Chapter 2

CURRENT LANDSCAPE IN DESIGN AND VERIFICATION

Before delving into error-repair techniques, we are going to review how digital circuits are developed and verified first. In this chapter we describe current flows for front-end design, back-end logic design, back-end physical design and post-silicon debugging. We also discuss the bugs that may appear at each design stage, as well as the current verification and debugging methodologies that attack them.

2.1 Front-End Design

Figure 2.1 illustrates the current front-end design flow. Given a specification, typically three groups of engineers will work on the same design, including architecture design, testbench creation and RTL development\(^1\). The flow shown in Figure 2.1 uses simulation-based verification; however, flows using formal verification are similar. Chapter 3 provides more detailed discussions on these verification methods.

In this design flow, the architecture group first designs a high-level initial model using high-level languages such as C, C++, SystemC, Vera [163], e [150] or SystemVerilog. At the same time, the verification group develops a testbench to verify the initial model. If verification fails, the testbench and/or model need to be corrected, after which their correctness is verified again. This process keeps repeating until the high-level model passes verification. At this time, a golden high-level model and testbench will be produced. They will be used to verify the RTL initial model developed by the RTL group. If verification passes, an RTL golden model will be produced. If verification

\(^1\) Although there may be other groups of engineers working on other design aspects, such as power, we do not consider them in this design flow.
fails, the RTL model contains bugs and must be fixed. Usually, a bug trace that exposes the bugs in the RTL model will be returned by the verification tool.

To address the debugging problem, existing error-repair techniques often partition the problem into two steps. In the first step, the circuit is diagnosed to identify potential changes that can alter the incorrect output responses. In the second step, the changes are implemented. The first step is called error diagnosis, and the second step is called error correction. Currently, functional error diagnosis and correction are often performed manually using the steps described below. This manual error-repair procedure is also shown in the “Manual functional error correction” block in Figure 2.1.

1 The bug trace is minimized to reduce its complexity for easier error diagnosis.

2 The minimized bug trace is diagnosed to find the cause of the bugs. Debugging expertise and design knowledge are usually required to find the cause of the bugs.

Figure 2.1. The current front-end design flow.