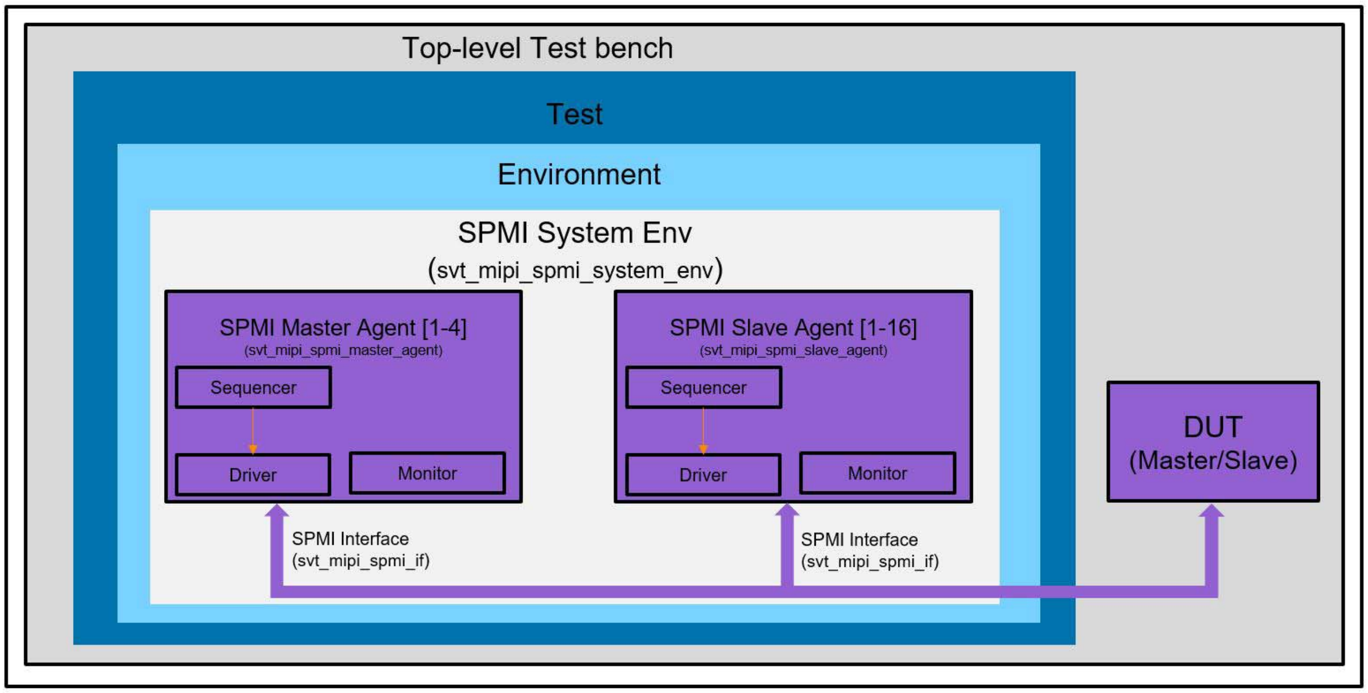


Figure 2-1 is a top-level architecture of a simple VC VIP for MIPI SPMI testbench. The steps for integrating the VIP into a UVM testbench are described in the following sections:

- "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, see the following example:

```
$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/examples/sverilog/
tb_mipi_spmi_svt_uvm_basic_sys
```

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:

- Include the standard UVM and VIP files and packages.

```
`include "svt_mipi_spmi.uvm.pkg"

import uvm_pkg::*;
import svt_uvm_pkg::*;
import svt_mipi_spmi_uvm_pkg::*;
```

2.1.1.1 Topology: DUT as Master or Slave or Both

If Rx VIP, then follow the following steps:

- In testbench top, import UVM package, macros and SPMI UVM package. Instantiate SPMI interface.

```
svt_mipi_spmi_system_if system_if();
```

