Truss Flow

CHAPTER 7

Expensive solutions to all kinds of problems are often signs of mediocrity.

Ingvar Kamprad, founder of IKEA

Have you ever bought and assembled a piece of furniture from IKEA? In the store most of their furniture looks very simple, but when you get it home and try to assemble it, you realize that it's built from several smaller and often confusing pieces. Even with IKEA's famous assembly instructions, showing the "intent" for each piece graphically, assembly can still be confusing. Imagine how hard it would be without instructions.

The authors have had to learn many verification environments through the years, and this has often been a very confusing experience. What seems like a great concept with a well-defined structure at a high level of abstraction is often obscured by troublesome details when you first try to implement it. Many times the confusion is increased because of a lack of description regarding how the high-level ideas are actually implemented. To help reduce the confusion around Truss, this chapter describes the *"dance"* in more detail.

Overview

This chapter looks at how the "dance" described in the preceding chapter is actually implemented. It shows the order in which each method is called, and describes the files to find the method, or its base. The chapter then looks at the structure for the major components of Truss.

First to be described is `verification_top()`, the first function called in Truss and the base of the "dance." Following this is a description of the methods, and their class, through which files are called for each step.

Then the test component is described. This component follows a dance similar to that of `verification_top()`, but for a different set of classes and files.

The `irritator` class is described next. While similar to a `test_component`, irritators have some unique method calls worth pointing out.

The last part of the chapter talks about steps that need to be taken to build a new Truss project, by taking the more-abstract description of classes and applying them to the first few tests in a new project.

About verification_top.cpp

When the simulator executes the `$teal_top` call in the HDL, control is passed to the `verification_top()` function in `verification_top.cpp` under the *truss* directory. In this handbook we refer to this function as the "dance," or top function. It is this function that interacts with your top-level components: the test, the testbench, and the watchdog timer.

Let's look at the dance with respect to the methods you have to write. This is illustrated in the figure on the following page. A square box indicates that the method has a default implementation, and a rounded box indicates it needs to be defined for your project.