In today's world, it is often required by the members of a design team to share their design information with people outside the organization. This process of sharing valuable design data with the outside world — third party vendors, OEM manufacturers, contractors etc. — is unavoidable in most cases. Functional Models, also known as Bus Functional Models (BFM), play an important role in protecting intellectual property in such circumstances. PLI can be used to take advantage of the easy-to-program aspect of C together with rich simulation features of Verilog to produce functional models. In this chapter, we will find out the reasons behind making PLI as the choice for writing such models. Our first example of a simple crossbar switch will illustrate how to write a functional model. Once this gives the basic idea, we will move to designing a functional model of a rudimentary processor. From these examples, it will become apparent why PLI is the only choice for writing such a model.

1. WHAT IS A BUS FUNCTIONAL MODEL?

The primary objective of any hardware description language is to describe a design concept that will eventually transform itself into a piece of silicon. However, at different stages of the design, depending on what level this concept is being conveyed to next, the description itself may take different shapes. For example, the description during the initial phase of the design when the designer needs to check the validity of the concept itself, is different from the description at the final phase when it is inputted into a
synthesis tool to obtain an equivalent gate structure. To summarize, as the design cycle matures, the description or the model also changes.

Depending on which phase of the design cycle they apply to, there are three approaches to write a model. They are:

- Behavioral/Bus Functional model
- RTL model
- Structural model.

We will discuss each of them separately below.

1.1 Behavioral / Functional Model

A behavioral or functional model allows you to use many high level constructs, such as nested loops, conditional statements, etc., which are common to any high-level language, such as C. Functional models are not synthesizable. The only objective of writing such a model is to achieve a port behavior which is exactly the same as what is desired from the actual silicon. How this behavior is achieved inside the model is not important. Or in other words, functional models, like a black box, hide the internal details of the implementation and derive only the results that will appear at the pins of the module. For this reason, functional models are often termed as Bus Functional Models (BFM). These types of models are typically used at the early phase of the design to prove the algorithm. They are also used in places where a gate level model is not required (such as designing a stub etc.).

1.2 Register Transfer Level (RTL) Model

Modeling at the register transfer level (RTL) also allows one to use high level constructs to some extent. But, at this stage, the main intention of the designer is to get a synthesized version after translating the design through a proper synthesis tool. Almost all synthesis tools impose certain rules/restrictions on coding methodology which must be adhered to in order to use that tool. From a designer's perspective, RTL is an intermediate stage of the design which eventually will lead to a gate level design.

1.3 Structural Model

A structural model is made up of primitive cells and interconnections between them. In this case the main intention is to get an application specific integrated circuit (ASIC) targeted to the technology of a specific foundry and as a result, all primitive cells used in the model must be defined in the