- Declare global slck and sdata signals in top file.

```
wire sclk, sdata;

//Connect the master and slave interfaces
  genvar i;
  genvar j;

  generate
    for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; j++) begin : wrapper_mm
      mipi_spmi_svt_hdl_interconnect
master_interconnect_wrapper(system_if.master_if[j].sdata,sdata,
system_if.master_if[j].sclk,sclk);
    end
  endgenerate

  generate
    for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; j++) begin : wrapper_ss
      mipi_spmi_svt_hdl_interconnect
slave_interconnect_wrapper(system_if.slave_if[j].sdata, sdata,
system_if.slave_if[j].sclk, sclk);
    end
  endgenerate

//Connect the debug interfaces of Master VIP components
  generate
    for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; j++) begin : wrapper_11
      for (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; i++) begin : wrapper_12
        mipi_spmi_svt_hdl_interconnect_1
interconnect_wrapper_1(system_if.master_if[j].sdata_en_master[i],
system_if.master_if[i].sdata_en);
      end
    end
  endgenerate

  generate
    for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; j++) begin : wrapper_13
      for (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; i++) begin : wrapper_14
        mipi_spmi_svt_hdl_interconnect_1
interconnect_wrapper_2(system_if.master_if[j].sdata_en_slave[i],
system_if.slave_if[i].sdata_en);
      end
    end
  endgenerate
```

```
          generate
              for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; j++) begin : wrapper_31
                for  (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; i++) begin : wrapper_32
                  mipi_spmi_svt_hdl_interconnect_2
          interconnect_wrapper_5(system_if.master_if[j].sclk_en_master[i],
          system_if.master_if[i].sclk_en);
                end
              end
            endgenerate
          //Connect the debug interfaces of Slave VIP components
          generate
              for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; j++) begin : wrapper_21
                for  (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; i++) begin : wrapper_22
                  mipi_spmi_svt_hdl_interconnect_1
          interconnect_wrapper_3(system_if.slave_if[j].sdata_en_master[i],
          system_if.master_if[i].sdata_en);
                end
              end
            endgenerate

            generate
              for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; j++) begin : wrapper_23
                for  (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; i++) begin : wrapper_24
                  mipi_spmi_svt_hdl_interconnect_1
          interconnect_wrapper_4(system_if.slave_if[j].sdata_en_slave[i],
          system_if.slave_if[i].sdata_en);
                end
              end
            endgenerate

          generate
              for (j=0; j<`SVT_MIPI_SPMI_MAX_NUM_SLAVES; j++) begin : wrapper_41
                for  (i=0; i<`SVT_MIPI_SPMI_MAX_NUM_MASTERS; i++) begin : wrapper_42
                  mipi_spmi_svt_hdl_interconnect_2
          interconnect_wrapper_6(system_if.slave_if[j].sclk_en_master[i],
          system_if.master_if[i].sclk_en);
                end
              end
            endgenerate
```

- ❖ Connect top-level SPMI system interface to Virtual interface of SPMI system environment.

```
          uvm_config_db#(svt_mipi_spmi_system_vif)::set(uvm_root::get(),
          "uvm_test_top.env","system_vif", system_if);
```

The `uvm_config_db` command connects the top-level MIPI SPMI interface to the virtual interface of the MIPI SPMI system environment. The `uvm_test_top` represents the top-level module in the UVM environment. The `env` is an instance of your testbench environment. The `system_env` is instance of MIPI SPMI system environment .

2.1.2 Instantiating and Configuring the VIP

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

❖ Create a customized MIPI SPMI system configuration class by extending the MIPI SPMI system configuration class (`svt_mipi_spmi_system_configuration`) and specifying the required configuration parameters.

For example:

```
class cust_svt_mipi_spmi_system_configuration extends
svt_mipi_spmi_system_configuration;
  function new(string name =
"cust_svt_mipi_spmi_system_configuration");
    super.new(name);

    /** Set number of masters and slaves */
    this.num_masters = 4;
    this.num_slaves = 4;

    /** Set the Master/Slave ID's and clock frequency
    foreach(this.master_cfg[i]) begin
      this.master_cfg[i].master_id = i;
      this.master_cfg[i].t_bt_bito = t_bt_bito;
      this.master_cfg[i].t_sdataz = $urandom_range(0,9);
      this.master_cfg[i].t_sclk = rand_clk_gen.t_sclk[i];
this.master_cfg[i].enable_data_and_sclk_en_signal_checks = 1;
    end

    foreach(this.slave_cfg[i]) begin
      this.slave_cfg[i].slave_id = i;
      this.slave_cfg[i].t_bt_bito = t_bt_bito;
      this.slave_cfg[i].t_sdataz = $urandom_range(0,9);
     this.slave_cfg[i].enable_data_and_sclk_en_signal_checks
= 1;
    end

   /** Enabling coverage for Master/Slave */
    for(int i=0; i<this.num_masters; i++) begin
      this.master_cfg[i]=new();
      this.master_cfg[i].enable_device_cov = 1;
      this.master_cfg[i].enable_device_xml_gen = 1;
      this.master_cfg[i].enable_device_chk_pass_cov = 1;
      thiss.master_cfg[i].enable_device_chk_fail_cov = 1;
    end
    for(int i=0; i<this.num_slaves; i++) begin
      this.slave_cfg[i]=new();
      this.slave_cfg[i].enable_device_cov = 1;
      this.slave_cfg[i].enable_device_xml_gen = 1;
      this.slave_cfg[i].enable_device_chk_pass_cov = 1;
      this.slave_cfg[i].enable_device_chk_fail_cov = 1;
    end
  endfunction
endclass
```

