

Introduction

In this demo we will demonstrate the simplest usage of the specification coverage package. This usage is not intended for projects where strict requirement to testcase mapping is required, but it can still be a helpful tool for keeping track of requirements. For more advanced features of the specification coverage package, please see the `advanced_usage` demo or the QuickRef.

Background Information

This basic example of the Specification Coverage concept will demonstrate how the functionality can be used to keep track of the requirements in a simple UART testbench. The testbench is based on a simplified version of the testbench available in the `bitvis_uart` example. The UART DUT is located under `bitvis_uart/src/`. For this example, the following requirements from the “customer” are used:

| Requirement | Description |
|-------------|---|
| FPGA_SPEC_1 | The default of the module shall be as follows: <ul style="list-style-type: none">- RX_DATA: 0x00- TX_READY: 0x01- RX_DATA_VALID: 0x00 |
| FPGA_SPEC_2 | Data written to the TX_DATA register shall be transmitted by the UART TX interface |
| FPGA_SPEC_3 | Data received by the UART RX interface shall be made available in the RX_DATA register, accessible over SPI |
| FPGA_SPEC_4 | The module shall handle simultaneous operation of UART transmit and receive. |

The information in this table is added to the `req_list_basic_demo.csv` file.

Running the demo

The demo can be run by executing the python script `run_basic_demo.py` from the `script/` directory:

```
>>python run_basic_demo.py
```

Or from the `sim/` directory:

```
>>python ../script/run_basic_demo.py
```

Note that Python 3.x is required to run this demo-script. The script will compile all the VHDL sources and execute the testbench in the simulator. In this demo all requirements are verified in a single testcase.

Once the VHDL testbench has completed, the `run_basic_demo.py` script will call the `run_spec_cov.py` script automatically. The script will be called as follows:

```
>>python run_spec_cov.py -r ../demo/basic_usage/req_list_basic_demo.csv
-i ../sim/partial_cov_basic_demo.csv
-s ../sim/spec_cov_basic_demo.csv
```

After reading all the input files, the script will go through the data and evaluate each requirement as compliant or non-compliant. The results of this evaluation are written to the specification coverage file, which is stored under `/sim/spec_cov_basic_demo.csv`.