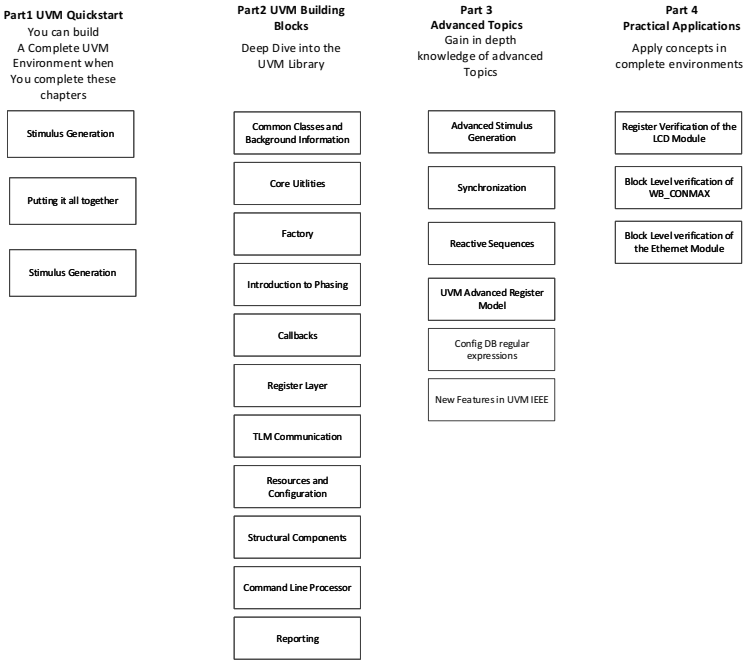

Fig. 0.1: Chapters in this book

Part 1: UVM Quickstart

This part is a UVM Tutorial. There are three chapters in this part. A simple DUT is used to walk you through a Verilog and a SystemVerilog in the first chapter.

The second chapter attempts to bring together concepts to put together a complete verification environment in UVM. It discusses in practical detail various aspects of a UVM environment and connects master and slave verification components through a simple pass-through DUT.

The Wishbone protocol is chosen instead of the UBUS protocol described in the UVM Users Guide. While it would have been possible to extend the UBUS protocol description to allow some other transfers and create examples using actual RTL with this protocol, such an effort would be rather significant. This book instead leverages RTL IP that has been proven elsewhere to teach UVM so that you can continue learning and exploring well-documented IP.

The third chapter goes over stimulus generation. The creation and driving of stimuli form an important activity in any verification effort. The UVM class library allows the user to write reusable code and provides some guidelines for how to structure this code. This chapter covers various concepts and considerations for stimulus generation.

Completing this Part should give you a good idea of what the UVM library contains and how it operates. If you know UVM, you can choose to skip this part, or delve deeper if you need to.

Part 2: UVM Building Blocks This part is a deep dive into the various building blocks in the UVM library. Read this part to get detailed knowledge of the library. The components and concepts developed in this chapter are reused in Part IV.

Part 3: Advanced Topics

Part3 goes into advanced topics in UVM that are not covered in earlier parts of the book. Content in this part assumes you have studied the earlier parts. Some of the primary areas covered are:

- Synchronization among processes in UVM
- Heartbeat applications to ensure your testbench is running
- Reactive sequencer applications in UVM
- Advanced Register topics
- Regular expressions in the UVM configuration database.
- Primer on Config DB regular expressions
- Advanced Core Utilities using IEEE 1800.2

Upon completion of this part of the book, you should be able to use UVM to verify your designs and tweak your environments to accomplish your specific goals.

Part 4: Block Level Verification Environments

The focus of this book is practical knowledge that you can use daily. UVM can be used to build reusable testbenches at various levels of design integration. Part IV of the book illustrates practical applications of UVM to validate several cores that are part of a SOC.

These cores mirror commonly available commercial designs. Each of the designs presented here is used to illustrate a specific aspect of the UVM library. Detailed exercises at the end of each chapter are intended to help the user to delve deeper into the design and UVM.

As some RTL cores with Verilog environments are available from www.opencores.org, it made it possible for readers who are coming into ASIC verification from a design world to relate more quickly to the concepts in UVM using these cores. Hence, all environments are built around these available cores merged into an example platform.

You may also choose to add other RTL cores to this design or delve into a variety of other topics in verification using these examples as a platform. All the cores utilize the Wishbone Protocol [18] to communicate between them. The master and slave agents developed in the previous chapters are reused in this part to verify the design.

The GitHub repository comes with all the PDF documents that describe the core. A brief description of each core is included in the relevant chapters so that you can get the context of the functionality of the core before exploring the verification environment and UVM features.

Each of the Verilog testbenches and UVM testbenches provides a Makefile. This Makefile helps compile the design with appropriate switches for the VCS simulator from Synopsys.

Note: While complete environments have been provided to you to learn and extend from, ***it has never been my intention to either verify the core provided nor delve into details of each core in this book.*** Such an exercise is left to you, hoping that you undertake it to explore deeper into UVM at your own pace. The provided RTL is merely a vehicle to allow you to learn.