For more information on the configuration class, see the `svt_mipi_spmi_configuration`, `svt_mipi_spmi_agent_configuration` and `svt_mipi_spmi_system_configuration` *Class References* at the following locations:

```
$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/
mipi_spmi_svt_uvm_class_reference/html/
class_svt_mipi_spmi_configuration.html

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/
mipi_spmi_svt_uvm_class_reference/html/
class_svt_mipi_spmi_agent_configuration.html

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/
mipi_spmi_svt_uvm_class_reference/html/
class_svt_mipi_spmi_system_configuration.html
```

❖ Construct the customized MIPI SPMI system configuration and pass the configuration to the MIPI SPMI system environment (instance of svt_mipi_spmi_system_env) in the build phase of your testbench environment.

```
cfg =
cust_svt_mipi_spmi_system_configuration::type_id::create("cfg
");

uvm_config_db#(svt_mipi_spmi_system_configuration)::set(this,
    "system_cfg", "cfg", cfg);
```

The `cust_svt_mipi_spmi_system_configuration` is the customized MIPI SPMI system configuration as defined in the previous step. The `cfg` is an instance of this configuration.

2.1.3 Creating a Test Sequence

The VIP provides a base sequence class for the MIPI SPMI Master and Slave agents (`svt_mipi_spmi_transaction_sequence_collection`). You can extend this base sequence to create test sequences for the MIPI SPMI Master and Slave agents.

For more information on the MIPI SPMI base sequence, and the VIP sequence collection, see the Sequence Page of the *VC VIP MIPI SPMI Class Reference* at the following location:

```
$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/
mipi_spmi_svt_uvm_class_reference/html/sequencepages.html
```

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

```
$DESIGNWARE_HOME/vip/svt/mipi_spmi_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 respective MIPI SPMI sequences.

```
class base_test extends uvm_test;

//Build phase
env = mipi_spmi_svt_basic_env::type_id::create("env", this);

 //Set the required sequence to the virtual sequencer
  uvm_config_db#(uvm_object_wrapper)::set(this, "env.sequencer.main_phase",
"default_sequence", mipi_spmi_svt_default_virtual_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_design_dir_path/include/sverilog
+incdir+project_design_dir_path/src/sverilog/simulator
```

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

For example,

```
+incdir+/home/project/design_dir/vip/include/sverilog
+incdir+/home/project/design_dir/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 MIPI SPMI:

```
+define+SVT_UVM_TECHNOLOGY
+define+UVM_DISABLE_AUTO_ITEM_RECORDING
+define+SYNOPSYS_SV
```

Macro	Description
SVT_UVM_TECHNOLOGY	Specifies SystemVerilog based VIPs that are compliant with UVM
UVM_DISABLE_AUTO_ITEM_RECORDING	Disables the UVM automatic transaction begin and end event triggering and recording
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_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 MIPI SPMI

VC VIP UVM Installation and Setup Guide:

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

VC VIP MIPI SPMI UVM User Guide:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/mipi_spmi_svt_uvm_user_guide.pdf

VC VIP MIPI SPMI UVM Getting Started Guide:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/mipi_spmi_svt_uvm_getting_started.pdf

VC VIP MIPI SPMI Class Reference:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/doc/mipi_spmi_svt_uvm_class_reference/html/index.html

VC VIP MIPI SPMI Verification Plans:

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

A.3 Example Home Directory

Directory that contains a list of VIP example directories:

$DESIGNWARE_HOME/vip/svt/mipi_spmi_svt/latest/examples/sverilog

View simulation options for each example,

```
        gmake help
```