CX-301: Project

UVM based Verification of a System-on-Chip (SoC)

Version 1.1

Information contained within this document is for the sole readership of the recipient, without authorization of distribution to individuals and / or corporations without prior notification and approval.

Copyright 2024 © ChiPXPRT Pvt. Ltd.

Document History

The changes and versions of the document are outlined below:

Version	State / Changes	Date	Author
1.0	Initial Draft	March, 2025	Qamar Moavia

Copyright 2024 © ChiPXPRT Pvt. Ltd.

Table of Contents

Copyright 2024 © ChiPXPRT Pvt. Ltd.

Objectives

This project will enable you to,

- Better understand how Multi-Interface Designs are verified using UVM.
- Better understand how UVM offers reusability of Verification Components.
- Apply the UVM concepts learned through labs on designs.
- Understand DUT behaviour through the specification document.
- Verify a multi Interface based design using multiple interface UVCs.
- Write Verification plan of any DUT based on its specification document.
- Create UVM based Testbench Architecture to verify multi Interface based design.
- Follow the verification plan to set up the verification environment.
- Write a Diverse Sequences Library that includes enough sequences that could be used in test cases to verify all the features of the design mentioned in the Specification Document.

Tools

- SystemVerilog
- Synopsys VCS

Instructions for the Project

The final version of design is not yet ready for verfication. Once rtl is ready, it can be found on the shared folder at the following location:

```
share_folder/CX-301-DesignVerification/Project/P1
```

Submit Verification plan and all the UVM testbench files. Make sure the file structure we used in the UVM labs is strictly followed. All the UVCs must be in a separate directory, with each having the sv and tb directories.

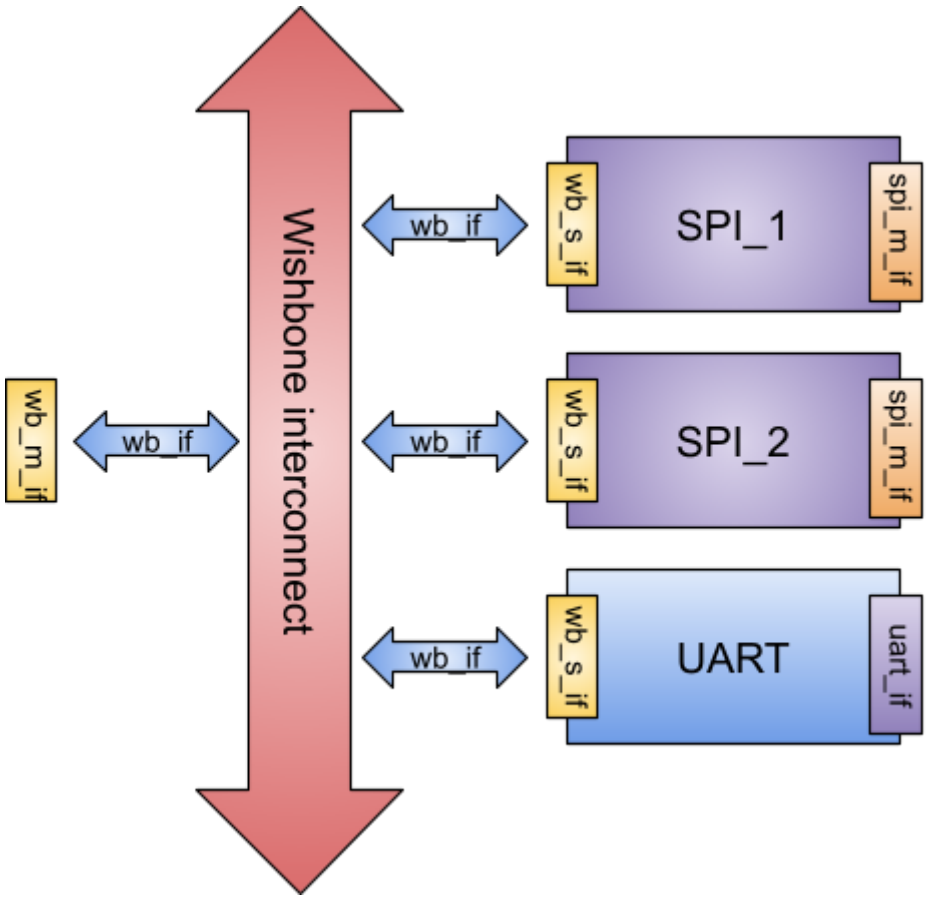
Copyright 2024 © ChiPXPRT Pvt. Ltd.

1. DUT Overview

This project focuses on verifying a System-on-Chip (SoC) design that includes:

- A **Wishbone Interconnect** serving as the on-chip bus
- Two **SPI** peripherals (SPI_1 and SPI_2)
- One **UART** peripheral

All peripherals communicate with the interconnect(or we can call it a bus) via their respective Wishbone slave interfaces. Additionally, each SPI block exposes an external SPI interface, while the UART block provides a standard UART interface.



Copyright 2024 © ChiPXPRT Pvt. Ltd.

2. Subcomponent Descriptions

2.1 Wishbone Interconnect

The Wishbone Interconnect is responsible for routing Wishbone transactions from a master to the appropriate slave. It handles address decoding, read/write data paths, and ensures proper handshaking between the Wishbone master and multiple slave devices.

