20.1 Introduction

Functional verification comprises a large portion of the resources required to design a complex system. To minimize wasted effort, coverage is used as a guide for directing verification resources by identifying tested and untested portions of the design.

Coverage is defined as the percentage of verification objectives that have been met. It is used as a metric for evaluating the progress of a verification project in order to reduce the number of simulation cycles spent in verifying a design

Broadly speaking, there are two types of coverage metrics. Those that can be automatically extracted from the design code, such as code coverage, and those that are user-specified in order to tie the verification environment to the design intent or functionality. This latter form is referred to as *Functional Coverage*, and is the topic of this section.

Functional coverage is a user-defined metric that measures how much of the design specification has been exercised. It can be used to measure whether *interesting* scenarios, corner cases, specification invariants or other applicable design conditions—captured as features of the test plan— have been observed, validated and tested.

The key aspects of functional coverage are:

- It is user-specified, and is not automatically inferred from the design
- It is based on the design specification (i.e., its intent) and is thus independent of the actual design code or its structure.

Since it is fully specified by the user, functional coverage requires more upfront effort (someone has to write the coverage model). Functional coverage also requires a more structured approach to verification. Although functional coverage can shorten the overall verification effort and yield higher quality designs, these shortcomings can impede its adoption.

The SystemVerilog functional coverage extensions address these shortcomings by providing language constructs for easy specification of functional coverage models. This specification can be efficiently executed by the SystemVerilog simulation engine, thus, enabling coverage data manipulation and analysis tools that speed up the development of high quality tests. The improved set of tests can exercise more corner cases and required scenarios, without redundant work.

The SystemVerilog functional coverage constructs enable:

- Coverage of variables and expressions, as well as cross coverage between them.
- Automatic as well as user-defined coverage bins.
- Associate bins with sets of values, transitions, or cross products.
- Filtering conditions at multiple levels.
- Events and sequences to automatically trigger coverage sampling.
- Procedural activation and query of coverage.
- Optional directives to control and regulate coverage.