- **Specification Document**: *[WB Interconnect Spec].*

2.2 SPI Peripherals

Each SPI block (SPI_1, SPI_2) is exactly similar. They are instances of the same SPI module. This SPI module was taken from the OpenCores and it implements a standard Serial Peripheral Interface protocol. The SPI module is configurable with many different configuration options.

- **Reference Design**: Based on OpenCores implementation
- **Specification Document**: *[SPI Spec]*

2.3 UART Peripheral

The UART block provides asynchronous serial communication. It supports programmable baud rates, data bits, stop bits, and parity configuration.

- **Reference Design**: Based on OpenCores implementation
- **Specification Document**: *[UART Spec]*

Copyright 2024 © ChiPXPRT Pvt. Ltd.

3. Project Tasks

3.1 Verification Plan

Develop a concise Verification Plan outlining your strategy to validate the SoC design. This plan should detail:

- **Goals:** Verify protocol compliance, data integrity, and system-level testing.
- **Testbench Architecture:** Use a UVM-based testbench with dedicated UVCs for the Wishbone, SPI, and UART interfaces.
- **Methodology:** Employ both constrained random and directed tests, including multi-channel sequences and diverse test cases that you are planning to test DUT on.
- **Metrics & Schedule:** Define key coverage metrics, milestones, and deliverables (simulation results, coverage reports, etc.).

Each trainee group must submit this plan first for review. Then start working on the UVC developments.

3.2 Wishbone UVC Development

Both trainee groups (each consisting of three members) will initially work on creating a UVM Verification Component (UVC) for the Wishbone interface.

Key Requirements

- The Wishbone Interface UVC should include both master and slave agents.
- The Wishbone Interface UVC should be parameterized so that the data length and the address length can be changed depending upon the DUT we are connecting it to. As some designs may have 8-bit, 16-bit or 32-bit data ports etc. Same is the case for address as well.
- Having both agents within one UVC allows standalone testing even before the actual DUT is ready.
- The master agent drives the Wishbone interface, while the slave agent can act as a “dummy DUT” for early functional checks.
- The UVC should be highly configurable, enabling different usage modes:
    - Master only (if DUT includes the slave interface)

Copyright 2024 © ChiPXPRT Pvt. Ltd.

- Slave only (if DUT includes the master interface)
- Master + Slave (for early testing without the real DUT)
- Passive mode (monitor only), e.g., for observing signals between the interconnect and any Wishbone slave device.

You will be using the Master only, Master + Slave and the Passive mode in the project.

3.3 SPI UVC and UART UVC Development

Following the successful development of the Wishbone UVC, the next phase focuses on building dedicated UVCs for the peripheral interfaces:

SPI UVC Development (Group A)

The SPI UVC will incorporate both master and slave agents.

- The master agent will generate SPI transactions, initiating communication and driving protocol-specific operations.
- The slave agent will simulate the peripheral's responses, enabling independent and early testing of the SPI interface even before the actual DUT integration.

This UVC is responsible for handling SPI protocol transactions, including data transfers, clock management (polarity and phase), clock rates, and chip select operations. It will monitor signal activity to verify timing and data integrity.

UART UVC Development (Group B)

The UART UVC will be designed with both transmitter and receiver agents.

- The transmitter agent will drive asynchronous serial communication by sending data according to the UART protocol.
- The receiver agent will capture and verify incoming data, ensuring that the transmitted information is accurately received.

The UART UVC will cover all aspects of UART communication, including the configuration of baud rates, data bits, stop bits, parity settings and fifo functionality.

Copyright 2024 © ChiPXPRT Pvt. Ltd.

It will closely monitor the signal flow to detect framing or parity errors and ensure compliance with timing requirements.

Both UVCs are designed to manage protocol-level transactions, monitor signal activities, and provide the necessary coverage metrics for thorough functional verification. These components will later be integrated into the overall testbench environment to interact with the Wishbone interface, ensuring that the complete system behaves as expected.

3.4 Standalone Verification of SPI and UART Peripherals

Group A: Once you have completed the Wishbone UVC and the SPI UVC then the next step for you would be to use these two interface UVCs to test the SPI Module. You will need a reference model of the SPI Module in the Scoreboard to test it properly.

Group B: Once you have completed the Wishbone UVC and the UART UVC then the next step for you would be to use these two interface UVCs to test the UART Module. You will need a reference model of the UART Module in the Scoreboard to test it properly.

3.5 Clock and Reset UVC

The clock and reset is also required to better control the clock and reset generation through the sequences instead of relying on the top module for clock and reset generation.

3.6 Integrated Verification Environment

In the final phase, all independently developed UVCs—Wishbone, SPI, and UART—are unified within a single UVM testbench to verify the system-level behavior. The top-level environment instantiates each UVC and manages their configuration (using as a master, as a salve, or as a passive monitor only), ensuring proper interaction between the interconnect and peripheral interfaces.

Copyright 2024 © ChiPXPRT Pvt. Ltd.

This integrated setup will then require multi-channel sequences that simulate realistic, concurrent operations, where transactions across different protocols occur simultaneously. By doing so, the testbench emulates the complex scenarios that the design will encounter in real-world applications.

A comprehensive scoreboard is to be added to monitor and cross-check data as it traverses from the Wishbone master through to the SPI and UART endpoints, ensuring protocol compliance and data integrity.

Scoreboard should have the correct reference model of the SoC with which it can validate the data going to different interfaces (SPI, UART, WB), and check if it's correct or not.

Good Luck 🙂

Copyright 2024 © ChiPXPRT Pvt. Ltd.